Proceedings

1997 Symposium on Document Image Understanding Technology

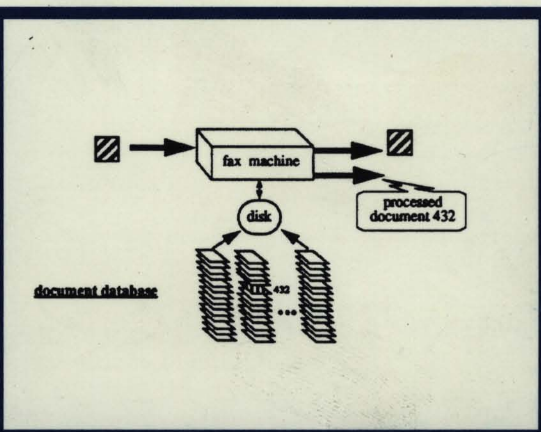
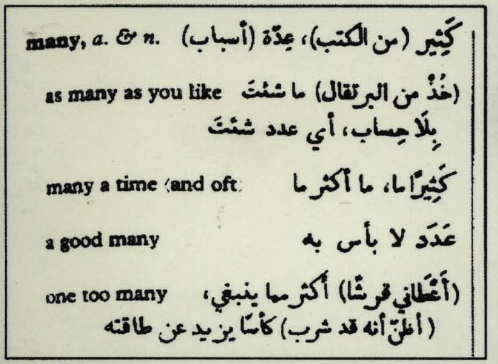
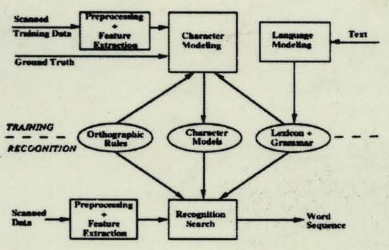


Image Retrieval



many, *a.* & *n.*	كَثِير (من الكتب)، عِدّة (أسباب)
as many as you like	(خُذْ من البرتقال) ما شئتَ بِلَا حِساب، أي عدد شئتَ
many a time (and oft)	كَثِيرًا ما، ما أكثر ما
a good many	عَدَد لا بأس به
one too many	(أعطاني قرشًا) أكثر مما ينبغي، (أظنّ أنه قد شرب) كأسًا يزيد عن طاقته

Scripts and Languages



Recognition and Architectures

April 30 - May 2, 1997
Historic Inns of Annapolis
Annapolis, Maryland

Proceedings

SDIUT97

The 1997 Symposium on Document Image Understanding Technology

Annapolis, Maryland
Historic Inns of Annapolis
April 30-May 2, 1997

Sponsored by: The United States Department of Defense
Organized by: The Laboratory for Language and Media Processing
University of Maryland
Institute for Advanced Computer Studies
College Park, MD 20742

A limited number of additional copies of this proceedings is available for $50 from:
University of Maryland
Institute for Advanced Computer Studies
College Park, MD 20742
Phone: (301) 405-4526
Fax: (301) 314-9115
email:sdiut97@documents.cfar.umd.edu

Table of Contents

SESSION 9 PERFORMANCE EVALUATION

A Message from the Organizers

On behalf of the Laboratory for Language and Media Processing, the Institute for Advanced Computer Studies and the University of Maryland, I would like to welcome you to the 1997 Symposium on Document Image Understanding Technology.

We have organized this symposium to provide a means for researchers and research sponsors from academia, industry and government to communicate ideas for advancing the field of document image understanding. After a successful workshop in 1993 and a symposium in 1995, we hope that this symposium will continue to provide valuable insight to participants.

This proceedings contains abstracts and papers from 31 technical presentations and two invited talks. The topics range from optical character recognition to document image retrieval and document image databases. Of special interest this year is a focused session on performance evaluation. As the community grows, there is an ever increasing need to be able to formally evaluate progress. The session on evaluation will include general methodologies for evaluation, a talk on the upcoming METREC conference and a panel where we can begin to gain insight into the important issues and challenges facing us in performance evaluation.

There are countless people who deserve special thanks for their hard work in making this Symposium a reality. I would like to especially thank Cecelia Kullman for her work in the role of Symposium Coordinator and the staffs of the Institute for Advanced Computer Studies and Center for Automation Research, for their endless support.

David Doermann
University of Maryland

Keynote Speakers

Reflections on High Volume Image Processing During Tax Season: Image Processing Income Tax Returns in New York

George A. Mitchell, III
New York State Department of Taxation and Finance
Room 205
Albany, New York 12227

Abstract

New York now processes millions of State and City income tax returns through a high volume image processing system developed with its business partner, Fleet Bank.

The tax season is a short, intense processing peak with three chief constraints: first, get refunds out fast; second, deposit remittance checks immediately; and third, do not make mistakes. We are also processing the most hateful of documents, the income tax return. This presentation will focus on how New York meets tax processing objectives within these limitations and constraints, and the advantages and disadvantages of image processing technology in this program.

Biographical Sketch

Mr. Mitchell has been the Chief Information Officer for three New York State agencies. He is currently the Deputy Commissioner for Revenue and Information Management at the Department of Taxation and Finance. In addition to information technology, his Division processes all New York income and business tax returns. Previously he served as Deputy Commissioner for Administration and Information Services at the Department of Social Services. Mr. Mitchell also was Executive Deputy Commissioner for the Division of Criminal Justice Services. Prior to that he served DCJS as General Counsel and Deputy Commissioner for Administration. He also held several positions at the Division of the Budget during a 14 year tenure.

Mr. Mitchell is a member of the Governor's Task Force on Information Resource Management and the Regents Visiting Committee on the State Archives. He was Chairman of the 1996 Eastern States Government Technology Conference and is a past Chairman of the NY Forum on Information Resource Management.

The U.S. Army Gulf War Declassification Project - Triumphs and Challenges

STEVE E. DIETRICH
Lieutenant Colonel
Director, U.S. Army Gulf War Declassification Project
U.S. Total Army Personnel Command

Biographical Sketch

Lieutenant Colonel (LTC) Steve E. Dietrich was born on 29 May 1954 in Delaware. He graduated from the U.S. Military Academy in 1976 and began his active service as an Armor officer. His military education includes the Armor Officer Basic and Advanced Courses, Motor Officer Course, and the U.S. Army Command and General Staff College. He holds a Master of Arts degree in History from Eastern Kentucky University.

LTC Dietrich has filled a variety of command and staff positions, culminating in his current assignment as the Director of the U.S. Army Gulf War Declassification Project (GWDP). LTC Dietrich served as a tank platoon leader, company executive officer, assistant battalion operations officer, and company commander in Kirch Goens, Germany between 1977 and 1981. From 1982 to 1985, he was an assistant professor of military science at Eastern Kentucky University where he taught military history. From 1985 to 1987, he served as a combat developments staff officer at the U.S. Army Training and Doctrine Command headquarters in Fort Monroe, Virginia. LTC Dietrich next served as a historian and analyst at the Center of Military History (CMH) in Washington, D.C. from 1987 to 1989. Following that assignment, he was the deputy community commander and area support team commander at Illesheim, Germany from 1989 to 1992. During the Gulf War, from December 1990 to May 1991, he commanded both the Illesheim community and the 11th Aviation Brigade, Rear. LTC Dietrich returned to CMH where he became chief of the military studies branch. He deployed to Haiti during Operation Uphold Democracy as the senior Army historian, where he pioneered the collection of electronic operational records. LTC Dietrich assumed his current duties on 15 May 1995.

LTC Dietrich recently received the Legion of Merit for his efforts at CMH and with the GWDP. His project won the Association for Information and Image Management International and Kinetic Information 1997 Process Innovation Award for best in Government and was one of two finalists for the coveted Vision Award honoring the technology initiative with the greatest potential for transforming a business or social process. His project has also been selected for permanent archiving as a "national treasure" by the Smithsonian Institution. A frequent public speaker and conference participant, LTC Dietrich also has over 30 publications in scholarly and professional journals, book chapters, and encyclopedias.

His military awards include the Legion of Merit, Meritorious Service Medal, Army Commendation Medal, and Armed Forces Expeditionary Medal.

LTC Dietrich is married to the former Barbara Marie Miller of Newburgh, New York. They have a two children, Breanna and Keith.

Image Retrieval

Document Image Routing and Retrieval

Stephen J. Dennis
U. S. Department of Defense
9800 Savage Road
Fort George G. Meade
Fort Meade, MD 20755-6514

Abstract

The Government is sponsoring research in document image analysis and recognition to produce a number of advanced modular algorithms that are scalable and address information routing and retrieval problems in large heterogeneous corpora. The model for end to end applications involves three processing stages. In the pre-processing stage, our goal is to diagnose the document image based on a shallow analysis of its content. Some examples of useful pre-processing include the determination of script or language; the presence of specific graphic elements; or the document type. Using cursory information, it is possible to progressively extract information from the document with useful accuracy. In the recognition stage, the Government is developing separate graphics and text recognition technologies. For graphics recognition, important targets are word images, logos, and signatures. For text recognition, systems developers are working with existing Optical Character Recognition (OCR) technologies to improve performance for degraded document image corpora. Last year, Government research organizations sponsored the development of a new text recognition approach based on Large Vocabulary Conversational Speech Recognition (LVCSR) technology. The resulting system performs across a variety of languages and image resolutions, yet has an undesirable Out-Of-Vocabulary (OOV) word problem. As a post-processing stage, and to possibly mitigate the OOV problem, attempts have been made to combine the results of multiple text recognizers and apply additional language processing techniques.

The results of progressive document image analysis enables data driven architectures for content based routing. Such architectures are capable of routing newly acquired documents to interested users. For a document image archive, document images can be clustered by metadata for more efficient retrieval based on user queries that span two media, text and image. However, document image routing and retrieval architectures are not well understood. The Government's attempts to build generic retrieval prototypes can be represented by at least three models, with increasing integration of document image analysis and text retrieval components. From simple text indexing of OCR output, to Query by Image Example, there are operating ranges for which systems architecture demonstrate accuracy.

Objective evaluation for document image analysis and recognition components have been adhoc with respect to routing and retrieval applications. Previous evaluation efforts have addressed Roman OCR performance and text retrieval performance at low error levels. Government agencies have used these results to predict the value of document image analysis research, related technologies, and potential products in the context of specific programs. Continued support for document imaging research requires that the community adopt system level evaluation to establish meaningful performance baselines. To leverage resources against common information technology requirements, Government Agencies are working with the National Institutes of Science and Technology to develop formal evaluation criteria for cross media retrieval systems. The METREC (METadata and Text Retrieval Evaluation Conference) will serve as an important baseline through which the Government and Industry can expand requirement's definitions for information access and retrieval.

The Skeleton Document Image Retrieval System

Carl Weir **Suzanne Liebowitz Taylor**
Lockheed Martin C^2 Integration Systems
590 Lancaster Ave.
Frazer, PA 19355

Stephen Harding **Bruce Croft**
Center for Intelligent Information Retrieval
University of Massachusetts
Amherst, MA 01003

Abstract

This paper describes Skeleton, a document image retrieval system whose design and implementation is based upon image analysis and document retrieval technologies incorporated in the IDUS image understanding system and the INQUERY information retrieval system, respectively.

Most of the current Skeleton development effort has been put into establishing indexing and query formulation methodologies in which both words and character subsequences of words (ngrams) are tokenized.

A Web-based interface has been developed for Skeleton which includes a Java image display applet to support word and region highlighting.

1 Overview

The Skeleton architecture supports document image analysis, indexing, and retrieval. The indexing component assumes an image analysis in which text regions are grouped into articles with head and body relations identified and text recognized using conventional OCR technology. Functional role distinctions such as *dateline* and *caption* are not taken advantage of in the current implementation but in future versions of the system they will be used to extend fielded search capabilities.

Both the OCR text output of an indexed unit of text and the corresponding page image containing it can be retrieved by Skeleton. It is assumed that end users will normally want to retrieve page images with appropriate regions highlighted and pointers to other pages of the same document provided. However, our experience has indicated that access to the OCR text output is useful in the search process.

In the next two sections, additional information about the image analysis process incorporated into Skeleton from the IDUS system and the system's basic retrieval capabilities inherited from the INQUERY system are described [1,2]. Following this discussion, Skeleton's Web-based interface is illustrated.

2 Document Image Analysis in Skeleton

Skeleton's image analysis component, which was originally developed for use in the IDUS system, is illustrated in Figure 1.

The image analysis component includes Scan*WorX*, a commercial product developed by Xerox Information Systems which provides segmentation and OCR support.

Image analysis modules developed specifically for use in IDUS which have been incorporated into Skeleton include a logical analyzer, a document classifier and a functional analyzer.

The IDUS image understanding system from which these modules were borrowed provides a sophisticated X-windows user interface for examining image analysis output in a graphical form. Although the IDUS interface is not a component of Skeleton, the ability to use IDUS to evaluate image analysis performance is an attractive option.

2.1 Scan*WorX* Modules

Since Skeleton relies upon Scan*WorX* to perform the initial segmentation of document images, it inherits that component's image input constraints. For Scan*WorX* to work properly, image resolution must be 200x200 dpi, 200x100 dpi, 400x200 dpi, or any RxR (square) dpi resolution, where $250 \leq R \leq 450$.[1] Scan*WorX* requires images to be black text on a white background for the OCR module to work properly. It is possible for the system to invert images which are white text on a black background, but color images need to be preprocessed into a proper format. Although a variety of image input formats are supported by Scan*WorX*,

[1] The Scan*WorX* OCR module can recognize text in images whose resolutions range from fax to 600x600 dpi, but for resolutions outside the previously mentioned ranges, manual segmentation is necessary. Since image analysis in the Skeleton architecture is expected to be an automated process, the system is constrained to processing images in the previously mentioned resolutions.

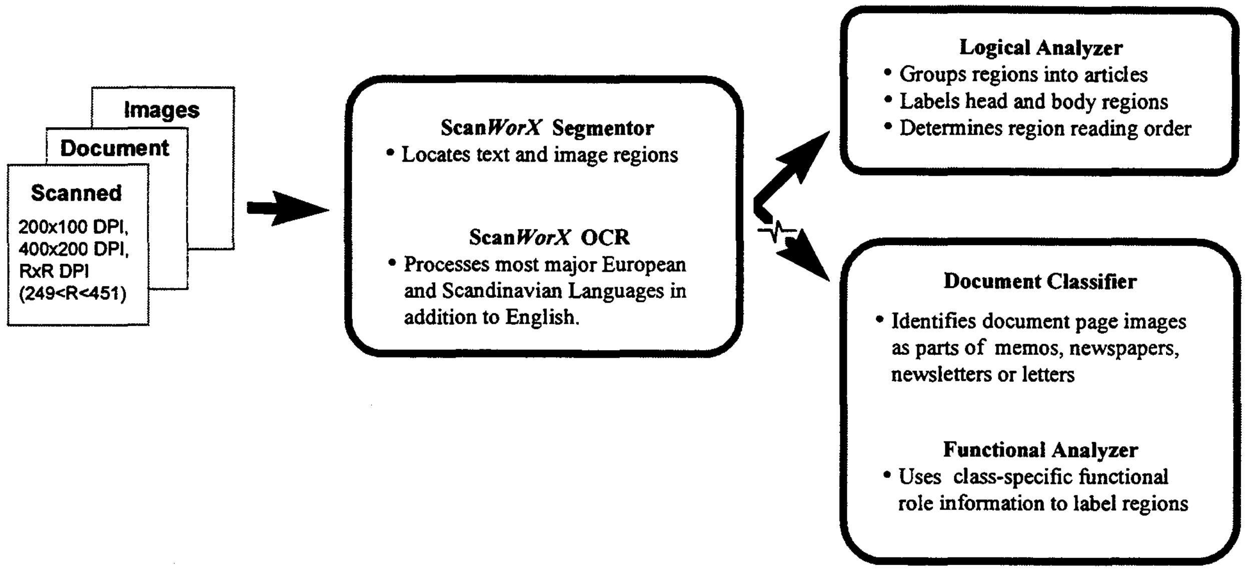

Images
Document
Scanned
200x100 DPI,
400x200 DPI,
RxR DPI
(249<R<451)
ScanWorX Segmentor
• Locates text and image regions
ScanWorX OCR
• Processes most major European and Scandinavian Languages in addition to English.
Logical Analyzer
• Groups regions into articles
• Labels head and body regions
• Determines region reading order
Document Classifier
• Identifies document page images as parts of memos, newspapers, newsletters or letters
Functional Analyzer
• Uses class-specific functional role information to label regions

Figure 1: Skeleton's image analysis components are capable of providing a rich array of information about document structure and content. In the current implementation of Skeleton, information about logical groupings of regions into articles is used to build a document collection. Document class and functional role information is not yet being used in the prototype, but will be incorporated in future implementations to further extend user options in fielded searches.

Skeleton development has been based solely upon the analysis of images in TIFF format.

The Scan*WorX* segmentation module identifies text and image regions within a document page image. Regions cannot span page boundaries, but can overlap one another.

The Scan*WorX* OCR module is capable of processing most major European and Scandinavian languages in addition to English [4].

The segmentation module provides a listing of the regions which have been identified, with the location of each region described in terms of four integers: the first two integers represent *X-Y* coordinate values and the second two represent the width and height of the region, respectively. The type of the region is also specified.

Separate output files are written which contain the OCR results for each text region. To support highlighting in retrieved images a Xerox proprietary output format called XDOC is used which represents information about the location of characters in a document page image, as well as character and word recognition confidence scores.[2]

[2]The character and word recognition confidence scores are not yet being used in Skeleton. In future work on the system, experiments which factor this information into the search process will be conducted. For more information on the XDOC format, see [5].

2.2 Logical Analyzer

The segmentation of a document page image into regions defines a *geometric* tree structure, and the goal of the logical analyzer is to derive from this geometric tree structure a *logical* tree structure which identifies clusters of regions functioning as articles. This process involves the labeling of regions as *head* and *body* constituents: head regions are interpreted as titles and body regions are interpreted as constituent columns of text. An important by-product of logical analysis is the determination of reading order. The analysis method used by this module is influenced by the work of Tsujimoto and Asada [6].

Output data provided by the logical analysis component includes a sequence of entries which describe the text regions delimited by the segmentation module. Each region is described in two such entries, one which describes its role in the geometric tree structure, and one which describes its role in the logical tree structure. These entries declare an index for a region and indicate if it plays a head or body role in the specified tree.

A listing of geometric and logical tree node descriptions is also provided in the component's output. Each node listing contains an integer value specifying how many regions are subsumed by the node followed by the indices for those regions. The reason for the indices declared in the text region entries is to facilitate this cross referencing.

During logical analysis, text regions which have been determined to be in the same column and exhibit some vertical overlap are merged into a single, composite region. A listing of such mergers is provided in the logical analysis component's output.

2.3 Document Classifier

The document classification module is currently able to distinguish business letters, memos, newspapers, and newsletters.

It is possible to stipulate a document class when Skeleton is asked to process a collection of document images. This is often a useful feature, since in many cases a document image collection consists of documents which are of the same type.

2.4 Functional Analyzer

Like the logical analyzer, the functional analyzer relies upon the segmentation module to identify text and image regions. However, unlike the logical analyzer, it also is dependent upon the document classifier. Given knowledge about the possible functional roles played by regions in documents of a given class, the functional analyzer attempts to infer the functional roles played by the regions which have been identified by the segmentation module.

Two different functional analysis methodologies have been implemented:

1. A *content-based* approach which focuses on the string matching of keywords associated with particular components and is thus dependent on good OCR accuracy
2. A *geometric-based* approach, which is OCR independent and relies only upon segmentation and document classification.

Evaluations involving English language business letters have shown that the geometric approach executes almost twice as fast as the content-based approach on the same document with little degradation in performance. However, since Skeleton relies upon OCR processing for document indexing anyway, the better processing speed of the geometric-based approach cannot be taken advantage of, and given that this is true, pursuing a content-based functional analysis methodology is more appropriate, since this method permits a richer document representation which may prove useful in future evaluations of functional analysis performance involving larger collections of document images and a greater variety of document classes.

The output of functional analysis includes an indication of the class of a document and a listing of regions associated with functional labels.

During functional analysis, as in logical analysis, one or more text regions identified by the segmentation module may be merged into a single, composite region. However unlike logical analysis, it is also possible that a single region may be split into multiple regions which are assigned different functional roles. A listing of any mergers or splits is provided in the functional analysis component's output.

3 Document Image Retrieval in Skeleton

Skeleton's information retrieval capabilities inherited from INQUERY are used to build a database of article-sized document representations in which the head and body components of each article recognized by the logical analysis component are labeled with SGML-style `TITLE` and `TEXT` tags. Fielded searches on the `TITLE` and `TEXT` fields within these article-sized documents is supported.

3.1 Indexing Terms using Ngrams

Most of the current research effort on the Skeleton project has been put into the development of an appropriate indexing methodology which takes into consideration the possibility of poor OCR text output without assuming that OCR output will always be poor.

To reduce the degradation in retrieval performance caused by high OCR error rates, image analysis text output is indexed using combinations of full word tokens and a sampling of word token character sub-sequences (`ngrams`). Which particular collection of `ngrams` to extract from a word token is an empirical question now being explored.

In the current implementation an ordered sequence of two, three, four and five-character `ngrams` is extracted, but to reduce the size of the document database a subset of at most eight `ngrams` are selected for indexing purposes.

If the number of `ngrams` in the ordered sequence extracted from a given word token is less than or equal to eight, then all are retained for indexing. Otherwise, the `ngrams` selected for inclusion in the subset are the leading three, the final two and three from "the middle". Letting N represent the number of `ngrams` in an ordered sequence, the first of the "middle" `ngrams` is in position $((N-4)/3)+2$, the second is in position $((N-4)/2)+2$ and the third is in position $(((N-4)/3)+2) \times 2$. The positions of `ngrams` in an ordered sequence are counted beginning with 0, not 1. Figure 2 illustrates this method for determining a subset of `ngrams` to use in document indexing.

The position declared for an `ngram` token in the document database is the same as the position of the word token from which it has been extracted (not its position in the ordered sequence of `ngrams` extracted from the word token). Given that this is the case, an effort is made to replace any duplicate `ngrams` which might arise in the subset selected for indexing once the above method has been used to identify them. The key issues in selecting a subset of `ngrams` is to be consistent and to attempt to arrive at a representative sample.

Example word token: *mexican*

Ordered sequence of two, three, four and five-character `ngrams`:

`me`	`mex`	`mexi`	`mexic`	`ex`	`exi`	`exic`	`exica`	`xi`	`xic`	`xica`	`xican`	`ic`	`ica`	`ican`	`ca`	`can`	`an`
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17

`Ngrams` retained for indexing:

Leading three `ngrams` at positions `0`, `1` and `2`	→	`me mex mexi`
Trailing two `ngrams` at positions `16` and `17`	→	`can an`
First of "middle three" `ngrams` at position `((`N`-4)/3)+2 = 7`, where N=`18` and division of `14` by `3` is rounded up	→	`exica`
Second of "middle three" `ngrams` at position `((`N`-4)/2)+2 = 9`	→	`xic`
Third of "middle three" `ngrams` at position `(((`N`-4)/3)+2) x 2 = 14`	→	`ican`

Figure 2: Method for determining which `ngrams` of a word token to select for indexing

3.2 Query Formulation

An important component of the information retrieval capabilities Skeleton has inherited from INQUERY is a query formalism which permits the use of operators to express more precisely relationships among search query terms and how they contribute to the likelihood of a match between the search query and a given document.

The following query operators commonly occur in Skeleton development:

Sum Operator: `#sum(`$T_1 ... T_n$`)`
Query terms in the scope of a `#sum` operator are treated as having equal influence on the belief in a match between a query and a document.

Weighted Sum Operator: `#wsum(`W_s $W_1 T_1 ... W_n T_n$`)`
Given the specified weight W_s, which declares the degree of belief in a match contributed by the `#wsum` expression, the weights $W_1 ... W_n$ specify the proportion to which the respective query terms $T_1 ... T_n$ in the scope of a `#wsum` operator contribute to that belief.

Ordered Distance Operator: `#`N`(`$T_1 ... T_n$`)`
All the query terms within the scope of an `#`N operator must be found within N words of each other in a document in order for the overall expression to contribute to a belief in a match.

And Operator: `#and(`$T_1 ... T_n$`)`
The more terms in the scope of an `#and` operator which are found in a document, the greater the belief in a match with the query.

Passage Operator: `#passage`N`(`$T_1 ... T_n$`)`
The more terms within the scope of a `#passage`N operator which are found within a passage window of N words in a document, the greater the belief in a match between the query and the document.

Field Operator: `#field(`F R $T_1 ... T_n$`)`
The terms $T_1 ... T_n$ contained in the scope of a `#field` operator are searched for in field F of documents in a given collection. An optional operator R may be specified to indicate a range of values to be searched for in F.

Search queries are generally submitted in plain text, in which case a default format is used to build a structured query which contains query terms identifed in the plain text input.

In Skeleton, the default format takes into consideration the fact that queries may be made against document representations containing OCR-generated text. In this format, which is illustrated in Figure 3, a `#wsum` operator is used to express a relationship between the relative importance of matching against the word tokens in a search query and matching against `ngram` character sub-sequences of them.

The first query term component of the `#wsum` expression is a `#sum` expression containing all the word tokens in the search query. This has the effect of treating all the word tokens as having equal influence on the matching of the query with a given document.

The second query term of the `#wsum` expression is another `#sum` expression whose constituents are `#passage5` expressions. Each of these `#passage5` expressions contains ngram sub-sequences for one of the word tokens in the plain text search query. It is important that the same method is used to determine the ngram samples for each search query token that is used to determine the ngram samples for word tokens in the document database.

The #passage5 operator used in constituent expressions of the second query term provides a window of five word tokens to help catch OCR errors in which whitespace or other word token delimiters might have been introduced. A document which contains all the ngram sequences within the scope of a #passage5 operator will receive a higher ranking than a document that contains only some of them. However, this operation does not over-penalize documents in which some of the search terms are not present; it merely ranks them lower in the list of retrieved documents.

3.3 Retrieval Performance

A series of experiments was run to determine the retrieval effectiveness of various ngram query formulations and database indexing methods. The measurement criterion was best improved performance using a given technique measured by the highest percent improvement in average precision over all recall levels when compared to baseline database and query formulations.

The experiments were performed using four different databases which were randomly degraded using Xerox OCR software error data developed by UNLV [3]. Higher word error rates were used to further degrade some of the randomly degraded databases. In such cases, the selected words were corrupted using typical character error substitutions. The actual word or character error rates achieved for the databases are unknown; the degree of degradation is measured only by how much worse retrieval performance is using standard queries on the degraded collections compared to performance on corresponding non-degraded collections.

A summary of the best retrieval performance results is given in Table 1. Collections with small document sizes are more negatively impacted than collections with larger average document lengths [2]. Thus collections with longer average sized documents have lower degradation levels for a given character or word error rate. Over all databases, the #passage5 operator generally performed the best in binding ngrams together in the ngram component of a structured query, although not always significantly so. It was discovered that the #and operator was not sufficiently strong in joining ngram components, resulting in too many irrelevant documents being assigned high rankings. The #0 operator, on the other hand, was observed to be too demanding in that any document missing one or more ngrams in its scope was assigned too low a retrieval ranking. The query formulations incorporating ngrams improved retrieval effectiveness over standard queries as database degradation increased, but the performance increases remained below retrieval effectiveness for non-degraded data.

Plain Text Query: *Mexican environmental newsletters*

Translated Query:
```
#wsum(10
          9 #sum(mexican environmental newsletters)
          5 #sum(#passage5(me mex mexi exica xic ican can an)
                 #passage5( en env envi ironm onm ment tal al)
                 #passage5( ne new news sl let tt ers rs)))
```

Figure 3: In structured queries formulated from plain text input, extracted word tokens in the first query term component of a #wsum expression are specified to contribute approximately twice as much to the belief of the #wsum expression as the ngram sequences of those tokens in the second component.

	CACM 6	NPL 6	TIME 7	WSJ89 6
Degradation	-26.6	-52.9	-10.9	-19.6
Docs	3204	11429	423	12380
Queries	50	93	83	49
Avg. Words/Doc	64	42	591	512
2-3 gram	#passage5 +8.4	#passage5 +36.0	#0 +3.9	#passage3/5 +4.5
2-5 gram	#passage5 +11.1	#passage5 +35.9	#passage5 +3.0	#passage3 +11.5
Restricted	#passage5 +14.7	#passage5 +38.8	#passage3 +3.8	#passage5 +11.9
Weighted	#passage5 +10.8	#passage5 +38.1	#and +5.1	#passage3 +11.7

Table 1: In this summary of retrieval performance experiments using assorted `ngram` query formulations, the reported values are percentage increases in average precision over all recall levels when the specified operators (`#passage5`, `#passage3`, `#0`, and `#and`) are used.

Indexing two, three, four and five-character `ngrams` generally outperformed query formulations using only two and three-character `ngrams`. Restricting the `ngrams` saved to the database to a subset of eight from the ordered sequence collected tended to improve performance, probably by reducing noise from extraneous `ngrams` and by improving overall term statistics used during query evaluation. However, the primary benefits of using a subset are a significant reduction in database size and improved evaluation speed.

An attempt was made to improve performance using a weighting of `ngram` components based on their positions in a word. Leading `ngrams` were assumed to be more important than middle `ngrams`, which were deemed more important than trailing `ngrams`. Each of the three ngram classes was down-weighted by roughly one half moving from leading to middle to trailing `ngram` samplings within a word. However such `ngram` weighting had no favorable impact on retrieval performance and only added complexity to query evaluation.

3.4 Future Indexing and Query Formulation Work

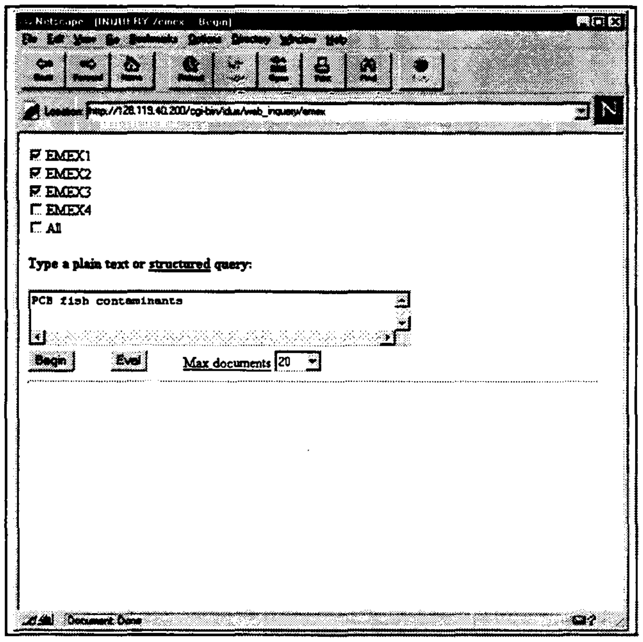
Future work on indexing OCR degraded texts and formulating queries using such databases will concentrate on reducing the size of the `ngram` database by being more selective in choosing which `ngrams` to index on. Furthermore, techniques will be developed to use `ngrams` for query term expansion rather than evaluating the `ngrams` themselves. A formulated query will then contain only matches and near matches of user query terms rather than `ngrams`, thus reducing query processing overhead. Query formulation may also include weightings based on the confidence measures for words produced by the OCR software itself. It is hoped that retrieval performance may continue to improve using such techniques. Finally, field based retrieval will be improved. The databases built for Skeleton processing already support fields, but the `ngram` query processor needs modification to maintain legal syntax in `ngram` binding operations when field based operators are present in the query.

4 Skeleton's Web-Based Interface

A Web-based interface was developed to demonstrate Skeleton's document image retrieval capabilities. A key component of the interface is a Java applet which supports the highlighting of query terms within GIF image representations.

An electronic form which permits the selection of one or more document collections and the submission of a search query is illustrated in Figure 4. (Document collections may be distributed across multiple machines running different operating systems.) After a plain text search query has been entered and the "Eval" button is pressed, a structured query is formulated and submitted to each each selected database.

Retrieval results from the selected databases are merged and presented as a ranked list of titles, along with the structured query which was formulated from the plain text input. Figure 5 contains an example


Netscape - [INQUERY/emex - Begin]
Location: http://128.119.40.200/cgi-bin/idua/web_inquery/emex

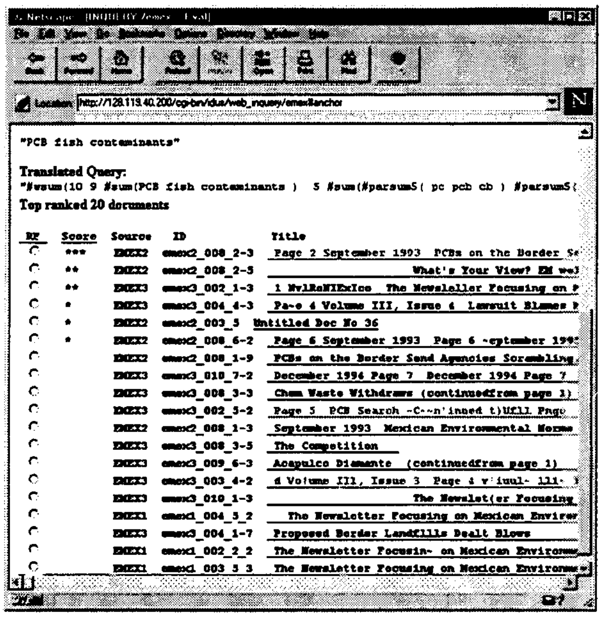
[✓ EMEX1]
[✓ EMEX2]
[✓ EMEX3]
☐ EMEX4
☐ All

Type a plain text or structured query:

PCB fish contaminants

Begin Eval Max documents 20

Figure 4: Skeleton Search Query Form



http://128.119.40.200/cgi-bin/idus/web_inquery/emex#anchor

"PCB fish contaminants"

Translated Query:

```
"#wsum(10 9 #sum(PCB fish contaminants )  5 #sum(#parsum5( pc pcb cb ) #parsum5(
```

Top ranked 20 documents

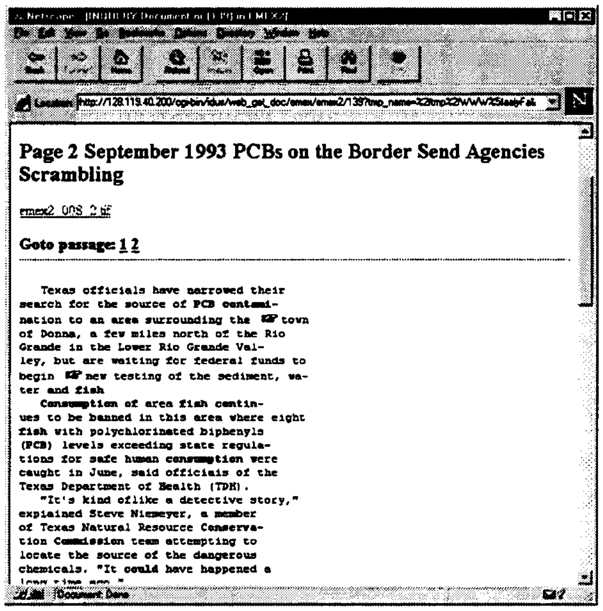
```
RF  Score  Source  ID            Title
◌   ***    EMEX2   emex2_008_2-3  Page 2 September 1993  PCBs on the Border Se
◌   **     EMEX2   emex2_008_2-5                       What's Your View? EM we
◌   **     EMEX3   emex3_002_1-3  1 NvlRoNIExIco  The Newslaller Focusing on M
◌   *      EMEX3   emex3_004_4-3  Pa-e 4 Volume III, Issue 4  Lawsuit Blames M
◌   *      EMEX2   emex2_003_5  Untitled Doc No 36
◌   *      EMEX2   emex2_008_6-2  Page 6 September 1993  Page 6 -eptember 199
◌          EMEX2   emex2_008_1-9  PCBs on the Border Send Agencies Scrambling.
◌          EMEX3   emex3_010_7-2  December 1994 Page 7  December 1994 Page 7
◌          EMEX3   emex3_008_3-3  Chem Waste Withdraws (continuedfrom page 1)
◌          EMEX3   emex3_002_5-2  Page 5  PCB Search -C--n'inued t)Ufll Page
◌          EMEX2   emex2_008_1-3  September 1993  Mexican Environmental Norms
◌          EMEX3   emex3_008_3-5  The Competition
◌          EMEX3   emex3_009_6-3  Acapulco Diamante  (continuedfrom page 1)
◌          EMEX3   emex3_003_4-2  4 Vo!ume III, Issue 3  Page 4 v'iuul- 111-
◌          EMEX3   emex3_010_1-3                         The Newslet(er Focusing
◌          EMEX1   emex1_004_5_2    The Newsletter Focusing on Mexican Enviror
◌          EMEX3   emex3_004_1-7  Proposed Border Landfills Dealt Blows
◌          EMEX1   emex1_002_2_2  The Newsletter Focusin- on Mexican Environme
◌          EMEX1   emex1_003_5_3  The Newsletter Focusing on Mexican Environme
```

Figure 5: Skeleton Retrieval Results Form

display of retrieved documents.

Titles of retrieved documents are linked to the OCR text output of their corresponding articles. Word tokens in the OCR text which correspond to word tokens or `ngrams` in the structured query are highlighted to faciliate understanding why the article was retrieved. Figure 7 contains an example display of the OCR text output of a retrieved article.

In a Web page display of the OCR text for a given article, a link is provided to a GIF image of the document page which contains the article. The GIF images used in the Web-based interface have been generated from the TIFF images used during image analysis. GIF images were needed in the current



http://128.119.40.200/cgi-bin/idus/web_get_doc/emex/emex2/139?tmp_name=%2Ftmp%2FWWW%25aabF&

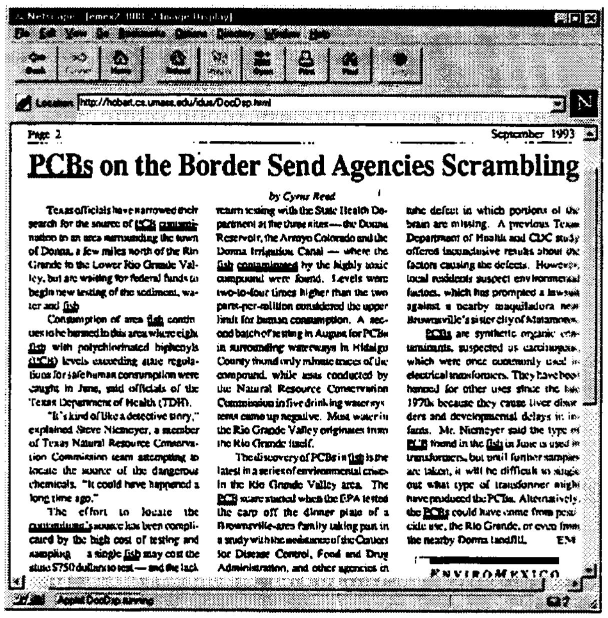
Page 2 September 1993 PCBs on the Border Send Agencies Scrambling

emex2_008_2.tif

Goto passage: 1 2

```
   Texas officials have narrowed their
search for the source of PCB contami-
nation to an area surrounding the  town
of Donna, a few miles north of the Rio
Grande in the Lower Rio Grande Val-
ley, but are waiting for federal funds to
begin  new testing of the sediment, wa-
ter and fish
   Consumption of area fish contin-
ues to be banned in this area where eight
fish with polychlorinated biphenyls
(PCB) levels exceeding state regula-
tions for safe human consumption were
caught in June, said officiais of the
Texas Department of Health (TDH).
   "It's kind oflike a detective story,"
expiained Steve Niemeyer, a member
of Texas Natural Resource Conserva-
tion Commission team attempting to
locate the source of the dangerous
chemicals. "It could have happened a
```

Figure 7: A display of OCR text for an article



http://hobart.cs.umass.edu/idus/DocDsp.html

Page 2 September 1993

PCBs on the Border Send Agencies Scrambling

by Cyrus Reed

Texas officials have narrowed their search for the source of PCB contamination to an area surrounding the town of Donna, a few miles north of the Rio Grande in the Lower Rio Grande Valley, but are waiting for federal funds to begin new testing of the sediment, water and fish.

Consumption of area fish continues to be banned in this area where eight fish with polychlorinated biphenyls (PCB) levels exceeding state regulations for safe human consumption were caught in June, said officials of the Texas Department of Health (TDH).

"It's kind of like a detective story," explained Steve Niemeyer, a member of Texas Natural Resource Conservation Commission team attempting to locate the source of the dangerous chemicals. "It could have happened a long time ago."

The effort to locate the contaminant's source has been complicated by the high cost of testing and sampling — a single fish may cost the state $750 dollars to test — and the lack

team testing with the State Health Department at the three sites — the Donna Reservoir, the Arroyo Colorado and the Donna Irrigation Canal — where the fish contaminated by the highly toxic compound were found. Levels were two-to-four times higher than the two parts-per-million considered the upper limit for human consumption. A second batch of testing in August for PCBs in surrounding waterways in Hidalgo County found only minute traces of the compound, while tests conducted by the Natural Resource Conservation Commission in five drinking water systems came up negative. Most water in the Rio Grande Valley originates from the Rio Grande itself.

The discovery of PCBs in fish is the latest in a series of environmental crises in the Rio Grande Valley area. The PCB scare started when the EPA tested the carp off the dinner plate of a Brownsville-area family taking part in a study with the assistance of the Centers for Disease Control, Food and Drug Administration, and other agencies in

tube defect in which portions of the brain are missing. A previous Texas Department of Health and CDC study offered inconclusive results about the factors causing the defects. However, local residents suspect environmental factors, which has prompted a lawsuit against a nearby maquiladora near Brownsville's sister city of Matamoros.

PCBs are synthetic organic contaminants, suspected as carcinogens, which were once commonly used in electrical transformers. They have been banned for other uses since the late 1970s because they cause liver disorders and developmental delays in infants. Mr. Niemeyer said the type of PCB found in the fish in June is used in transformers, but until further samples are taken, it will be difficult to single out what type of transformer might have produced the PCBs. Alternatively, the PCBs could have come from pesticide use, the Rio Grande, or even from the nearby Donna landfill. EM

ENVIROMEXICO

Figure 6: A display of a document page image containing an article

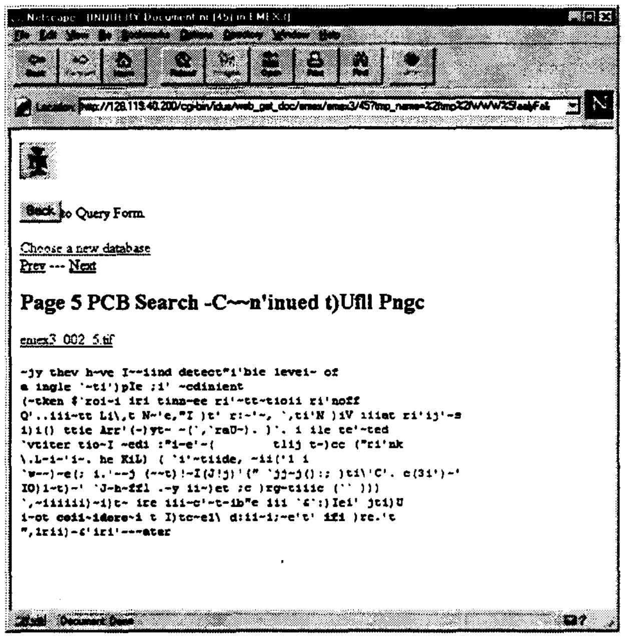
implementation of the interface because Java does not currently support TIFF image display.

Figure 6 contains an example display of a document image using the Java applet developed for the interface. Terms in the structured query are highlighted with red underlining to help focus attention on relevant parts of the page. Term highlighting is expanded somewhat in the image display with a rudimentary partial matching function which recognizes near matches of query terms through the use of a stemming routine. It is possible that a document image will be displayed without any terms highlighted if the query terms and their near matches are not actually found in the image. This may occur if the selected document was retrieved solely on the basis of `ngrams`.

Figure 9 contains a display of the text output for an article in which significant OCR errors occur. Despite the high error rate, a user may access the document image containing the article, illustrated in Figure 8, and determine that it is indeed relevant to the search query.

In future work on Skeleton's interface, evaluations of interface design will be made to determine how best to coordinate the display of OCR text output and document images. To further facilitate the search process, navigation buttons will be introduced into the image displays to make it possible to search through other page images of the same document.

If possible, the need to maintain GIF images of documents will be eliminated, but it may nevertheless be desirable to maintain higher resolution, color images of documents for display to users. The attractiveness of this design feature is particularly compelling when viewing relatively poor quality images such as the one displayed in Figure 9.



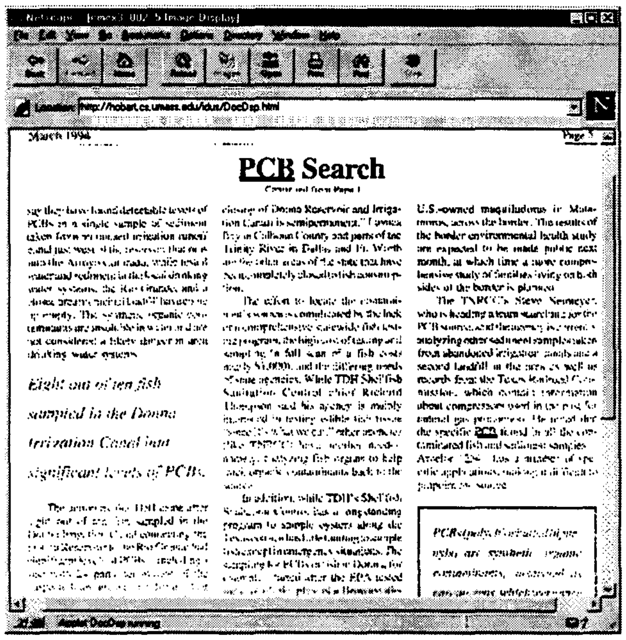
Location: http://128.119.40.200/cgi-bin/idus/web_get_doc/emex/emex3/45?tmp_name=%2tmp%2fWWW%25aajFak
Back to Query Form
Choose a new database
Prev --- Next
Document Done

Page 5 PCB Search -C~~n'inued t)Ufll Pngc

emex3_002_5.tif

```
~jy thev h~ve I~~iind detect"i'bie levei~ of
a ingle `~ti')pIe ;i' ~cdinient
(~tken $'roi~i iri tinn~ee ri'~tt~tioii ri'noff
Q'..iii~tt Li\,t N~'e,"I )t' r:~'~, `,ti'N )iV iiiat ri'ij'~s
i)i() ttie Arr'(~)yt~ ~(`,`raU~). )`. i iie te'~ted
`vtiter tio~I ~edi :"i~e'~(        tlij t~)cc ("ri'nk
\.L~i~'i~. he KiL) ( `i'~tiide, ~ii('1 i
`w~~)~e(; i.'~~j (~~t)!~I(J!j)'(" `jj~j():; )ti\'C'. e(3i')~'
IO)i~t)~' `J~h~ffl .~y ii~)et ;c )rg~tiiic (`` )))
`,~iiiiii)~i)t~ ire iii~c'~t~ib"e iii `£':)Iei' jti)U
i~ot ceii~idere~i t I)tc~el\ d:ii~i;~e't' ifi )re.'t
",lrii)~£'iri'~~~ater
```

Figure 9: A display of an article's text containing significant OCR errors



PCB Search

[illegible]

Eight out of ten fish sampled in the Donna Irrigation Canal had significant levels of PCBs.

[illegible]

Figure 8: A display of a document page image containing an article with significant OCR errors

7 Conclusions

The Skeleton architecture is based upon existing image analysis and document retrieval technologies. Incremental improvements in the technologies are being made in an effort to retain the scalability of the existing technologies while extending them to meet new challenges.

Acknowledgements

Work on Skeleton is supported by the U.S. Department of Defense.

References

[1] S.L. Taylor, M. Lipshutz, D. A. Dahl and C. Weir. An Intelligent Document Understanding System. In *Second International Conference on Document Analysis and Recognition*, pp 107-220, Tsukuba City, Japan, October 1993.

[2] J.P. Callan, W.B. Croft and S.M. Harding. The INQUERY Retrieval System. Proceedings of the 3rd International Conference on Database and Expert Systems Applications, pp 78-83, 1992.

[3] S. Rice, J. Kanai and T. Nartker. An Evaluation of Information Retrieval Accuracy. In UNLV Information Science Research Institute Annual Report, pp 9-20, 1993.

[4] Scan*WorX* API Programmer's Guide. Xerox Imaging Systems. Part Number 00-07570-00. February 1993.

[5] XDOC Data Format: Technical Specification. Xerox Imaging Systems. Part Number 00-7571-00. February 1993.

[6] S. Tsujimoto and H. Asada, Document Image Analysis. In 8^{th} *Int. Conf. On Pattern Recognition.* 1986, pp. 434-438.

[7] W.B. Croft, S.M. Harding, K. Taghva and J. Borsack. An evaluation of Information Retrieval Accuracy with Simulated OCR Output. University of Massachusetts Computer Science Technical Report 93-70:1-12, 1993.

Multimedia Indexing And Retrieval Research at the Center for Intelligent Information Retrieval

R. Manmatha *
Multimedia Indexing and Retrieval Group
Center for Intelligent Information Retrieval
Computer Science Department
University of Massachusetts, Amherst, MA 01003
manmatha@cs.umass.edu

Abstract

The digital libraries of the future will include not only (ASCII) text information but scanned paper documents as well as still photographs and videos. There is, therefore, a need to index and retrieve information from such multi-media collections. The Center for Intelligent Information Retrieval (CIIR) has a number of projects to index and retrieve multi-media information. These include:

1. *The extraction of text from images which may be used both for finding text zones against general backgrounds as well as for indexing and retrieving image information.*
2. *Indexing hand-written and poorly printed documents using image matching techniques (word spotting).*
3. *Indexing images using their content.*

1 Introduction

The digital libraries of the future will include not only (ASCII) text information but scanned paper documents as well as still photographs and videos. There is, therefore, a need to index and retrieve information from such multi-media collections. The Center for Intelligent Information Retrieval (CIIR) has a number of projects to index and retrieve multi-media information. These include:

1. Finding Text in Images: The conversion of scanned documents into ASCII so that they can be indexed using INQUERY (CIIR's text retrieval engine). Current Optical Character Recognition Technology (OCR) can convert scanned text to ASCII but is limited to good clean machine printed fonts against clean backgrounds. Handwritten text, text printed against shaded or textured backgrounds and text embedded in images cannot be recognized well (if it can be recognized at all) with existing OCR technology. Many financial documents, for example, print text against shaded backgrounds to prevent copying.

   The Center has developed techniques to detect text in images. The detected text is then cleaned up and binarized and run through a commercial OCR. Such techniques can be applied to zoning text found against general backgrounds as well as for indexing and retrieving images using the associated text.

2. Word Spotting: The indexing of hand-written and poorly printed documents using image matching techniques. Libraries hold vast collections of original handwritten manuscripts, many of which have never been published. Word Spotting can be used to create indices for such handwritten manuscript archives.

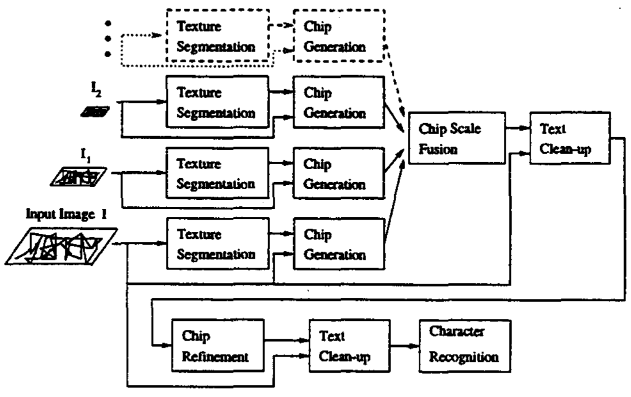
3. Image Retrieval: Indexing images using their content. The Center has also developed techniques to index and retrieve images by color and appearance.

2 Finding Text in Images

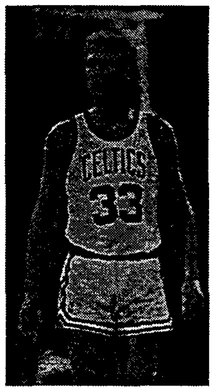
Most of the information available today is either on paper or in the form of still photographs and videos. To build digital libraries, this large volume of information needs to be digitized into images and the text converted to ASCII for storage, retrieval, and easy manipulation. For example, video sequences of events such as a basketball game can be annotated and indexed by extracting a player's number, name and the team name that appear on the player's uniform (Figure 1(b, c)). This maybe combined with methods for image indexing and retrieval based on image content (see section 3).

Current OCR technology [1, 20] is largely restricted to finding text printed against clean backgrounds, since

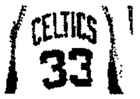
This material is based on work supported in part by the National Science Foundation, Library of Congress and Department of Commerce under cooperative agreement number EEC-9209623, in part by the United States Patent and Trademarks Office and the Defense Advanced Research Projects Agency/ITO under ARPA order number D468, issued by ESC/AXS contract number F19628-95-C-0235, in part by NSF IRI-9619117 and in part by NSF Multimedia CDA-9502639. Any opinions, findings and conclusions or recommendations expressed in this material are the author(s) and do not necessarily reflect those of the sponsors.



(a)



(b)



(c)

Figure 1: The system, example input image, and extracted text. (a) The top level components of the text detection and extraction system. The pyramid of the input image is shown as I, I_1, I_2 ...; (b) An example input image; (c) Output of the system before being fed to the Character Recognition module.

in these cases it is easy to binarize the input images to extract text (text binarization) before character recognition begins. It cannot handle text printed against shaded or textured backgrounds, nor text embedded in pictures. More sophisticated text reading systems usually employ page segmentation schemes to identify text regions. Then an OCR module is applied only to the text regions to improve its performance. Some of these schemes [32, 33, 21, 23] are top-down approaches, some are bottom-up methods [7, 22], and others are based on texture segmentation techniques in computer vision [8]. However, the top-down and bottom-up approaches usually require the input image to be binary and have a Manhattan layout. Although the approach in [8] can in principle be applied to greyscale images, it was only used on binary document images, and in addition, the text binarization problem was not addressed. In summary, few working systems have been reported that can read text from document pages with both structured and non-structured layouts. A brief overview of a system developed at CIIR for constructing a complete automatic text reading system is presented here (for more details see [34, 35]).

2.1 System Overview

The system takes advantage of the following distinctive characteristics of text which make it stand out from other image information: (1) Text possesses a distinctive frequency and orientation attributes; (2) Text shows spatial cohesion — characters of the same text string are of similar heights, orientation and spacing.

The first characteristic suggests that text may be treated as a distinctive texture, and thus be segmented out using texture segmentation techniques. Thus, the first phase of our system is Texture Segmentation as shown in Figure 1(a). In the Chip Generation phase, strokes are extracted from the segmented text regions. Using reasonable heuristics on text strings based on the second characteristic, the extracted strokes are then processed to form tight rectangular bounding boxes around the corresponding text strings. To detect text over a wide range of font sizes, the above steps are applied to a pyramid of images generated from the input image, and then the boxes formed at each resolution level of the pyramid are fused at the original resolution. A Text Clean-up module which removes the background and binarizes the detected text is applied to extract the text from the regions enclosed by the bounding boxes. Finally, text bounding boxes are refined (re-generated) by using the extracted items as strokes. These new boxes usually bound text strings better. The Text Clean-up process is then carried out on the regions bounded by these new boxes to extract cleaner text, which can then be passed through a commercial OCR engine for recognition if the text is of an OCR-recognizable font. The phases of the system are discussed in the following sections.

2.2 The Texture Segmentation Module

A standard approach to texture segmentation is to first filter the image using a bank of linear filters such as Gaussian derivatives [11] or Gabor functions, followed by some non-linear transformation such as a hyperbolic function $tanh(\alpha t)$. Then features are computed to form a feature vector for each pixel from the filtered images. These feature vectors are then classified to segment the textures into different classes (for more details see [34, 35]).

Figure 2(a) shows a portion of an original input image with a variety of textual information to be extracted. There is text on a clean dark background, text printed on Stouffer boxes, Stouffer's trademarks (in script), and a picture of the food. Figure 2(b) shows the final segmented text regions.

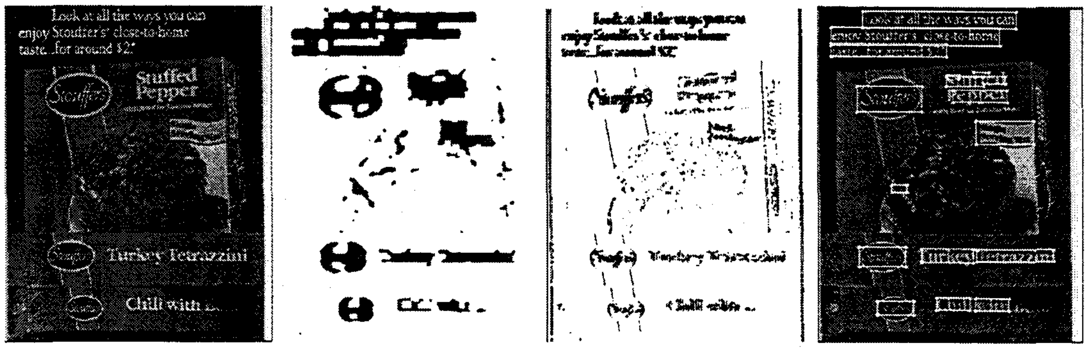

(a) (b) (c) (d)

Figure 2: Results of Texture Segmentation and Chip Generation. (a) Portion of an input image; (b) The final segmented text regions; (c) Extracted strokes; (d) Text chips mapped on the input image.

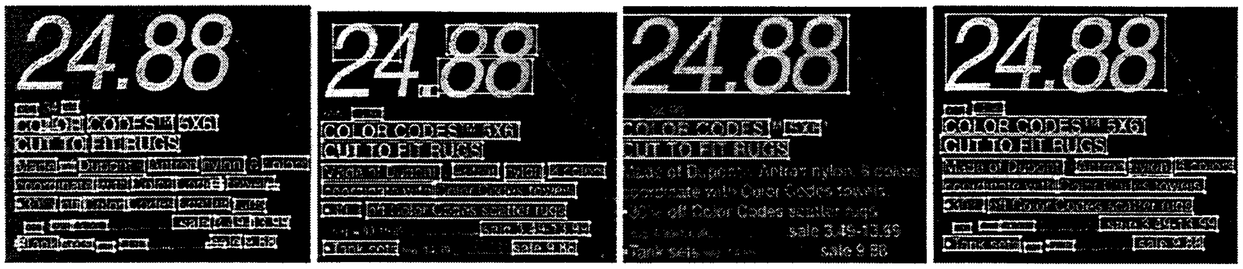

(a) (b) (c) (d)

Figure 3: The scale problem and its solution. (a) Chips generated for the input image at full resolution; (b) half resolution; (c) $\frac{1}{4}$ resolution; (d) Chips generated at all three levels mapped onto the input image. Scale-redundant chips are removed.

2.3 The Chip Generation Phase

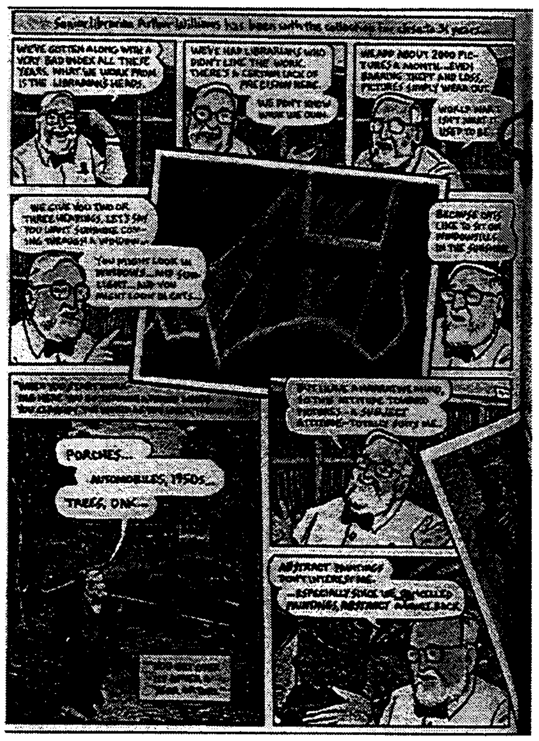
In practice, text may occur in images with complex backgrounds and texture patterns, such as foliage, windows, grass etc. Thus, some non-text patterns may pass the filters and initially be misclassified as text (Figure 2(b)). Furthermore, segmentation accuracy at texture boundaries is a well-known and difficult problem in texture segmentation. Consequently, it is often the case that text regions are connected to other regions which do not correspond to text, or one text string might be connected to another text string of a different size or intensity. This might cause problems for later processing. For example, if two text strings with significantly different intensity levels are joined into one region, one intensity threshold might not separate both text strings from the background.

Therefore, heuristics need to be employed to refine the segmentation result. Since the segmentation process usually finds text regions while excluding most of those that are non-text, these regions can be used to direct further processing (**focus of attention**). Furthermore, since text is intended to be readable, there is usually a significant contrast between it and the background. Thus contrast can be utilized finding text. Also, it is usually the case that characters in the same word/phrase/sentence are of the same font and have similar heights and inter-character spaces. Finally, it is obvious that characters in a horizontal text string are horizontally aligned. Therefore, all the heuristics above are incorporated in the Chip Generation phase in a bottom-up fashion: significant edges form strokes (Figure 2(c)); strokes from the segmented regions are aggregated to form chips corresponding to text strings. The rectangular bounding boxes of the chips are used to indicate where the hypothesized (detected) text strings are (Figure 2(d)). These steps are described in detail in [34, 35].

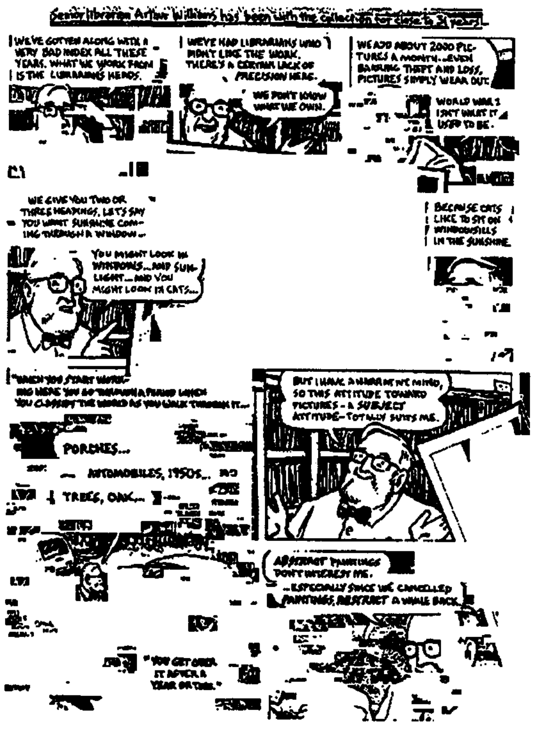
2.4 A Solution to the Scale Problem

The three frequency channels used in the segmentation process work well to cover text over a certain range of font sizes. Text from larger font sizes is either missed or fragmented. This is called the **scale problem**. Intuitively, the larger the font size of the text, the lower the frequency it possesses. Thus, when the text font size gets too large, its frequency falls outside the three channels selected in section 2.2.

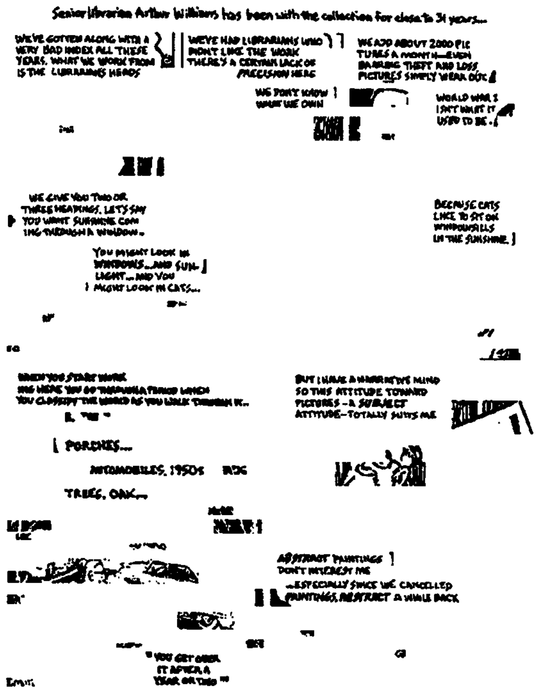
A pyramid approach (Figure 1(a)) is used to solve the scale problem: a pyramid of the input image is formed and each image in the pyramid is processed using the standard channels ($\sigma = 1, \sqrt{2}, 2$) as described in the previous sections. At the bottom of the pyramid is the original image; the image at each level (other than the bottom)


Senior librarian Arthur Williams has been with the collection for close to 31 years...
WE'VE GOTTEN ALONG WITH A VERY BAD INDEX ALL THESE YEARS. WHAT WE WORK FROM IS THE LIBRARIANS' HEADS.
WE'VE HAD LIBRARIANS WHO DIDN'T LIKE THE WORK. THERE'S A CERTAIN LACK OF PRECISION HERE.
WE DON'T KNOW WHAT WE OWN.
WE ADD ABOUT 2000 PICTURES A MONTH...EVEN BARRING THEFT AND LOSS, PICTURES SIMPLY WEAR OUT.
WORLD WAR 1 ISN'T WHAT IT USED TO BE.
WE GIVE YOU TWO OR THREE HEADINGS. LET'S SAY YOU WANT SUNSHINE COMING THROUGH A WINDOW...
YOU MIGHT LOOK IN WINDOWS...AND SUNLIGHT...AND YOU MIGHT LOOK IN CATS...
BECAUSE CATS LIKE TO SIT ON WINDOWSILLS IN THE SUNSHINE.
PORCHES...
AUTOMOBILES, 1950s...
TREES, OAK...
ABSTRACT PAINTINGS DON'T INTEREST ME.
...ESPECIALLY SINCE WE CANCELLED PAINTINGS, ABSTRACT A WHILE BACK.

(a)


Senior librarian Arthur Williams has been with the collection for close to 31 years...
WE'VE GOTTEN ALONG WITH A VERY BAD INDEX ALL THESE YEARS. WHAT WE WORK FROM IS THE LIBRARIANS' HEADS.
WE'VE HAD LIBRARIANS WHO DIDN'T LIKE THE WORK. THERE'S A CERTAIN LACK OF PRECISION HERE.
WE ADD ABOUT 2000 PICTURES A MONTH...EVEN BARRING THEFT AND LOSS, PICTURES SIMPLY WEAR OUT.
WE DON'T KNOW WHAT WE OWN.
WORLD WAR 1 ISN'T WHAT IT USED TO BE.
WE GIVE YOU TWO OR THREE HEADINGS. LET'S SAY YOU WANT SUNSHINE COMING THROUGH A WINDOW...
BECAUSE CATS LIKE TO SIT ON WINDOWSILLS IN THE SUNSHINE.
YOU MIGHT LOOK IN WINDOWS...AND SUNLIGHT...AND YOU MIGHT LOOK IN CATS...
BUT I HAVE A WARNING IN MIND, SO THIS ATTITUDE TOWARD PICTURES - A SUBJECT ATTITUDE - TOTALLY SUITS ME.
PORCHES...
AUTOMOBILES, 1950s...
TREES, OAK...
ABSTRACT PAINTINGS DON'T INTEREST ME.
...ESPECIALLY SINCE WE CANCELLED PAINTINGS, ABSTRACT A WHILE BACK.
"YOU GET OVER IT AFTER A YEAR OR TWO."

(b)


Senior librarian Arthur Williams has been with the collection for close to 31 years...
WE'VE GOTTEN ALONG WITH A VERY BAD INDEX ALL THESE YEARS. WHAT WE WORK FROM IS THE LIBRARIANS' HEADS
WE'VE HAD LIBRARIANS WHO DIDN'T LIKE THE WORK. THERE'S A CERTAIN LACK OF PRECISION HERE
WE ADD ABOUT 2000 PICTURES A MONTH—EVEN BARRING THEFT AND LOSS PICTURES SIMPLY WEAR OUT
WE DON'T KNOW WHAT WE OWN
WORLD WAR 1 ISN'T WHAT IT USED TO BE.
WE GIVE YOU TWO OR THREE HEADINGS. LET'S SAY YOU WANT SUNSHINE COMING THROUGH A WINDOW...
BECAUSE CATS LIKE TO SIT ON WINDOWSILLS IN THE SUNSHINE.
YOU MIGHT LOOK IN WINDOWS...AND SUNLIGHT...AND YOU MIGHT LOOK IN CATS...
BUT I HAVE A WARNING IN MIND SO THIS ATTITUDE TOWARD PICTURES - A SUBJECT ATTITUDE - TOTALLY SUITS ME
PORCHES...
AUTOMOBILES, 1950s
TREES, OAK...
ABSTRACT PAINTINGS DON'T INTEREST ME
...ESPECIALLY SINCE WE CANCELLED PAINTINGS, ABSTRACT A WHILE BACK
"YOU GET OVER IT AFTER A YEAR OR TWO"

(c)

Figure 4: Binarization results before and after the Chip Refinement step. (a) Input image; (b) binarization result before refinement; (c) after refinement.

has half of the resolution as that of the image one level below. Text of smaller font sizes can be detected using the images lower in the pyramid (Figure 3(a)), while text of large font sizes is found using images higher in the pyramid (Figure 3(c). The bounding boxes of detected text regions at each level are mapped back to the original input image and the redundant boxes are then removed as shown in Figure 3(d). Details are presented in [34, 35].

2.5 Text on Complex Backgrounds

The previous sections describe a system which detects text in images and puts boxes around detected text strings in the input image. Since text may be printed against complex image backgrounds, which current OCR systems cannot handle well, it is desirable to have the backgrounds removed first. In addition, OCR systems require that the text must be binarized before actual recognition starts. In this system, the background removal and text binarization is done by applying an algorithm to the text boxes individually instead of trying to binarize the input image as a whole. This allows the process to adapt to the individual context of each text string. The details of the algorithm are in [34, 35].

2.6 The Text Refinement

Sometimes non-text items are identified as text as well. In addition, the bounding boxes of the chips sometimes do not tightly surround the text strings. The consequence of these problems is that non-text items may occur in the binarized image, produced by mapping the extracted items onto the original page. An example is shown in Figure 4(a,b). These non-text items are not desirable.

However, by treating the extracted items as strokes, the Chip Refinement module which is essentially similar to the chip Generation module but with stronger constraints, can be applied here to eliminate the non-text items and hence form tighter text bounding boxes. This can be achieved because (1) the clean-up procedure is able to extract most characters without attaching to nearby characters and non-text items (Figure 4(b)), and (2) most of the strokes at this stage are composed of complete or almost complete characters, as opposed to the vertical connected edges of the characters in the initial processing. Thus, it can be expected that the correct text strokes comply more consistently with the heuristics used in the early Chip Generation phase. The significant improvement is clearly shown in 4c.

2.7 Experiments

The system has been tested over 48 images from a wide variety of sources: digitized video frames, photographs, newspapers, advertisements in magazines or sales flyers, and personal checks. Some of the images have regular page layouts, others do not. It should be pointed out that all the system parameters remain the same throughout the entire set of test images, showing the robustness of the system.

Characters and words (as perceived by one of the authors) were counted in each image as ground truth. The total numbers over the whole test set are shown in the "Total Perceived" column in Table 1. The detected characters and words are those which are completely enclosed by the boxes produced after the Chip Scale Fusion step. The total numbers of detected characters and words over the entire test set are shown in the "Total Detected" column. Characters and words clearly readable by a person after the Chip Refinement and Text Clean-up steps (final extracted text) are also counted for each image, with the total numbers shown in the "Total Clean-up" column. The column "Total OCRable" shows the total numbers of cleaned-up characters and words that appear to be of OCR recognizable fonts in 35 of the binarized images. Note that only the text which is horizontally aligned is counted (skew angle of the text string is less than roughly 30 degrees)[1]. The "Total OCRed" column shows the numbers of characters and words from the "Total OCRable" sets correctly recognized by Caere's commercial WordScan OCR engine.

Figure 5(a) is a portion of an original input image which has no structured layout. The final binarization result is shown in (b) and the corresponding OCR output is shown in (c). Notice that most of the text is detected, and most of the text of machine-printed fonts are correctly recognized by the OCR engine. It should be pointed out that the cleaned-up output looks fine to a person in the places where the OCR errors occurred.

3 Word Spotting: Indexing Handwritten Archival Manuscripts

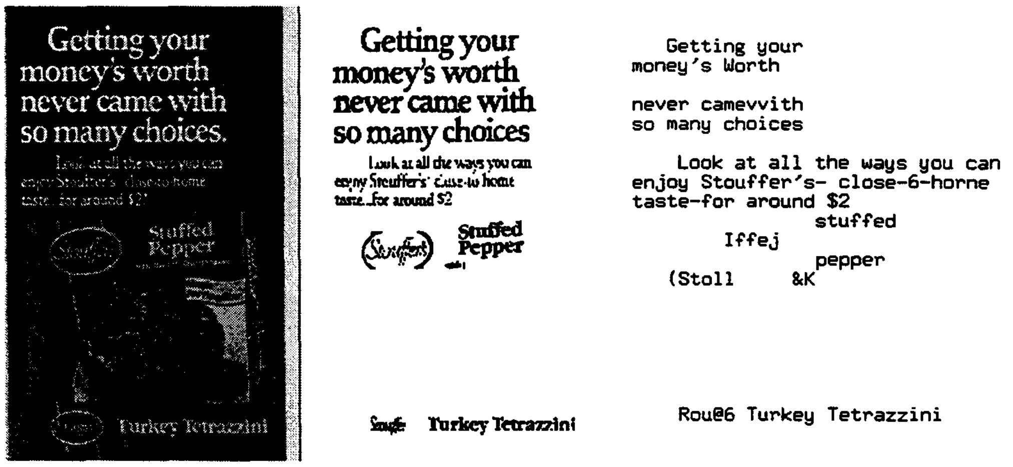
There are many historical manuscripts written in a single hand which it would be useful to index. Examples include the W. B. DuBois collection at the University of Massachusetts, Margaret Sanger's collected works at Smith College and the early Presidential libraries at the Library of Congress. These manuscripts are largely written in a single hand. Such manuscripts are valuable resources for scholars as well as others who wish to consult the original manuscripts and considerable effort has gone into manually producing indices for them. For example, a substantial collection of Margaret Sanger's work has been recently put on microfilm (see http://MEP.cla.sc.edu/Sanger/SangBase.HTM) with an item by item index. These indices were created manually. The indexing scheme described here will help in the automatic creation and production of indices and concordances for such archives.

One solution is to use Optical Character Recognition (OCR) to convert scanned paper documents into ASCII.

[1]Here, the focus is on finding horizontal, linear text strings only. The issue of finding text strings of any orientation will be addressed in future work.

Table 1: Summary of the system's performance. 48 images were used for detection and clean-up. Out of these, 35 binarized images were used for the OCR process.

	Total Perceived	Total Detected	Total Clean-up	Total OCRable	Total OCRed
Char	21820	20788 (95%)	91%	14703	12428 (84%)
Word	4406	4139 (93%)	86%	2981	2314 (77%)



```
   Getting your
money's Worth

never camevvith
so many choices

     Look at all the ways you can
enjoy Stouffer's- close-6-horne
taste-for around $2
                       stuffed
          Iffej
                     pepper
   (Stoll        &K



   Rou@6 Turkey Tetrazzini
```

(a) (b) (c)

Figure 5: Example 1. (a) Original image (ads11); (b) Extracted text; (c) The OCR result using Caere's WordScan Plus 4.0 on b.

Existing OCR technology works well with standard machine printed fonts against clean backgrounds. It works poorly if the originals are of poor quality or if the text is handwritten. Since Optical Character Recognition (OCR) does not work well on handwriting, an alternative scheme based on matching the images of the words was proposed by us in [18, 17, 15] for indexing such texts. Here a brief summary of the work is presented.

Since the document is written by a single person, the assumption is that the variation in the word images will be small. The proposed solution will first segment the page into words and then match the actual word images against each other to create equivalence classes. Each equivalence class will consist of multiple instances of the same word. Each word will have a link to the page it came from. The number of words in each equivalence class will be tabulated. Those classes with the largest numbers of words will probably be stopwords, i.e. conjunctions such as "and" or articles such as "the". Classes containing stopwords are eliminated (since they are not very useful for indexing). A list is made of the remaining classes. This list is ordered according to the number of words contained in each of the classes. The user provides ASCII equivalents for a representative word in each of the top m (say m = 2000) classes. The words in these classes can now be indexed. This technique will be called "word spotting" as it is analogous to "word spotting" in speech processing [9].

The proposed solution completely avoids machine recognition of handwritten words as this is a difficult task [20]. Robustness is achieved compared to OCR systems for two reasons:

1. Matching is based on entire words. This is in contrast to conventional OCR systems which essentially recognize characters rather than words.

2. Recognition is avoided. Instead a human is placed in the loop when ASCII equivalents of the words must be provided.

Some of the matching aspects of the problem are discussed here (for a discussion of page segmentation into words, see [18]). The matching phase of the problem is expected to be the most difficult part of the problem. This is because unlike machine fonts, there is some variation in even a single person's handwriting. This variation is difficult to model. Figure (6) shows two examples of the word "Lloyd" written by the same person. The last image is produced by XOR'ing these two images. The white areas in the XOR image indicate where the two versions of "Lloyd" differ. This result is not unusual. In fact, the differences are sometimes even larger.

The performance of two different matching techniques is discussed here. The first, based on Euclidean distance mapping [2], assumes that the deformation between words can be modelled by a translation (shift). The second, based on an algorithm by Scott and Longuet

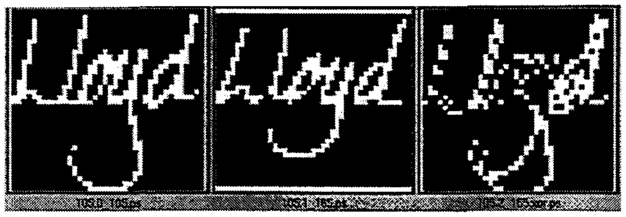


Figure 6: Two examples of the word "Lloyd" and the XOR image

Higgins [28] models the transformation between words using an affine transform.

3.1 Prior Work

The traditional approach to indexing documents involves first converting them to ASCII and then using a text based retrieval engine [30]. Scanned documents printed in standard machine fonts against clean backgrounds can be converted into ASCII using an OCR [1]. However, handwriting is much more difficult for OCRs to handle because of the wide variability present in handwriting (not only is there variability between writers, but a given person's writing also varies).

Image matching of words has been used to recognize words in documents which use machine fonts [5, 10]. Recognition rates are much higher than when the OCR is used directly [10]. Machine fonts are simpler to match than handwritten fonts since the variation is much smaller; multiple instances of a given word printed in the same font are identical except for noise. In handwriting, however, multiple instances of the same word on the same page by the same writer show variations. The first two pictures in Figure 6 are two identical words from the same document, written by the same writer. It may thus be necessary to account for these variations.

3.2 Outline of Algorithm

1. A scanned greylevel image of the document is obtained.
2. The image is first reduced by half by gaussian filtering and subsampling.
3. The reduced image is then binarized by thresholding the image.
4. The binary image is now segmented into words. this is done by a process of smoothing and thresholding (see [18]).
5. A given word image (i.e. the image of a word) is used as a template. and matched against all the other word images. This is repeated for every word in the document. The matching is done in two phases. First, the number of words to be matched is pruned using the areas and aspect ratios of the word images - the word to be matched cannot have an area or aspect ratio which is too different from the template. Next, the actual matching is done by using a matching algorithm. Two different matching algorithms are tried here. One of them only accounts for translation shifts, while the other accounts for affine matches. The matching divides the word images into equivalence classes - each class presumably containing other instances of the same word.
6. Indexing is done as follows. For each equivalence class, the number of elements in it is counted. The top n equivalence classes are then determined from this list. The equivalence classes with the highest number of words (elements) are likely to be stopwords (i.e. conjunctions like 'and' , articles like 'the', and prepositions like 'of') and are therefore eliminated from further consideration. Let us assume that of the top n, m are left after the stopwords have been eliminated. The user then displays one member of each of these m equivalence classes and assigns their ASCII interpretation. These m words can now be indexed anywhere they appear in the document.

We will now discuss the matching techniques in detail.

3.3 Determination of Equivalence Classes

The list of words to be matched is first pruned using the areas and aspect ratios of the word images. The pruned list of words is then matched using a matching algorithm.

3.4 Pruning

It is assumed that

$$\frac{1}{\alpha} \leq \frac{A_{word}}{A_{template}} \leq \alpha \tag{1}$$

where $A_{template}$ is the area of the template and A_{word} is the area of the word to be matched. Typical values of α used in the experiments range between 1.2 and 1.3. A similar filtering step is performed using aspect ratios (ie. the width/height ratio). It is assumed that

$$\frac{1}{\beta} \leq \frac{Aspect_{word}}{Aspect_{template}} \leq \beta. \tag{2}$$

The value of β used in the experiments range between 1.4 and 1.7. In both the above equations, the exact factors are not important but it should not be so large so that valid words are omitted, nor so small so that too many words are passed onto the matching phase. The pruning values may be automatically determined by running statistics on samples of the document [15].

3.5 Matching

The template is then matched against the image of each word in the pruned list. The matching function must satisfy two criteria:

1. It must produce a low match error for words which are similar to the template.
2. It must produce a high match error for words which are dissimilar.

Two matching algorithms have been tried. The first algorithm - Euclidean Distance Mapping (EDM) - assumes that no distortions have occured except for relative translation and is fast. This algorithm usually ranks the matched words in the correct order (i.e. valid words first, followed by invalid words) when the variations in words is not too large. Although, it returns the lowest errors for words which are similar to the template, it also returns low errors for words which are dissimilar to the template. The second algorithm [28],referred to as SLH here, assumes an affine transformation between the words. It thus compensates for some of the variations in the words. This algorithm not only ranks the words in the correct order for all examples tried so far, it also seems to be able to better discriminate between valid words and invalid words. As currently implemented the SLH algorithm is much slower than the EDM algorithm (we expect to be able to speed it up).

3.6 Using Euclidean Distance Mapping for Matching

This approach is similar to that used by [6] to match machine generated fonts. A brief description of the method follows (more details are available from [18]).

Consider two images to be matched. There are three steps in the matching:

1. First the images are roughly aligned. In the vertical direction, this is done by aligning the baselines of the two images. In the horizontal direction, the images are aligned by making their left hand sides coincide.

   The alignment is, therefore, expected to be accurate in the vertical direction and not as good in the horizontal direction. This is borne out in practice.
2. Next the XOR image is computed. This is done by XOR'ing corresponding pixels (see Figure 6).
3. An Euclidean distance mapping [2] is computed from the XOR image by assigning to each white pixel in the image, its minimum distance to a black pixel. Thus a white pixel inside a blob is assigned a larger distance than an isolated white pixel. An error measure E_{EDM} can now be computed by adding up the distance measures for each pixel.
4. Although the approximate translation has been computed using step 1, this may not be accurate and may need to be fine-tuned. Thus steps (2) and (3) are repeated while sampling the translation space in both x and y. A minimum error measure E_{EDMmin} is computed over all the translation samples.

3.7 SLH Algorithm for Matching

The EDM algorithm does not discriminate well between good and bad matches. In addition, it fails when there is significant distortion in the words. This happens with the writing of Erasmus Hudson (Figure 7). Thus a matching algorithm which models some of the variation is needed. A second matching algorithm (SLH), which models the distortion as an affine transformations, was therefore tried (note that it is expected that the real variation is probably much more complex). An affine transform is a linear transformation between coordinate systems. In two dimensions, it is described by

$$\mathbf{r}' = \mathbf{A}\mathbf{r} + \mathbf{t} \tag{3}$$

where $\mathbf{t}$ is a 2-D vector describing the translation, $\mathbf{A}$ is a 2 by 2 matrix which captures the deformation, $\mathbf{r}'$ and $\mathbf{r}$ are the coordinates of corresponding points in the two images between which the affine transformation must be recovered. An affine transform allows for the following deformations - scaling in both directions, shear in both directions and rotation.

The algorithm chosen here is one proposed by Scott and Longuet-Higgins [28] (see [16]). The algorithm recovers the correspondence between two sets of points I and J under an affine transform.

The sets I and J are created as follows. Every white pixel in the first image is a member of the set I. Similarly, every white pixel in the second image is a member of set J. First, the centroids of the point sets are computed and the origins of the coordinate systems is set at the centroid. The SLH algorithm is then used to compute the correspondence between the point sets.

Given the (above) correspondence between point sets I and J, the affine transform $\mathbf{A}, \mathbf{t}$ can be determined by minimizing the following least mean squares criterion:

$$E_{SLH} = \sum_{l} (I_l - \mathbf{A}J_l - \mathbf{t})^2 \tag{4}$$

where I_l, J_l are the (x,y) coordinates of point I_l and J_l respectively.

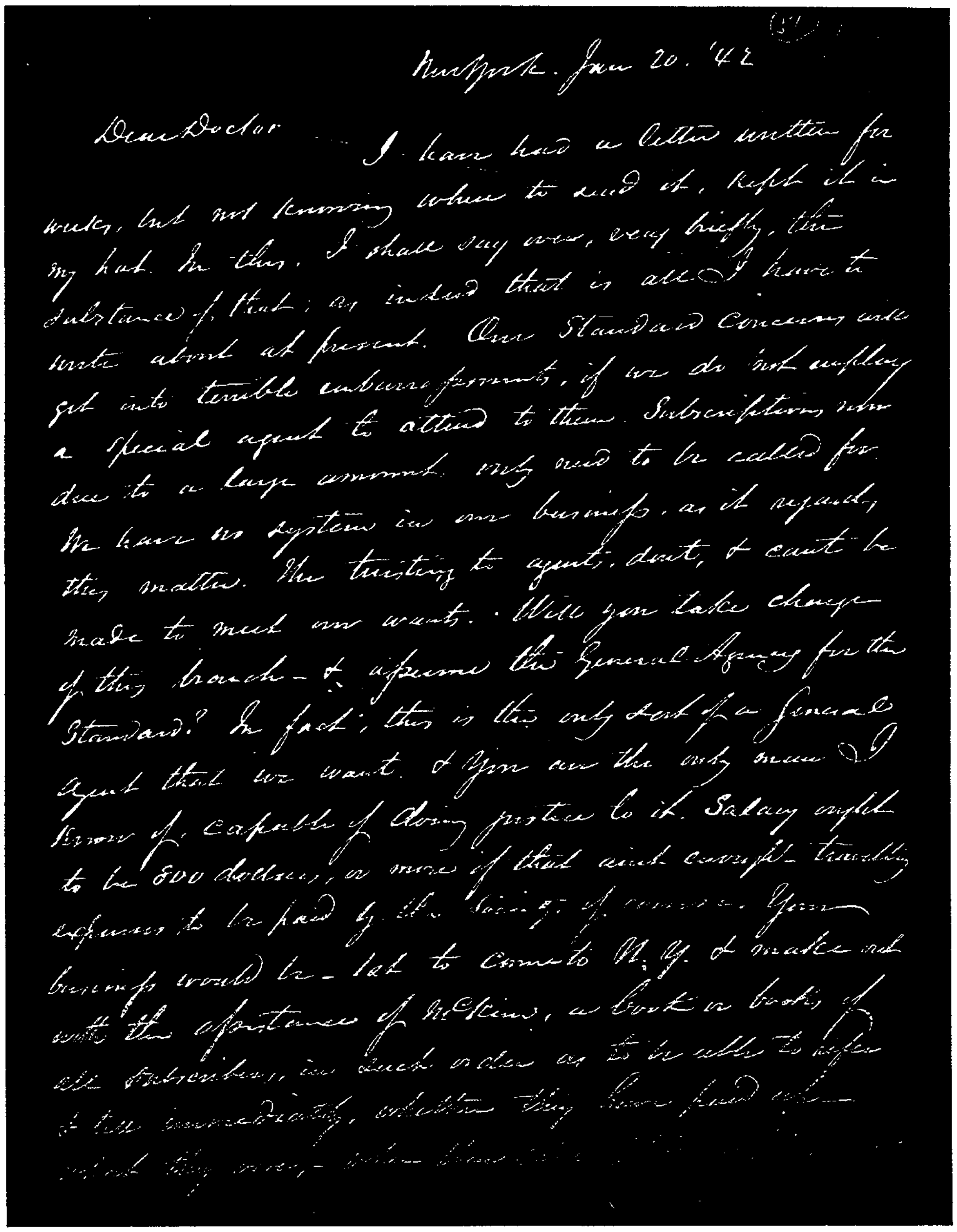
The values are then plugged back into the above equation to compute the error E_{SLH}. The error E_{SLH} is an estimate of how dissimilar two words are and the words can, therefore, be ranked according to it.

It will be assumed that the variation for valid words is not too large. This implies that if A_{11} and A_{22} are considerably different from 1, the word is probably not a valid match.

Note: The SLH algorithm assumes that pruning on the basis of the area and aspect ratio thresholds is performed.

3.8 Experiments

The two matching techniques were tested on two handwritten pages, each written by a different writer. The first page can be obtained from



New York. Jan 20. '42

Dear Doctor — I have had a letter written for
weeks, but not knowing where to send it, kept it in
my hat. In this, I shall say over, very briefly, the
substance of that; as indeed that is all I have to
write about at present. Our Standard Concerns will
get into terrible embarrassments, if we do not employ
a special agent to attend to them. Subscriptions now
due to a large amount, only need to be called for.
We have no system in our business, as it regards
this matter. The trusting to agents, don't, & can't be
made to meet our wants. Will you take charge
of this branch — & assume the General Agency for the
Standard? In fact; this is the only sort of a General
Agent that we want. & You are the only man I
know of, capable of doing justice to it. Salary ought
to be 800 dollars, or more if that [illegible] except travelling
expenses to be paid by the Society, of course. Your
business would be — 1st to come to N. Y. & make out
with the assistance of McKim, a book or books of
all subscribers, in such order as to be able to refer
to them immediately, whether they have paid or —
what they owe, — when [illegible]

Figure 7: Part of a page from the collected papers of the Hudson family

the DIMUND document server on the internet http://documents.cfar.umd.edu/resources/database/ handwriting.database.html This page will be referred to as the Senior document. The handwriting on this page is fairly neat (see [18] for a picture). The second page is from an actual archival collection - the Hudson collection from the library of the University of Massachusetts (part of the page is shown in Figure (7). This page is part of a letter written by James S. Gibbons to Erasmus Darwin Hudson. The handwriting on this page is difficult to read and the indexing technique helped in deciphering some of the words.

The experiments will show examples of how the matching techniques work on a few words. For more examples of the EDM technique see [18]. For more examples using the SLH technique and comparisons with the EDM technique see [16]. In general, the EDM method ranks most words in the Senior document correctly but ranks some words in the Hudson document incorrectly. The SLH technique performs well on both documents.

Both pages were segmented into words (see [18] for details) The algorithm was then run on the segmented words. In the following figures, the first word shown is the template. After the template, the other words are ranked according to the match error. Note that only the first few results of the matching are shown although *the template has been matched with every word on the page.* The area threshold α was chosen to be 1.2 and the aspect ratio threshold β was chosen as 1.4. The translation values were sampled to within ± 4 pixels in the X direction and ± 1 pixel in the y direction. Experimentally, this gave the best results.

3.9 Results using Euclidean Distance Mapping

The Euclidean Distance Mapping algorithm works reasonably well on the Senior document. An example is shown below.

In Figure (8), the template is the word "Lloyd". The figure shows that the four other instances of "Lloyd" present in the document are ranked before any of the other words. As Table (2) shows, the match errors for other instances of "Lloyd" is less than that for any other word. In the table, the first column is the Token number (this is needed for identification purposes), the second column is a transcription of the word, the third column shows the area in pixels, the fourth gives the match error and the last two columns specify the translation in the x and y directions respectively. Note the significant change in area of the words.

The performance on other words in the Senior document is comparable (for other examples see [18]). This is because the page is written fairly neatly. The performance of the method is expected to correlate with the quality of the handwriting. This was verified by running experiments on a page from the Hudson collection (Figure 7). The handwriting in the Hudson collection is difficult to read even for humans looking at grey-level images at 300 dpi The writing shows wide variations in size - for example, the area of the word "to" varies by as much as 100% ! However, this large a variation is not expected to occur and is not seen when the words are larger. Since humans have difficulty reading this material, we do not expect that the method will perform very well on this document.

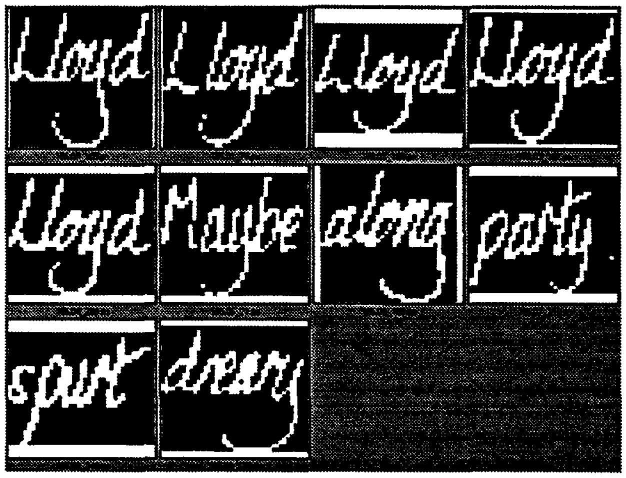


Figure 8: Ranked matches for template "Lloyd" using the EDM algorithm (the rankings are ordered from left to right and from top to bottom).

The Euclidean Distance Mapping technique fails for the template "Standard" in the Hudson document (see Figure (9)). The failure occurs because the two instances of "Standard" are written differently. The template "Standard" has a gap between the "t" and the "a". This gap is not present in the second example of "Standard" (this is more clearly visible in Figure (10). A technique to model some distortions is, therefore, necessary.

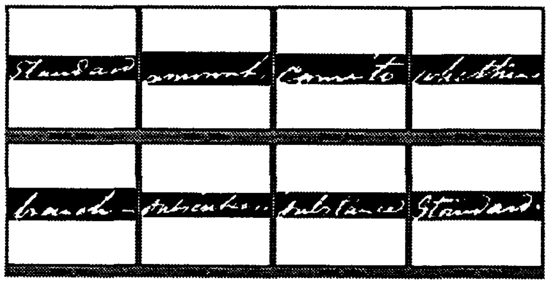


Figure 9: Rankings for template "Standard" using the EDM algorithm(the rankings are ordered from left to right and from top to bottom).

3.10 Experiments Using the SLH Algorithm

The SLH algorithm handles affine distortions and is, therefore more powerful then the EDM algorithm. Since

Token	Word	Area	E_{EDMmin}	Xshift	Yshift
105	Lloyd	1360	0.000	0	0
70	Lloyd	1224	0.174	0	0
165	Lloyd	1230	0.175	-2	0
197	Lloyd	1400	0.194	4	0
239	Lloyd	1320	0.197	-3	0
21	Maybe	1147	0.199	-1	0
180	along	1156	0.200	1	0
215	party	1209	0.202	1	0
245	spurt	1170	0.205	-1	0
121	dreary	1435	0.206	3	0

Table 2: Rankings and match Errors for template "Lloyd".

Token	Word	Area	CP	E_{SLH}	A		T
105	Lloyd	1368	233	0.00	1.00	0.00	0.00
					0.00	1.00	0.00
197	Lloyd	1400	199	1.302	0.96	-0.04	1.58
					0.01	1.04	0.14
70	Lloyd	1224	176	1.356	0.94	0.09	-1.02
					0.03	0.92	-1.38
165	Lloyd	1230	189	1.631	1.03	0.05	-0.43
					-0.01	0.87	-2.60
239	Lloyd	1320	203	1.795	0.99	-0.05	1.44
					0.03	1.07	2.21
157	lawyer	1518	185	3.393	0.96	-0.03	1.89
					0.05	1.11	0.03
240	Selwyn	1564	188	3.673	0.94	0.06	-4.23
					0.05	1.05	-0.75
91	thought	1178	181	3.973	0.97	0.03	2.33
					-0.01	1.08	2.91

Table 3: Rankings and Match Errors for template "Lloyd" Using SLH Algorithm.

the current version of the SLH algorithm is slow, the initial matches were pruned using the EDM algorithm and then the SLH algorithm run on the pruned subset.

Experiments were performed using both the Senior document and the Hudson documents. A few examples are shown here (for more details see [16]). For the Senior documents the same pruning ratios were chosen as before. To account for the large variations in the Hudson papers, the area threshold α was fixed at 1.3 and the aspect ratio threshold at 1.7. The value of σ depends on the expected translation. Since it is small, $\sigma = 2.0$. A lower value of $\sigma = 1.5$ yielded poorer results.

The matches for the template "Lloyd" are shown in Table (3). The succesive columns of the table, tabulate the Token Number, the transcription of the word, the area of the word image, the number of corresponding points recovered by the SLH algorithm, the match error E_SLH using the SLH algorithm and the affine transform. The entries are ranked according to the match error E_{SLH}. If either of A_{11} or A_{22} is less than 0.8 or greater than 1/0.8, that word is eliminated from the rankings. A comparison with Table (2) shows that the rankings change. This is not only true of the invalid words (for example the sixth entry in Table (2) is "Maybe" while the sixth entry in Table (3) is "lawyer" but is also true of the "Lloyd"'s. Both tables rank instances of "Lloyd" ahead of other words. The technique also shows a much greater discrimination in match error - the match error for "lawyer" is almost double the match error for the fifth "Lloyd".

The method was also run on the Hudson document (Figure (7)) and it ranked most of the words correctly on this document. As an example, we look at the word "Standard" on which the EDM method did not rank correctly. The SLH method produces the correct ranking inspite of the significant distortions in the word (see Figure (10)).

3.10.1 Recall–Precision Results

Indexing and retrieval techniques may be evaluated using recall and precision. Recall is defined as the "proportion of relevant documents actually retrieved" while precision is defined as the "proportion of retrieved documents that are relevant" [31]. Figure 3.10.1 shows the recall–precision results for both algorithms on the Senior

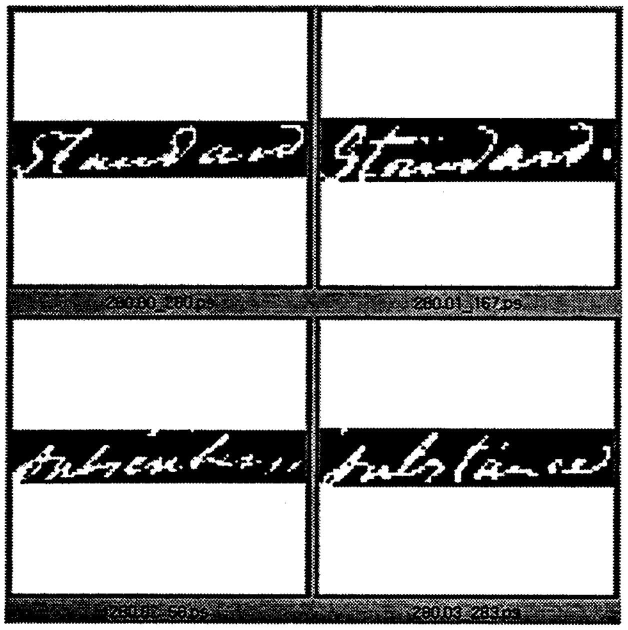


Figure 10: Rankings for template "Standard" for the SLH algorithm (the rankings are ordered from left to right and from top to bottom).

document. The two EDM graphs are for two different values of the area ratio (1.22 and 1.3). Notice that they do not differ significantly, thus showing that the exact values of the area ratio are not significant. The average precision for the EDM and SLH algorithms on the Senior document are 79.7 % and 86.3 % respectively. Note that SLH performs significantly better than EDM. Similar results are obtained with the Hudson document.

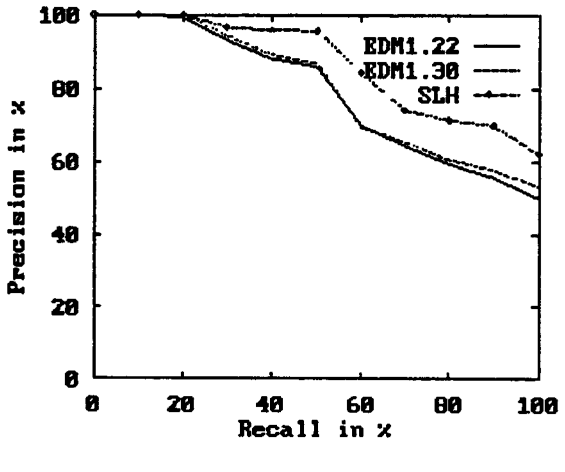


Figure 11: Recall precision results for Senior document

4 Image Retrieval

The indexing and retrieval of images using their content is a poorly understood and difficult problem. A person using an image retrieval system usually seeks to find semantic information. For example, a person may be looking for a picture of a leopard from a certain viewpoint. Or alternatively, the user may require a picture of Abraham Lincoln from a particular viewpoint.

Retrieving semantic information using image content is difficult to do. The automatic segmentation of an image into objects is a difficult and unsolved problem in computer vision. However, many image attributes like color, texture, shape and "appearance" are often directly correlated with the semantics of the problem. For example, logos or product packages (e.g., a box of Tide) have the same color wherever they are found. The coat of a leopard has a unique texture while Abraham Lincoln's appearance is uniquely defined. These image attributes can often be used to index and retrieve images.

The Center has carried out pioneering research in this area. The Center conducts research in both color based image retrieval see and appearance based image retrieval (the methods applied to appearance based image retrieval may also be directly applied to texture based image retrieval). We will now discuss appearance based retrieval (the reader is referred to [3] for discussions about the color based retrieval.

4.1 Retrieval by Appearance

Some attempts have been made to retrieve objects using their shape [4, 24]. For example, the QBIC system [4], developed by IBM, matches binary shapes. It requires that the database be segmented into objects. Since automatic segmentation is an unsolved problem, this requires the user to manually outline the objects in the database. Clearly this is not desirable or practical.

Except for certain special domains, all methods based on shape are likely to have the same problem. An object's appearance depends not only on its three dimensional shape, but also on the object's albedo, the viewpoint from which it is imaged and a number of other factors. It is non-trivial to separate the different factors constituting an object's appearance. For example, it is usually not possible to separate an object's three dimensional shape from the other factors.

The Center has overcome this difficulty by developing methods to retrieve objects using their appearance [26, 27, 19, 25]. The methods involve finding objects similar in appearance to an example object specified by the query.

To the best of our knowledge, ours is the first general query by appearance image retrieval system. Systems have been built to retrieve specific objects like faces (e.g., [29])). However, these systems require a number of training examples and it is not clear whether they can be generalized to retrieve other objects.

Some of the salient features of our system include:

1. The ability to retrieve "similar" images. This is in contrast with techniques which try to recover the *same* object. In our system, a car used as a query

will also retrieve other cars rather than retrieving only cars of a specific model.

2. The ability to retrieve images embedded in a background (see for example the cars in Figure 13 which appear against various backgrounds).
3. It does not require any prior manual segmentation of the database.
4. No training is required.
5. It can handle a range of variations in size.
6. It can handle 3D viewpoint changes up to about 20 to 25 degrees.

The user constructs the query by taking an example picture, and marking regions which she considers important aspects of the object. The query may be refined later depending on the retrieval results. Consider, for example, the first car shown in Figure 4.1. The user marks the region shown in the figure using a mouse. Notice that the region reflects the fact that wheels are central to a car. The user's query in this situation is to find visually similar objects (i.e., other cars) from a similar viewpoint (where the viewpoint can vary up to 25 degrees from the query).

The database images are filtered with derivatives of Gaussians at multiple scales. Derivatives of the first and second order are used. Differential invariants (invariants to 2D rotation) are created using the derivatives. [19, 25]. An inverted list is constructed from these invariants. The inverted list is indexed using the value of each invariant. The entire computation may be carried out off-line.

The on-line computation consists of calculating invariants for points in the query (which is a region in the image). Points with similar invariant values are now recovered from the database by indexing on the invariant values. The points obtained by indexing must also satisfy certain spatial constraints. That is, the values of the invariants at a pixel and at some of its neighbors must match. This ensures that the indexing scheme preserves the spatial layout of objects. Points which satisfy this spatial relationship vote and the database images are ranked on the basis of this vote.

The scheme described above works if the object is roughly the same size in the query and the image database. In practice it is quite common for the objects to be of different sizes in a database. The variation in size is handled by doing a search over scale space. That is, the query is filtered with Gaussian derivatives of different standard deviations [14, 13, 12] and the image simultaneously warped. This allows objects over a range of sizes to be matched [26, 27].

The query is outlined by the user with a mouse Figure 4.1. Figure 13 shows the results of a query. Notice that a large number of cars with white wheels have been retrieved. For more examples, see [19, 25]. This retrieval

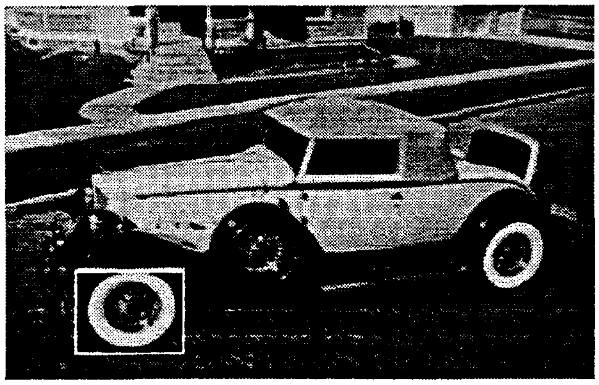


Figure 12: Car Query for retrieval by indexing

was performed on a database of 1600 images taken from the Internet, the Library of Congress and other sources. The database consists of faces, monkeys, apes, cars, diesel and steam locomotives and a few houses. Lighting and camera parameters are not known.

5 Conclusion

This paper has described the multimedia indexing and retrieval work being done at the Center for Intelligent Information Retrieval. Work on systems for finding text in images, indexing archival handwritten documents and image retrieval by content has been described. The research described is part of an on-going research effort focused on indexing and retrieving multimedia information in as many ways as possible. The work described here has many applications, principally in the creation of the digital libraries of the future.

6 Acknowledgements

This paper includes research contributions by Victor Wu and Srinivas Ravela of the multimedia indexing and retrieval group. David Hirvonen and Adam Jenkins provided programming support. Ed Riseman gave comments on some of this work. I would like to thank Bruce Croft and CIIR for supporting this work and Gail Giroux and the University of Massachusetts Library for the scanned page from the Hudson collection.

References

[1] M. Bokser. Omnidocument technologies. *Proceedings IEEE*, 80(7):1066–1078, 1992.

[2] Per-Erik Danielsson. Euclidean distance mapping. *Computer Graphics and Image Processing*, 14:227–248, 1980.

[3] M. Das, E. M. Riseman, and B. A. Draper. Focus : Searching for multi-colored objects in a diverse image database. *accepted to the IEEE CVPR '97*, June 1997.

[4] Myron Flickner et al. Query by image and video content: The qbic system. *IEEE Computer Magazine*, pages 23–30, Sept. 1995.

[5] L. D. Wilcox F. R. Chen, D. S. Bloomberg. Spotting phrases in lines of imaged text. In *Proceedings of the SPIE conf. on Document Recognition II*, volume 2422, pages 256–269, San Jose, CA, Feb. 1995.

[6] Paul Filiski and Jonathan J. Hull. Keyword selection from word recognition results using definitional overlap. In *Third Annual Symposium on Document Analysis and Information Retrieval, UNLV, Las Vegas*, pages 151–160, 1994.

[7] L. Fletcher and R. Kasturi. A robust algorithm for text string separation from mixed text/graphics images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(6):910–918, Nov. 1988.

[8] Anil K. Jain and Sushil Bhattacharjee. Text Segmentation Using Gabor Filters for Automatic Document Processing. *Machine Vision and Applications*, 5, 1992.

[9] G. J. F. Jones, J. T. Foote, K. Sparck Jones, and S. J. Young. Video mail retrieval: The effect of word spotting accuracy on precision. In *International Conference on Acoustics, Speech and Signal Processing*, volume 1, pages 309–316, 1995.

[10] Siamak Khoubyari and Jonathan J.Hull. Keyword location in noisy document image. In *Second Annual Symposium on Document Analysis and Information Retrieval, UNLV, Las Vegas*, pages 217–231, 1993.

[11] J. Malik and P. Perona. Preattentive texture discrimination with early vision mechanisms. *Journal of the Optical Society of America A*, 7(5):923–932, May 1990.

[12] R. Manmatha. Image matching under affine deformations. In *Invited Paper, Proc. of the 27nd Asilomar IEEE Conf. on Signals, Systems and Computers*, pages 106–110, 1993.

[13] R. Manmatha. A framework for recovering affine transforms using points, lines or image brightnesses. In *Proc. Computer Vision and Pattern Recognition Conference*, pages 141–146, 1994.

[14] R. Manmatha. Measuring the affine transform using gaussian filters. In *Proc. 3rd European Conference on Computer Vision*, pages 159–164, 1994.

[15] R. Manmatha and W. B. Croft. Word spotting: Indexing handwritten manuscripts. In Mark Maybury, editor, *Intelligent Multi-media Information Retrieval*. AAAI/MIT Press, April 1998.

[16] R. Manmatha, Chengfeng Han, and E. M. Riseman. Word spotting: A new approach to indexing handwriting. Technical Report CS-UM-95-105, Computer Science Dept, University of Massachusetts at Amherst, MA, 1995.

[17] R. Manmatha, Chengfeng Han, and E. M. Riseman. Word spotting: A new approach to indexing handwriting. In *Proc. Computer Vision and Pattern Recognition Conference*, pages 631–637, 1996.

[18] R. Manmatha, Chengfeng Han, E. M. Riseman, and W. B. Croft. Indexing handwriting using word matching. In *Digital Libraries '96: 1st ACM International Conference on Digital Libraries*, pages 151–159, 1996.

[19] R. Manmatha and S. Ravela. A syntactic characterization of appearance and its application to image retrieval. In *Proceedings of the SPIE conf. on Human Vision and Electronic Imaging II*, volume 3016, San Jose, CA, Feb. 1997.

[20] S. Mori, C. Y. Suen, and K. Yamamoto. Historical review of ocr research and development. *Proceedings of the IEEE*, 80(7):1029–1058, July 1992.

[21] G. Nagy, S. Seth, and M. Viswanathan. A Prototype Document Image Analysis System for Technical Journals. *Computer*, pages 10–22, July 1992.

[22] Lawrence O'Gorman. The Document Spectrum for Page Layout Analysis. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 15(11):1162–1173, Nov. 1993.

[23] Theo Pavlidis and Jiangying Zhou. Page Segmentation and Classification. *CVGIP: Graphical Models and Image Processing*, 54(6):484–496, Nov. 1992.

[24] A. Pentland, R. W. Picard, and S. Sclaroff. Photobook: Tools for content-based manipulation of databases. In *Proc. Storage and Retrieval for Image and Video Databases II,SPIE*, volume 185, pages 34–47, 1994.

[25] S. Ravela and R. Manmatha. Image retrieval by appearance. In *Accepted to the 20th Intl. Conf. on Research and Development in Information Retrieval (SIGIR'97)*, July 1997.

[26] S. Ravela, R. Manmatha, and E. M. Riseman. Image retrieval using scale-space matching. In Bernard Buxton and Roberto Cipolla, editors, *Computer Vision - ECCV '96*, volume 1 of *Lecture Notes in Computer Science*, Cambridge, U.K., April 1996. 4th European Conf. Computer Vision, Springer.

[27] S. Ravela, R. Manmatha, and E. M. Riseman. Scale space matching and image retrieval. In *Proc. DARPA Image Understanding Workshop*, 1996.

[28] G. L. Scott and H. C. Longuet-Higgins. An algorithm for associating the features of two patterns. *Proc. Royal Society of London B*, B244:21–26, 1991.

[29] M. Turk and A. Pentland. Eigenfaces for recognition. *J. of Cognitive NeuroScience*, 3:71–86, 1991.

[30] H.R. Turtle and W.B. Croft. A comparison of text retrieval models. *Computer Journal*, 35(3):279–290, 1992.

[31] C. J. van Rijsbegen. *Information Retrieval*. Butterworths, 1979.

[32] F. Wahl, K. Wong, and R. Casey. Block segmentation and text extraction in mixed text/image documents. *Computer Vision Graphics and Image Processing*, 20:375–390, 1982.

[33] D. Wang and S. N. Srihari. Classification of newspaper image blocks using texture analysis. *Computer Vision Graphics and Image Processing*, 47:327–352, 1989.

[34] V. Wu, R. Manmatha, and E. M. Riseman. Finding Text In Images. *Technicial Report 97-09, Computer Science Department, UMass, Amherst, MA*, 1997.

[35] V. Wu, R. Manmatha, and E. M. Riseman. Finding Text In Images. *accepted to the Second ACM Intl. conf. on Digitial Libraries DL'97*, July 1997.

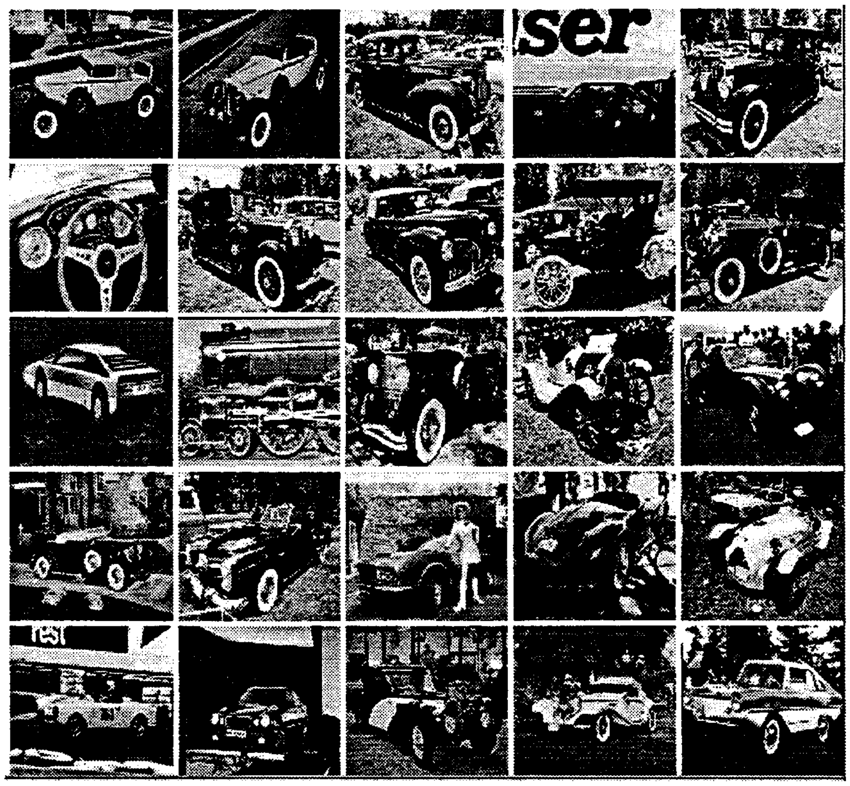


Figure 13: The results of the car query.

Document Image Matching and Retrieval Techniques

Jonathan J. Hull, John Cullen, and Mark Peairs
Ricoh California Research Center
2882 Sand Hill Road, Suite 115
Menlo Park, CA 94025
hull@crc.ricoh.com

Abstract

A brief survey is presented of several techniques for document image matching and retrieval developed at the Ricoh California Research Center. These methods are given a document image as input and locate visually similar or identical copies of the same image in a large database.

1. Introduction

Document image matching algorithms are useful in applications where the objective is to locate visually similar or identical copies of a given document in a large database. Applications of this technology include *automatic filing* in which a user would like to store documents with a similar appearance (e.g., business letters, utility bills, etc.) in the same location. Content-based *retrieval* is another application in which a single sheet from a multi-page original is used to locate the other pages.

Figure 1 illustrates a confidential document monitoring system in which the objective is to determine whether a given document image exists in a database. Such an approach could be used to monitor facsimile traffic. The transmission or reception of specific images could be recorded or alerts issued when certain documents were processed.

The rest of this paper describes several techniques for document image matching. A general framework is presented first. This is followed by a brief presentation of two algorithms: one that uses symbolic features extracted from text and another that uses features extracted directly from CCITT group 3 or group 4 fax compressed images.

2. General Framework

A general framework for document image matching is presented in Figure 2. This follows the paradigm of hypothesis generation and testing commonly used to solve computer vision problems. Features are extracted from an input document image as well as the images in a database. Those feature descriptions are compared by a similarity detection algorithm that locates a group of N documents that are visually similar to a given image.

Equivalence detection is performed by extracting another (perhaps different) feature description from both the input document image and the N visually similar document images output by the similarity detection step. The output of equivalence detection are duplicates of the input document that are contained in the database.

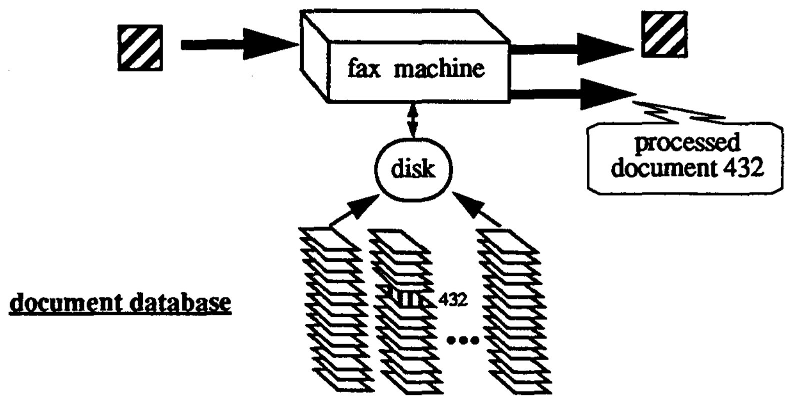


Figure 1. Fax alerting application for content-based document image matching (from [2]).

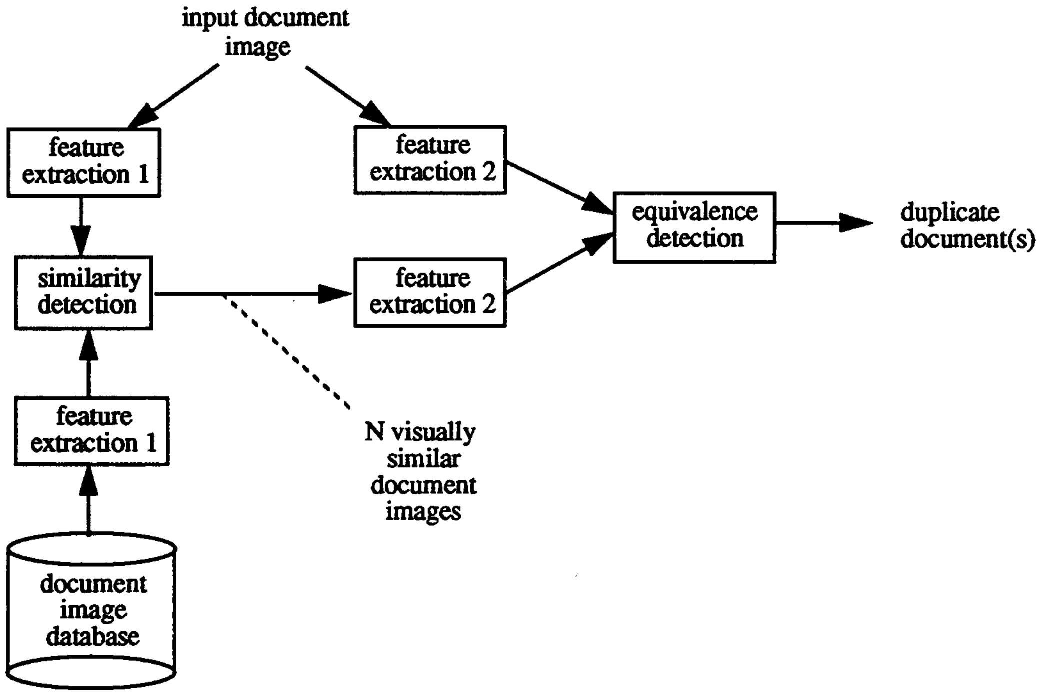


Figure 2. General framework for document image matching.

2.1 Similarity Detection

A technique for similarity detection is reported in [3] that calculates a feature vector by imposing a fixed grid on a document image and counting the number of connected components of a certain size that occur in each grid cell. Only connected components that are approximately the size of characters are counted.

Experimental results showed that as few as 9 features (a 3x3 grid) could be used to locate duplicate documents in a database of 979 images with a 98% accuracy. The test set for this application was extracted from University of Washington CDROM1 [6] and the object of the test was to locate duplicates of the 125 images tagged as 'E' that are indicated by a corresponding 'S' tag. The algorithm was applied to each of the 979 images in turn. The ten images with the minimum Euclidean distance to the input document were output. The result quoted above means that 98% of the time the correct match was contained among the ten documents with the minimum Euclidean distance to the input.

2.2 Equivalence Detection Using Word Lengths

A method for detecting equivalent document images in a large database is reported in [1]. This technique estimated the number of characters in each word of text and formed features by concatenating the lengths of M adjacent words. For example, with M=3, the phrase "the rain in Spain" can be described by the features 3-4-2 and 4-2-5. This feature description is tolerant to noise in that incorrectly estimating the length of any word changes at most M features. Also, this feature description maps easily from images of documents to their symbolic representations as ASCII files, independent of how they are formatted.

Each such feature was used as a key for a hash function and the identity of the passage of text from which each feature was extracted was stored in a hash table. At run-time, the features extracted from a text passage were hashed and votes were accumulated in the hash table. Documents in the database that obtained a suitable number of votes were assumed to match the input image.

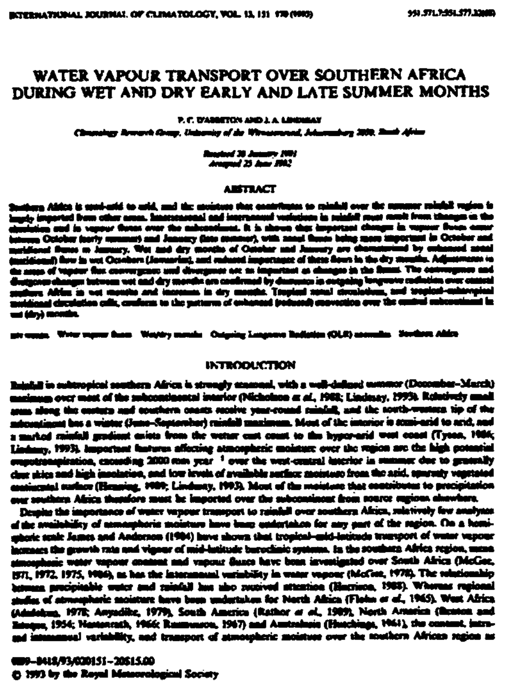
A result of this work is the observation that the sequence of word lengths extracted from a passage of text can provide a unique identifier for the passage. Experimental results showed that as few as 50 descriptors of length M=6 can be used to locate a matching document in a database of 997 images. Increasing the value of M reduces the number of descriptors and increases the number of documents that can be described by this method.

An adaptation of the word length hashing method was used in a paper-based technique for document image retrieval [5]. Small iconic representations for document images were printed in such a way that the visual appearance of the document was retained. However, an address of the original high resolution scanned image of the document in a large database could still be derived directly from the icon. This was done by printing text so that the numbers of characters in each word could be determined from a scanned image of the icon.

An example of an original document and the icon derived from it are shown in Figure 3. It can be seen that the general appearance of the document is retained in the icon. This figure also shows the actual size of an icon. Experimental results showed that 49 of these icons could be printed at 600 dpi on a single sheet of paper.

2.3 Equivalence Detection Using Pass Codes in Fax Images

Another technique for equivalence detection (outlined in Figure 4) addresses the application scenario presented in Figure 1 [2]. The input image (also referred to as the query) and the images in the database are assumed to be compressed in CCITT group 3 or group 4 format. The x,y locations of the centers of pass coded runs in each image are extracted. A subset of pass code locations in each image are chosen that are contained in rectangular patches of text. The two-dimensional arrangements of x,y locations are compared using a modified Hausdorff distance measure that compensates for x-y translation [4]. It is assumed that skew would be normalized by preprocessing using a technique similar to that proposed in [7]. It is further assumed that it is not necessary to compensate for scale change. A binary decision is output that indicates whether the query image is equivalent to a given image from the database. This procedure is used to compare a query image sequentially to each image in the database.



INTERNATIONAL JOURNAL OF CLIMATOLOGY, VOL. 13, 151–170 (1993) 551.571.7:551.577.2(680)

WATER VAPOUR TRANSPORT OVER SOUTHERN AFRICA DURING WET AND DRY EARLY AND LATE SUMMER MONTHS

P. C. D'ABRETON AND J. A. LINDESAY

Climatology Research Group, University of the Witwatersrand, Johannesburg 2050, South Africa

Received 30 January 1991
Accepted 25 June 1992

ABSTRACT

Southern Africa is semi-arid to arid, and the moisture that contributes to rainfall over the summer rainfall region is largely imported from other areas. Intraseasonal and interannual variations in rainfall must result from changes in the circulation and in vapour fluxes over the subcontinent. It is shown that important changes in vapour fluxes occur between October (early summer) and January (late summer), with zonal fluxes being more important in October and meridional fluxes in January. Wet and dry months of October and January are characterized by enhanced zonal (meridional) flow in wet Octobers (Januaries), and reduced importance of these flows in the dry months. Adjustments to the areas of vapour flux convergence and divergence are as important as changes in the fluxes. The convergence and divergence changes between wet and dry months are confirmed by decreases in outgoing longwave radiation over central southern Africa in wet months and increases in dry months. Tropical zonal circulations, and tropical–extratropical meridional circulation cells, conform to the patterns of enhanced (reduced) convection over the central subcontinent in wet (dry) months.

KEY WORDS Water vapour fluxes Wet/dry months Outgoing Longwave Radiation (OLR) anomalies Southern Africa

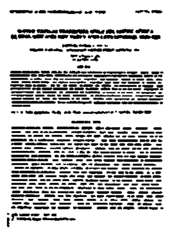
INTRODUCTION

Rainfall in subtropical southern Africa is strongly seasonal, with a well-defined summer (December–March) maximum over most of the subcontinental interior (Nicholson *et al.*, 1988; Lindesay, 1993). Relatively small areas along the eastern and southern coasts receive year-round rainfall, and the south-western tip of the subcontinent has a winter (June–September) rainfall maximum. Most of the interior is semi-arid to arid, and a marked rainfall gradient exists from the wetter east coast to the hyper-arid west coast (Tyson, 1986; Lindesay, 1993). Important features affecting atmospheric moisture over the region are the high potential evapotranspiration, exceeding 2000 mm year $^{-1}$ over the west-central interior in summer due to generally clear skies and high insolation, and low levels of available surface moisture from the arid, sparsely vegetated continental surface (Henning, 1989; Lindesay, 1993). Most of the moisture that contributes to precipitation over southern Africa therefore must be imported over the subcontinent from source regions elsewhere.

Despite the importance of water vapour transport to rainfall over southern Africa, relatively few analyses of the availability of atmospheric moisture have been undertaken for any part of the region. On a hemispheric scale James and Anderson (1984) have shown that tropical–mid-latitude transport of water vapour increases the growth rate and vigour of mid-latitude baroclinic systems. In the southern Africa region, mean atmospheric water vapour content and vapour fluxes have been investigated over South Africa (McGee, 1971, 1972, 1975, 1986), as has the interannual variability in water vapour (McGee, 1978). The relationship between precipitable water and rainfall has also received attention (Harrison, 1988). Whereas regional studies of atmospheric moisture have been undertaken for North Africa (Flohn *et al.*, 1965), West Africa (Adedokun, 1978; Anyadike, 1979), South America (Rathor *et al.*, 1989), North America (Benton and Estoque, 1954; Hastenrath, 1966; Rasmusson, 1967) and Australasia (Hutchings, 1961), the content, intra- and interannual variability, and transport of atmospheric moisture over the southern African region as

0899-8418/93/020151-20$15.00
© 1993 by the Royal Meteorological Society

(a)



(b)

Figure 3. Original image (a) and the icon derived from it (b).

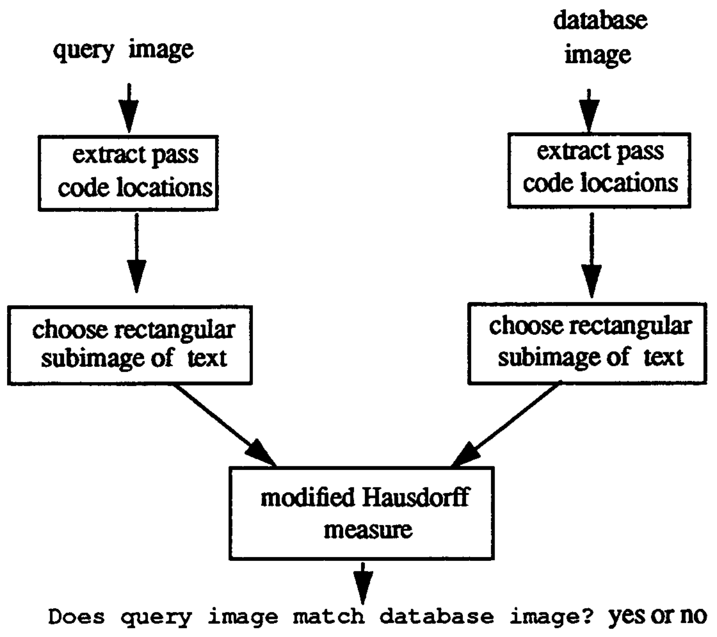

query image
database image
extract pass code locations
extract pass code locations
choose rectangular subimage of text
choose rectangular subimage of text
modified Hausdorff measure
Does query image match database image? yes or no

Figure 4. Document image equivalence detection on CCITT group3 or group 4 fax images (from [2]).

Figure 5 shows a one inch square patch of image data and the pass codes extracted from it. It can be seen that the two-dimensional arrangement of pass codes represents the rigid arrangement of characters in a passage of text. Aspects of the identities of the characters, their font size, line spacing, and word spacing are reflected in the layout of pass codes.

Experimental results on a database of 800 document images showed that the two-dimensional arrangement of pass codes extracted from a one-inch square patch was 95% accurate in locating duplicate documents. On average, only 190 pass coded runs were present in each patch. These experiments were conducted on a subset of the University of Washington CDROM. Analysis of the errors showed that most of them could be accounted for by various non-linear distortions not expected to occur in practice.

3. Discussion and Conclusions

A general framework for document image matching was presented that is similar to a hypothesis generation and test paradigm. A group of N documents that are visually similar to a given query image are first located. The query image is then compared to each of these images to determine whether they are equivalent (i.e., scanned from the same original).

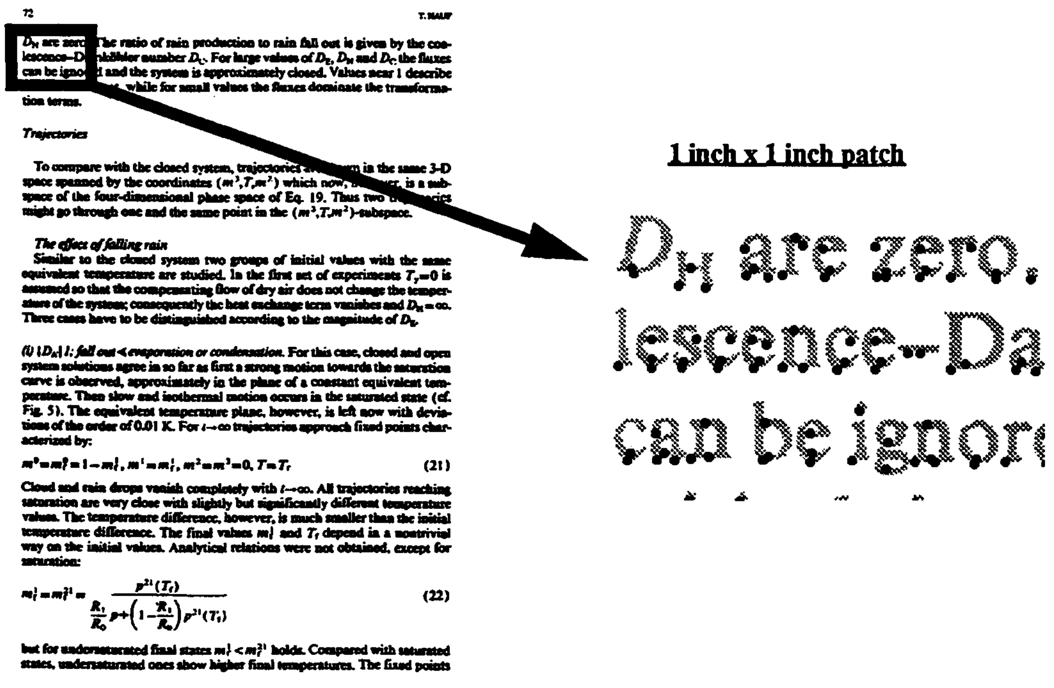
Experimental investigation of several alternative methods for document image equivalence detection has yielded promising results. A method that used pass codes extracted from CCITT fax images has shown particular promise. Further investigation is needed to determine how well this method performs in the context of a complete systems solution that includes an appropriate method for similarity detection. The algorithm should be tested on a larger and more heterogeneous database that includes a range of image qualities.

References

[1] J.J. Hull, "Document image matching and retrieval with multiple distortion-invariant descriptors," in *Document Analysis Systems*, World Scientific, 1995, 379-396.

[2] J.J. Hull, "Document matching on CCITT group 4 compressed images," SPIE Conference on Document Recognition, February 8-14, 1997, San Jose, CA.

[3] J.J. Hull and J. Cullen, "Document image similarity and equivalence detection," International Conference on Document Analysis and Recognition, Ulm, Germany, August, 1997.


1 inch x 1 inch patch
D_H are zero.
lescence–Da
can be ignor

72 T. HAUF

D_H are zero. The ratio of rain production to rain fall out is given by the coalescence–Damköhler number D_C. For large values of D_E, D_H and D_C the fluxes can be ignored and the system is approximately closed. Values near 1 describe [illegible], while for small values the fluxes dominate the transformation terms.

Trajectories

To compare with the closed system, trajectories are drawn in the same 3-D space spanned by the coordinates (m^1, T, m^2) which now, however, is a subspace of the four-dimensional phase space of Eq. 19. Thus two trajectories might go through one and the same point in the (m^1, T, m^2)-subspace.

The effect of falling rain

Similar to the closed system two groups of initial values with the same equivalent temperature are studied. In the first set of experiments $T_y=0$ is assumed so that the compensating flow of dry air does not change the temperature of the system; consequently the heat exchange term vanishes and $D_H=\infty$. Three cases have to be distinguished according to the magnitude of D_E.

(i) $|D_E| 1$; fall out < evaporation or condensation. For this case, closed and open system solutions agree in so far as first a strong motion towards the saturation curve is observed, approximately in the plane of a constant equivalent temperature. Then slow and isothermal motion occurs in the saturated state (cf. Fig. 5). The equivalent temperature plane, however, is left now with deviations of the order of 0.01 K. For $t\to\infty$ trajectories approach fixed points characterized by:

$$m^0 = m_f^0 = 1 - m_f^1,\ m^1 = m_f^1,\ m^2 = m^3 = 0,\ T = T_f \tag{21}$$

Cloud and rain drops vanish completely with $t\to\infty$. All trajectories reaching saturation are very close with slightly but significantly different temperature values. The temperature difference, however, is much smaller than the initial temperature difference. The final values m_f^1 and T_f depend in a nontrivial way on the initial values. Analytical relations were not obtained, except for saturation:

$$m_f^1 = m_f^{21} = \frac{p^{21}(T_f)}{\frac{R_1}{R_0}p + \left(1 - \frac{R_1}{R_0}\right)p^{21}(T_f)} \tag{22}$$

but for undersaturated final states $m_f^1 < m_f^{21}$ holds. Compared with saturated states, undersaturated ones show higher final temperatures. The fixed points

Figure 5. Pass code locations extracted from a document image (from [2]).

[4] D. Huttenlocher, G. Klanderman, and W. Rucklidge, "Comparing images using the Hausdorff distance," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 15, no. 9, September, 1993, 850-863.

[5] M. Peairs, "Iconic paper," Proceedings of the Third International Conference on Document Analysis and Recognition," August 14-16, 1995, Montreal, Canada, 1174-1179.

[6] I. T. Phillips, S. Chen, R. M. Haralick, "CD-ROM document database standard," Proceedings of the Second International Conference on Document Analysis and Recognition, October 20-22, 1993, Tsukuba Science City, Japan, 478-483.

[7] A.L. Spitz, "Skew determination in CCITT group 4 compressed document images," Proceedings of the Symposium on Document Analysis and Information Retrieval, Las Vegas, Nevada, March 16-18, 1992, 11-25.

Applications

Declassification Overview

Tom Curtis
Department of Energy
Declassification Productivity Initiative

Abstract

Openness in government is a core principle of our democracy. Openness provides information the American people need to make informed choices and builds citizen trust in our governmental institutions. However, the federal government is faced with an huge backlog of classified documents to be reviewed. So far, several billion pages have been identified. Motivated by the end of the cold war, a new spirit of openness, and a new Presidential Executive Order on classifying and declassifying National Security Information (NSI) dramatic changes are underway in the review, processing, and release of classified documents. Many federal agencies now have programs underway to use automation to sanitize and release the unclassified portion of this information to the American public. While almost all of these effort currently use automation only to computerize manual review and redaction; these agencies will soon have accumulated tens even hundreds of millions of document images to manage. At the same time the new Electronic Freedom of Information Act (EFOIA) will allow the public access to request such documents and information in electronic form.

One effort to develop leading-edge automated declassification tools is the Department of Energy (DOE) Declassification Productivity Initiative (DPI). The DPI, a program in DOE's Office of Declassification (OD), is a comprehensive effort to use advanced technology to increase document declassification productivity and accuracy. Almost all documents to be reviewed and declassified are only available in non-electronic form. Many of these documents are more than 40 years old; including carbon, Xerox, and facsimile copies; and range from very poor to fair condition. The program's current research and development agenda is focused on the following areas:

Conversion of these documents to usable electronic form including automated scanning, evaluating, enhancing, conversion to text, and meta data extraction.

Automatic duplicate detection and document content characterization.
Classification guidance knowledge representation.
Document content understanding and classified information detection.
Electronic storage and easy public access to declassified and release documents.

While currently, advanced technology development for declassification is limited to a few programs such as DPI the new Executive Order, EFOIA, increasing demand for openness in government, and dramatic shift by the public to electronic access to information will drive the demand for powerful document Processing, access, and management tools.

Declassification

Thomas Curtis
tom.curtis@hq.doe.gov

Office of Declassification
Department of Energy

Why Declassification?

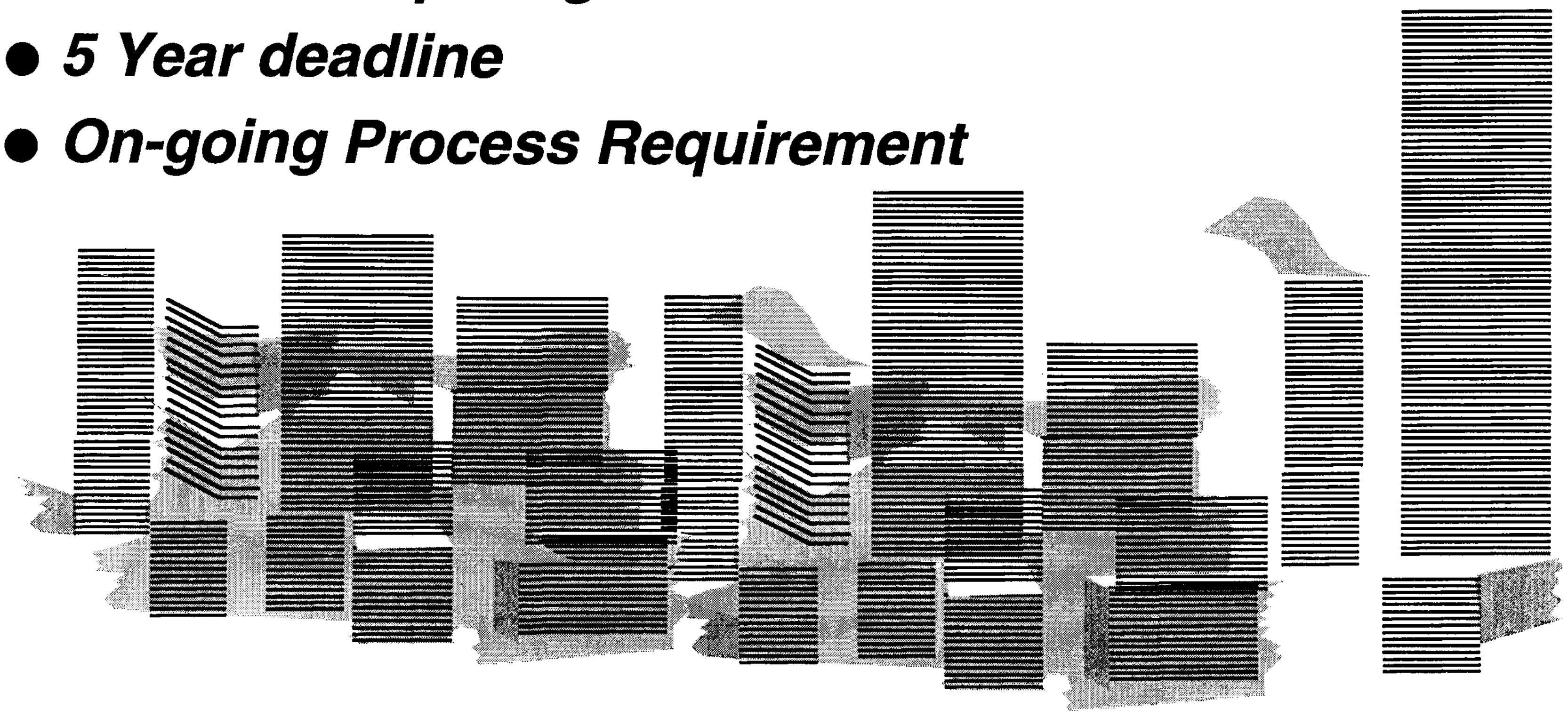
- ***Need Information for Public Debate***
- ***Government Operation is a Trust***
- ***Government Records are History***

Drivers for Change

- ***End of the Cold War***
- ***Public Demands for Openness***
- ***E.O. 12958 - NSI Documents 25 years or older***
- ***Electronic Freedom of Information Act***
- ***Changes in How Public Accesses Information***

Magnitude of the E.O. NSI Problem

- 2 Billion Pages Just for E.O. 12598
- 20 - 25% Requiring Coordination
- 5 Year deadline
- On-going Process Requirement



Document Sanitization
A Paper Process

- ***Locate documents***
- ***Determine if classified***
- ***Mark sensitive parts***
- ***Remove marked portions***
- ***Prepare sanitized copy***
- ***Remove other sensitive non-classified information***
- ***Prepare & release final copy***

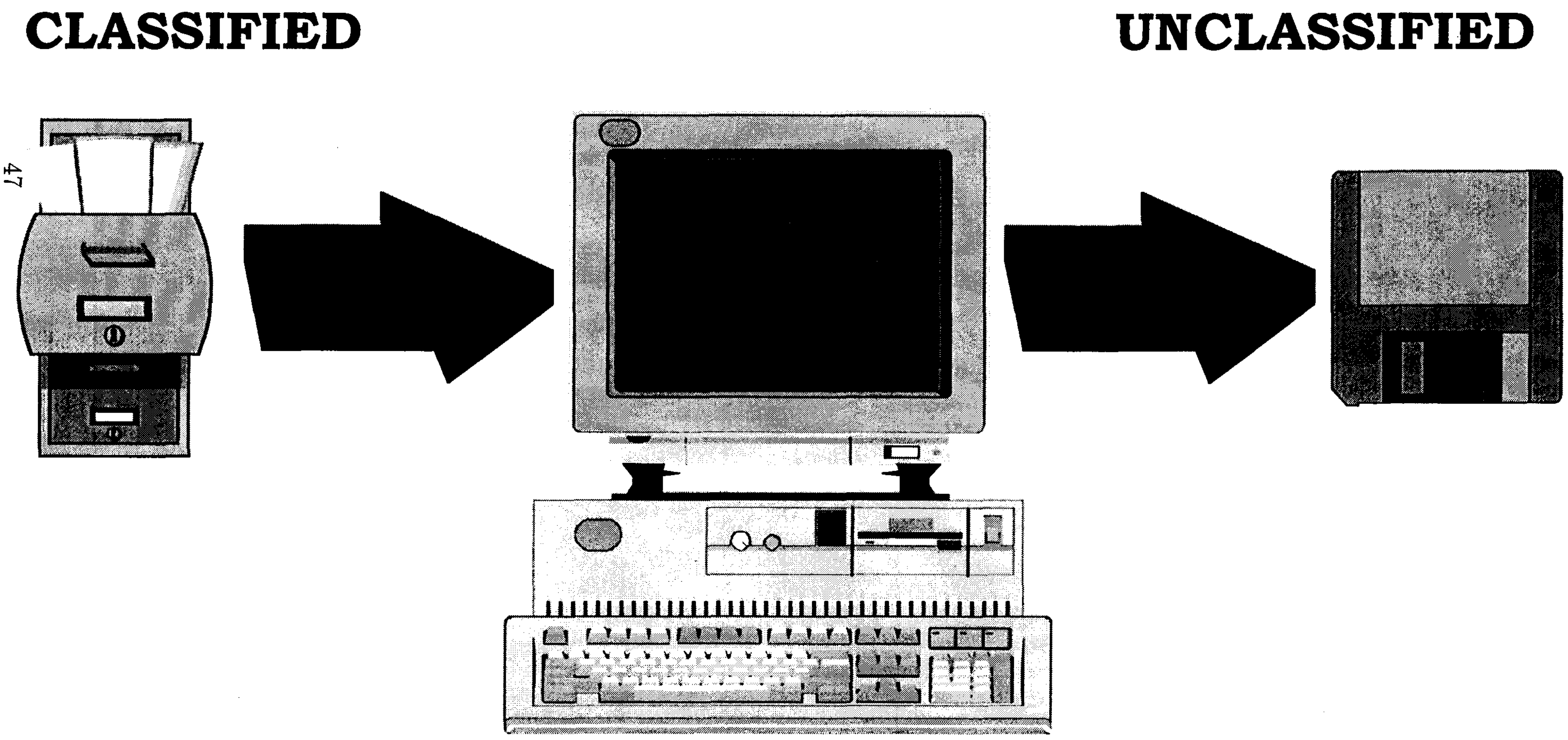
Initial Government Declassification Automation Efforts

- Page Scanning
  - 200-300 dpi - black & white
- Workflow - Electronic Routing
- On Screen Redaction & Annotation
- Indexing
- Quality Assurance
- Output / Distribution
  - Paper
  - Electronic

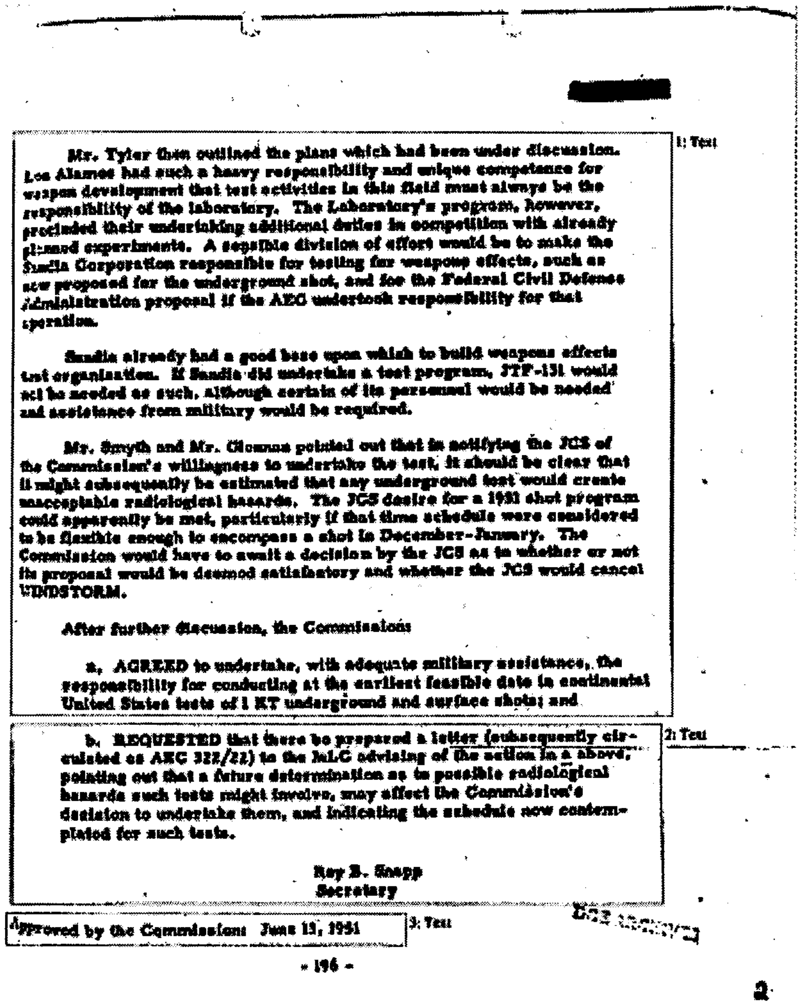
DOE Declassification Productivity Initiative (DPI)

- ***An advanced technology program to improve document declassification productivity and accuracy***
- ***Objective: Develop advanced computer-based document declassification tools***
- ***Ultimately: Develop an intelligent computer declassification system***

What We Ultimately Want



Examples of DOE Documents



[redacted]

Mr. Tyler then outlined the plans which had been under discussion. Los Alamos had such a heavy responsibility and unique competence for weapon development that test activities in this field must always be the responsibility of the laboratory. The Laboratory's program, however, precluded their undertaking additional duties in competition with already planned experiments. A sensible division of effort would be to make the Sandia Corporation responsible for testing for weapons effects, such as now proposed for the underground shot, and for the Federal Civil Defense Administration proposal if the AEC undertook responsibility for that operation. 1: Text

Sandia already had a good base upon which to build weapons effects test organization. If Sandia did undertake a test program, JTF-131 would not be needed as such, although certain of its personnel would be needed and assistance from military would be required.

Mr. Smyth and Mr. Gleason pointed out that in notifying the JCS of the Commission's willingness to undertake the test, it should be clear that it might subsequently be estimated that any underground test would create unacceptable radiological hazards. The JCS desire for a 1951 shot program could apparently be met, particularly if that time schedule were considered to be flexible enough to encompass a shot in December-January. The Commission would have to await a decision by the JCS as to whether or not its proposal would be deemed satisfactory and whether the JCS would cancel WINDSTORM.

After further discussion, the Commission:

a. AGREED to undertake, with adequate military assistance, the responsibility for conducting at the earliest feasible date in continental United States tests of 1 KT underground and surface shots; and

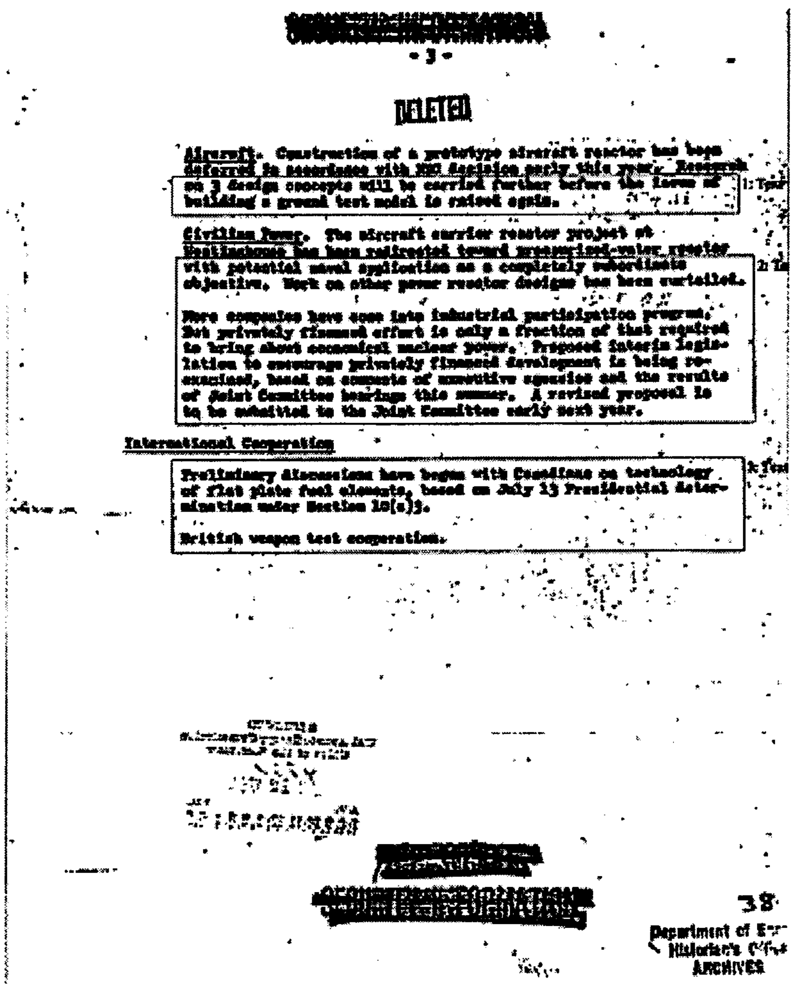
b. REQUESTED that there be prepared a letter (subsequently circulated as AEC 322/22) to the MLC advising of the action in a above, pointing out that a future determination as to possible radiological hazards such tests might involve, may affect the Commission's decision to undertake them, and indicating the schedule now contemplated for such tests. 2: Text

Roy B. Snapp
Secretary

Approved by the Commission: June 13, 1951 3: Text

- 196 -

2



[illegible]

- 3 -

DELETED

Aircraft. Construction of a prototype aircraft reactor has been deferred in accordance with NSC decision early this year. Research on 3 design concepts will be carried further before the issue of building a ground test model is raised again. 1: Text

Civilian Power. The aircraft carrier reactor project at Westinghouse has been redirected toward pressurized-water reactor with potential naval application as a completely subordinate objective. Work on other power reactor designs has been curtailed. 2: Text

More companies have come into industrial participation program. But privately financed effort is only a fraction of that required to bring about economical nuclear power. Proposed interim legislation to encourage privately financed development is being reexamined, based on comments of executive agencies and the results of Joint Committee hearings this summer. A revised proposal is to be submitted to the Joint Committee early next year.

International Cooperation

Preliminary discussions have begun with Canadians on technology of flat plate fuel elements, based on July 13 Presidential determination under Section 10(a)(3). 3: Text

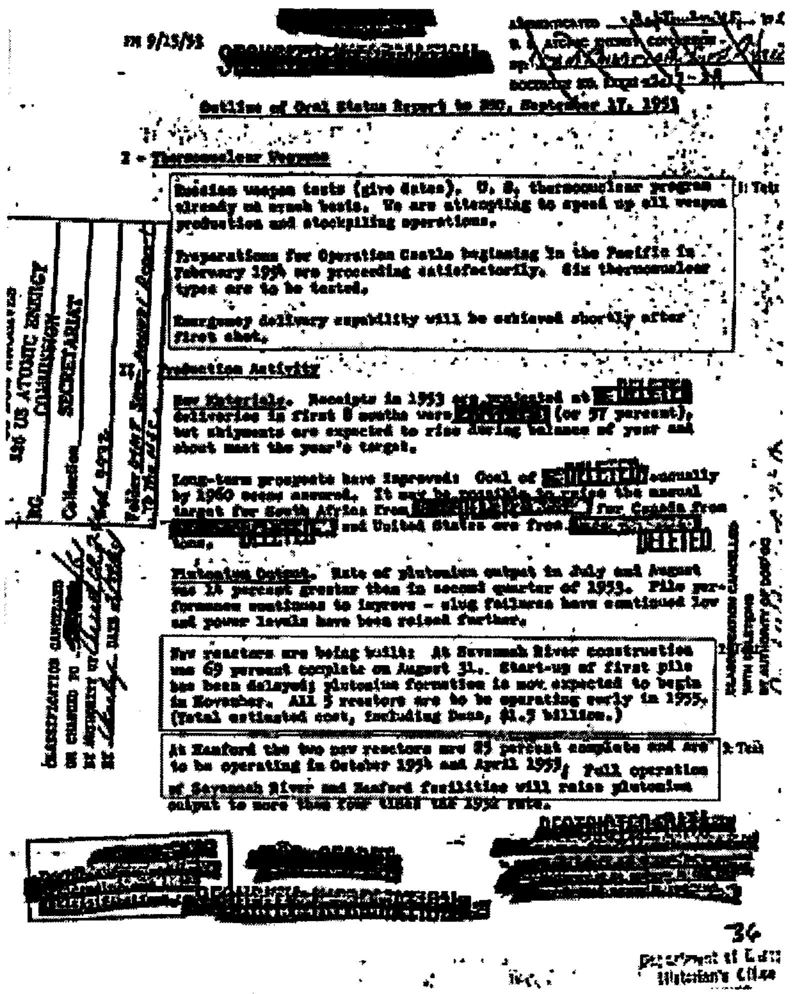
British weapon test cooperation.

[illegible]

38

Department of En
Historian's Office
ARCHIVES

Examples of DOE Documents



US ATOMIC ENERGY COMMISSION
SECRETARIAT
DOC. [illegible]
Collection [illegible]
Folder [illegible]

CLASSIFICATION CANCELLED
OR CHANGED TO [illegible]
BY AUTHORITY OF [illegible]
BY [illegible] DATE [illegible]

FM 9/25/53 [redacted]

[redacted]

~~U. S. ATOMIC ENERGY COMMISSION~~
~~DOCUMENT NO. [illegible]~~

Outline of Oral Status Report to NSC, September 17, 1953

I - Thermonuclear Weapons

Russian weapon tests (give dates). U. S. thermonuclear program already on crash basis. We are attempting to speed up all weapon production and stockpiling operations.

Preparations for Operation Castle beginning in the Pacific in February 1954 are proceeding satisfactorily. Six thermonuclear types are to be tested.

Emergency delivery capability will be achieved shortly after first shot.

II - Production Activity

Raw Materials. Receipts in 1953 are projected at DELETED. Deliveries in first 8 months were [redacted] (or 57 percent), but shipments are expected to rise during balance of year and about meet the year's target.

Long-term prospects have improved: Goal of DELETED annually by 1960 seems assured. It may be possible to raise the annual target for South Africa from [redacted] for Canada from [redacted] and United States are from [redacted] tons. DELETED DELETED

Plutonium Output. Rate of plutonium output in July and August was 14 percent greater than in second quarter of 1953. Pile performance continues to improve - slug failures have continued low and power levels have been raised further.

New reactors are being built: At Savannah River construction was 69 percent complete on August 31. Start-up of first pile has been delayed; plutonium formation is now expected to begin in November. All 5 reactors are to be operating early in 1955. (Total estimated cost, including Dana, $1.5 billion.)

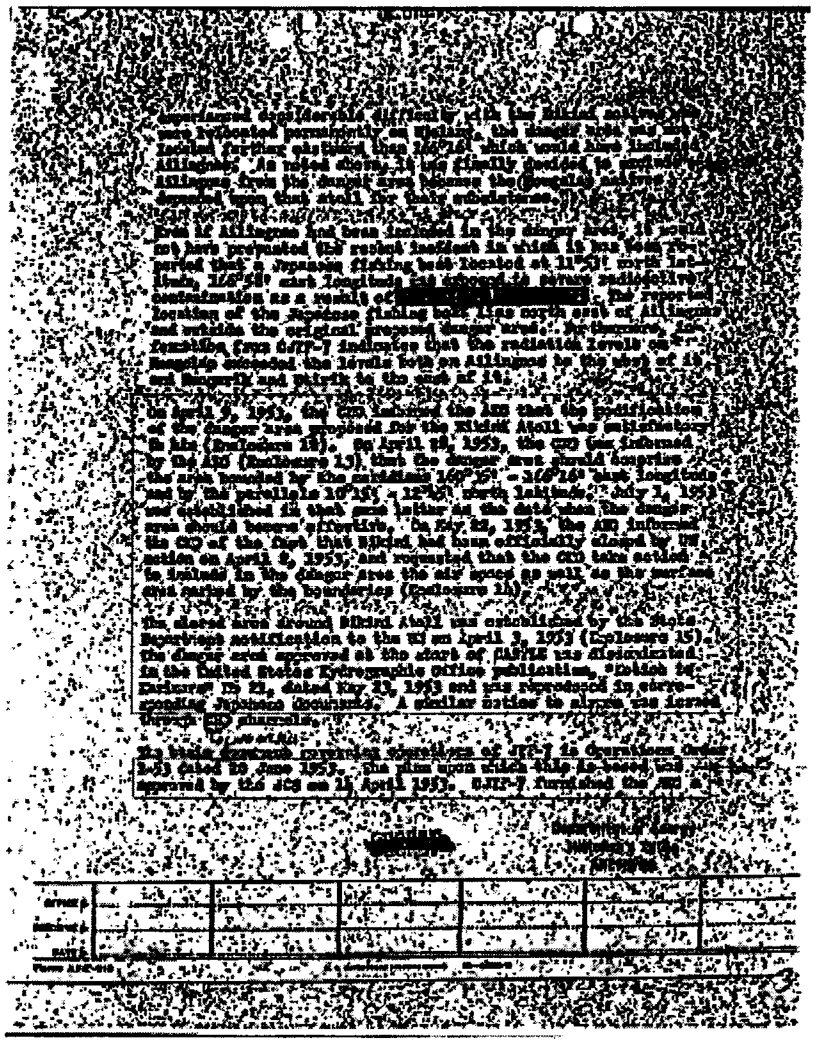
At Hanford the two new reactors are 85 percent complete and are to be operating in October 1954 and April 1955. Full operation of Savannah River and Hanford facilities will raise plutonium output to more than four times the 1952 rate.

CLASSIFICATION CANCELLED
WITH DELETIONS
BY AUTHORITY OF DOE/OC

[redacted]

34

Department of Energy
Historian's Office



experienced considerable difficulty with the Bikini natives who were relocated permanently on Kili, the danger area was not extended further eastward than 166°16' which would have included Ailinginae. As noted above, it was finally decided to exclude Ailinginae from the danger area because the Rongelap natives depended upon that atoll for their subsistence.

Even if Ailinginae had been included in the danger area, it would not have prevented the recent incident in which it has been reported that a Japanese fishing boat located at 11°53' north latitude, 166°58' east longitude was exposed to severe radioactive contamination as a result of [redacted]. The reported location of the Japanese fishing boat lies north east of Ailinginae and outside the original proposed danger area. Furthermore, information from CJTF-7 indicates that the radiation levels on Rongelap exceeded the levels both on Ailinginae to the west of it and Rongerik and Utirik to the east of it.

On April 9, 1953, the CNO informed the AEC that the modification of the danger area proposed for the Bikini Atoll was satisfactory to him (Enclosure 12). On April 28, 1953, the CNO was informed by the AEC (Enclosure 13) that the danger area should comprise the area bounded by the meridians 160°35' - 166°16' east longitude and by the parallels 10°15' - 12°45' north latitude. July 1, 1953 was established in that same letter as the date when the danger area should become effective. On May 22, 1953, the AEC informed the CNO of the fact that Bikini had been officially closed by UN action on April 2, 1953, and requested that the CNO take action to include in the danger area the air space as well as the surface area marked by the boundaries (Enclosure 14).

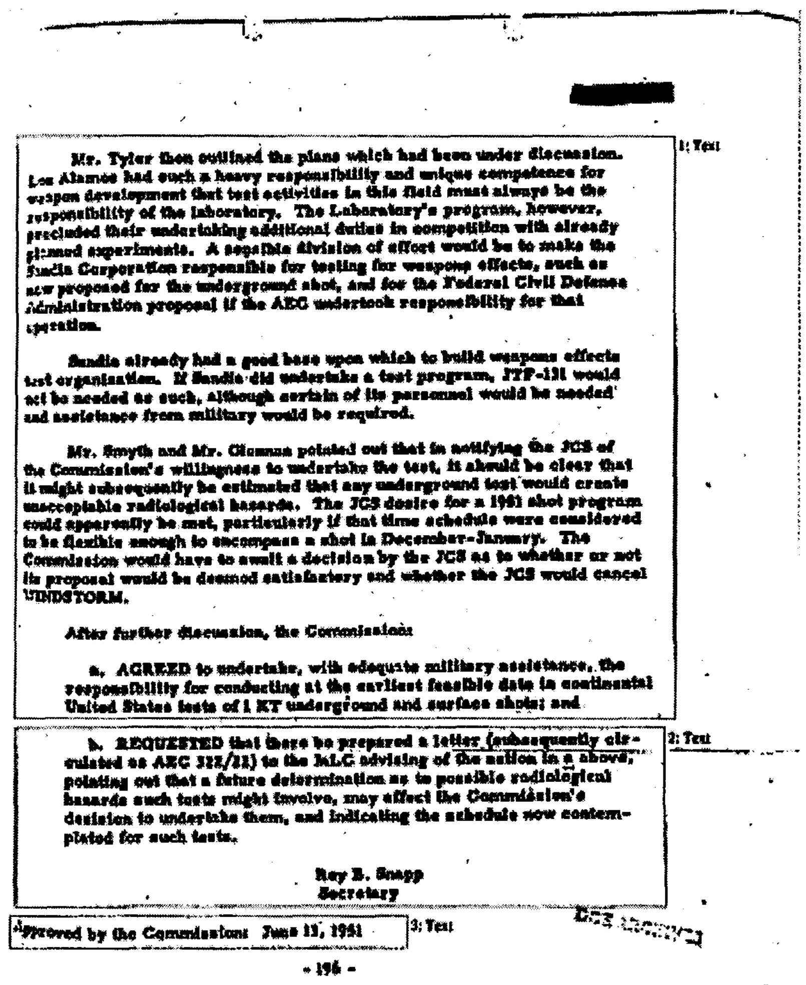
The closed area around Bikini Atoll was established by the State Department notification to the UN on April 2, 1953 (Enclosure 15). The danger area approved at the start of CASTLE was disseminated in the United States Hydrographic Office publication, "Notice to Mariners" No 21, dated May 23, 1953 and was reproduced in corresponding Japanese documents. A similar notice to airmen was issued through CAA channels.

The basic document governing operations of JTF-7 is Operations Order 1-53 dated 20 June 1953. The plan upon which this is based was [illegible] approved by the JCS on 15 April 1953. CJTF-7 furnished the AEC a [illegible]

[redacted]

[illegible]

Examples of DOE Documents



[redacted]

1: Text

Mr. Tyler then outlined the plans which had been under discussion. Los Alamos had such a heavy responsibility and unique competence for weapon development that test activities in this field must always be the responsibility of the laboratory. The Laboratory's program, however, precluded their undertaking additional duties in competition with already planned experiments. A sensible division of effort would be to make the Sandia Corporation responsible for testing for weapons effects, such as now proposed for the underground shot, and for the Federal Civil Defense Administration proposal if the AEC undertook responsibility for that operation.

Sandia already had a good base upon which to build weapons effects test organization. If Sandia did undertake a test program, JTF-132 would not be needed as such, although certain of its personnel would be needed and assistance from military would be required.

Mr. Smyth and Mr. Gleason pointed out that in notifying the JCS of the Commission's willingness to undertake the test, it should be clear that it might subsequently be estimated that any underground test would create unacceptable radiological hazards. The JCS desire for a 1951 shot program could apparently be met, particularly if that time schedule were considered to be flexible enough to encompass a shot in December-January. The Commission would have to await a decision by the JCS as to whether or not its proposal would be deemed satisfactory and whether the JCS would cancel WINDSTORM.

After further discussion, the Commission:

a. AGREED to undertake, with adequate military assistance, the responsibility for conducting at the earliest feasible date in continental United States tests of 1 KT underground and surface shots; and

2: Text

b. REQUESTED that there be prepared a letter (subsequently circulated as AEC 322/22) to the MLC advising of the action in a above, pointing out that a future determination as to possible radiological hazards such tests might involve, may affect the Commission's decision to undertake them, and indicating the schedule now contemplated for such tests.

Roy B. Snapp
Secretary

Approved by the Commission: June 13, 1951

3: Text

- 196 -

[illegible]

2

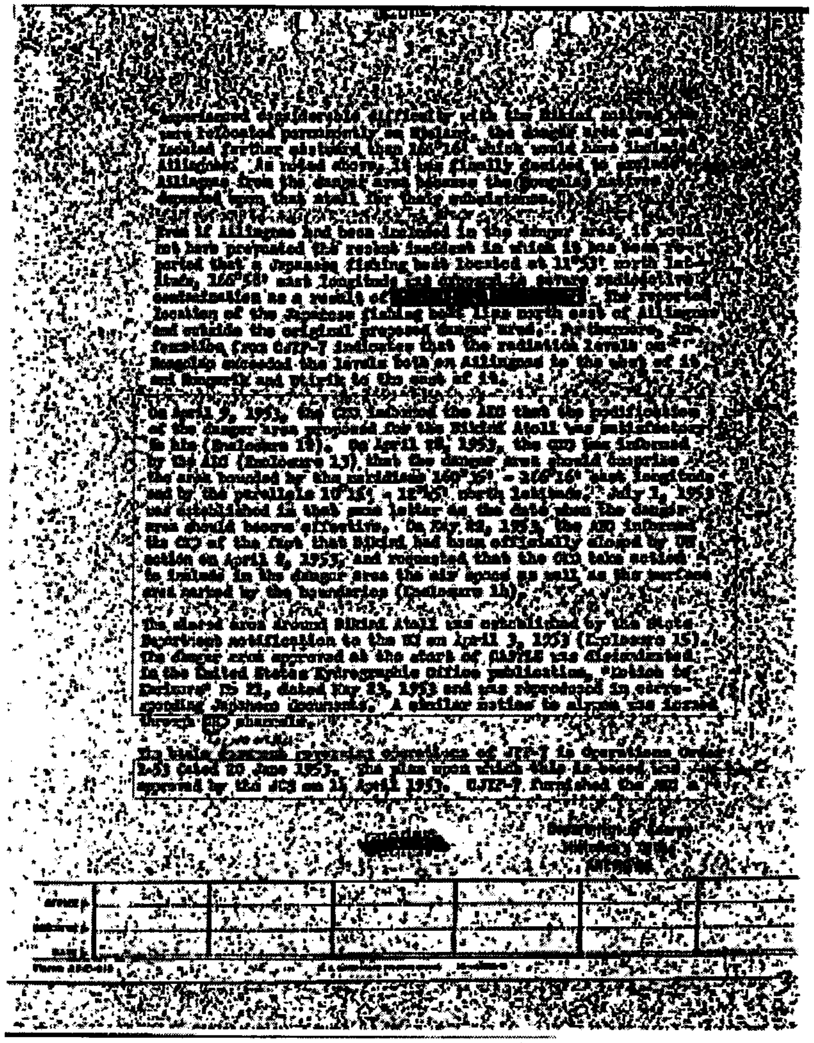
	Caere	ISRI	Nankai	Recognita	Xerox
Error Correction	Accuracy (Marked)	Accuracy (Marked)	Accuracy (Marked)	Accuracy (Marked)	Accuracy (Marked)
None	98.71% (–)	99.03% (–)	98.11% (–)	96.31% (–)	96.91% (–)
Rejects only	98.71% (0.00%)	N/A (–)	98.66% (0.37%)	97.83% (0.69%)	98.02% (0.42%)
Rejects and 1st Level Markers	98.85% (0.32%)	N/A (–)	99.31% (1.52%)	N/A (–)	98.25% (1.52%)
Rejects, 1st and 2nd Level Markers	99.03% (0.55%)	N/A (–)	99.35% (4.48%)	N/A (–)	98.52% (2.22%)

Table 1: Character Accuracy

	Caere	ISRI	Nankai	Recognita	Xerox
Accuracy	97.33%	98.22%	94.36%	91.39%	94.36%

Table 2: Word Accuracy

Examples of DOE Documents



[illegible]

experienced considerable difficulty with the Bikini natives who were relocated permanently on Kwajalein, the danger area was not located further eastward than 166°16' which would have included Ailinginae. As noted above, it was finally decided to exclude Ailinginae from the danger area because the (Rongelap) natives depended upon that atoll for their subsistence.

Even if Ailinginae had been included in the danger area, it would not have prevented the recent incident in which it has been reported that a Japanese fishing boat located at 11°53' north latitude, 166°58' east longitude was exposed to severe radioactive contamination as a result of [redacted]. The reported location of the Japanese fishing boat lies north east of Ailinginae and outside the original proposed danger area. Furthermore, information from CJTF-7 indicates that the radiation levels on Rongelap exceeded the levels both on Ailinginae to the west of it and Rongerik and Utirik to the east of it.

On April 9, 1953, the CNO informed the AEC that the modification of the danger area proposed for the Bikini Atoll was satisfactory to him (Enclosure 12). On April 18, 1953, the CNO was informed by the AEC (Enclosure 13) that the danger area should comprise the area bounded by the meridians 160°35' - 166°16' east longitude and by the parallels 10°15' - 12°45' north latitude. July 1, 1953 was established in that same letter as the date when the danger area should become effective. On May 22, 1953 the AEC informed the CNO of the fact that Bikini had been officially closed by UN action on April 2, 1953, and requested that the CNO take action to include in the danger area the air space as well as the surface area marked by the boundaries (Enclosure 14).

The closed area around Bikini Atoll was established by the State Department notification to the UN on April 3, 1953 (Enclosure 15). The danger area approved at the start of CASTLE was disseminated in the United States Hydrographic Office publication, "Notice to Mariners" No 21, dated May 23, 1953 and was reproduced in corresponding Japanese documents. A similar notice to airmen was issued through CAA channels.

The basic document governing operations of JTF-7 is Operations Order 2-53 dated 20 June 1953. The plan upon which this is based was approved by the JCS on 15 April 1953. CJTF-7 furnished the AEC a [illegible]

[illegible]

Form AEC-318 [illegible]

	Caere	ISRI	Nankai	Recognita	Xerox
Error Correction	Accuracy (Marked)	Accuracy (Marked)	Accuracy (Marked)	Accuracy (Marked)	Accuracy (Marked)
None	27.20% (–)	1.17% (–)	22.29% (–)	6.94% (–)	1.17% (–)
Rejects only	27.20% (0.00%)	N/A (–)	53.70% (6.55%)	75.53% (21.43%)	1.17% (0.00%)
Rejects and 1st Level Markers	69.60% (16.99%)	N/A (–)	67.73% (27.12%)	N/A (–)	1.95% (0.23%)
Rejects, 1st and 2nd Level Markers	74.43% (27.98%)	N/A (–)	76.07% (36.79%)	N/A (–)	5.38% (0.39%)

Table 1: Character Accuracy

	Caere	ISRI	Nankai	Recognita	Xerox
Accuracy	15.58%	0.00%	9.55%	14.57%	0.00%

Table 2: Word Accuracy

DPI Program Initiatives

- **_Automation and Reengineering the Declassification Process_**
- **_Document Conversion & Characterization_**
  - **_Scanned Image Processing, OCR Improvement_**
  - **_Sorting & Categorizing,_**
- **_Classification Knowledge Representation_**
  - **_HyperLinked Classification Guidance Database_**
- **_Classified Information Detection_**
  - **_Text Analysis Project_**
  - **_UltraStructure System_**
- **_Distribution & Access_**
  - **_Opennet Internet Database_**
  - **_Fractal Compression R&D_**

Summary

- Declassification Community Automation Increasing
- Automation Sophistication Increasing
- Law & Societal Trends Will Drive Further Changes
- Document Image Technology Can Help Meet the Communities Needs

Detection of Duplicate Documents and Text Retrieval in Image Domain (D3/TR)

Henry F. Villarama, Harish Kathpal
Kathpal Technologies, Inc.
2230 Gallows Road, Suite 380
Dunn Loring, Virginia 22027
kti@kathpal.com

Abstract

In keeping with Executive Order 12958, the CIA intends to declassify documents numbering in the millions. Currently, the document declassification process is very labor intensive requiring an average of 3 to 10 minutes per page. Any reduction in the processing time will significantly improve productivity in massive declassification operations. This project investigates the availability and effectiveness of technologies in identifying duplicate documents in very large collections of document images and retrieval of text from images of a single document. The ability to be integrated into the CIA's declassification environment will also be analyzed. This research focuses on duplicate database detection and text retrieval in the raster image domain in the absence of optical character recognition. Areas of evaluation are functional performance, interoperability, human and computer interface, and maintainability. The project will involve research of applicable techniques/technologies such as:

- Graphical Zoning
- Page Segmentation
- Automatic Scan and Indexing
- Image Feature Extraction
- N-gram Based Techniques
- Character Shape Coding

This project is currently on-going under the direction of the Federal Intelligence Document Understanding Laboratory (FIDUL). Project scope is limited to research of availability and effectiveness of applicable technologies in the area of concern. Both the duplicate document detection tool and text retrieval tool will be considered for integration into the CIA Declassification Factory at project completion.

A System For Table Understanding

Charles Peterman Dr. C.Hwa Chang**
EECS Department, Tufts University, Medford, MA 02155

*Hassan Alam***
BCL Computers, 2540 Mission College Blvd, Santa Clara, CA 95054

**{peterman,hchang}@eecs.tufts.edu **hassana@netcom.com*

Abstract

The tabular format is the most compact way to display data with multiple indices in an image. This fact, combined with the prevalence of faxes and phone lines as a means of data transfer throughout the world, indicates the need for a flexible method of interpretation of these images and the transfer of that information to a database. The flexibility of this method is gained by using a loosely constrained model for the table as a whole, while using information derived from the image to constrain the models for each of five types of text blocks (data, vertical indices, horizontal indices, title, and footnotes) and structure (rows, columns, hierarchy of descriptive blocks) that might be found within the table. The parameters for the models are based on the white space and syntactic relationships present in the table. Syntactic relationships are built using a variant of the edit distance algorithm proposed by Horst Bunke and by using direct string matching. The boundaries between entities based on these structures allow for the detection of complex structures within the table, such as the presence of multi-lined rows, and changes in the column and row structures.

1.0 Introduction

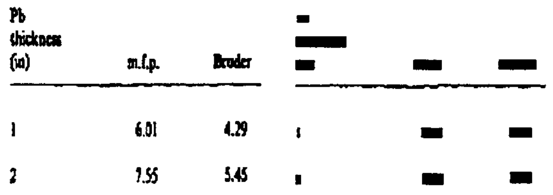
The value of information depends upon how quickly it can be disseminated to the people who need it. Printed tables represent both a wealth of information and a transmission bottleneck. As long as the data in the table remains only on paper, it is slow to transfer and correlate with other information. The process of manual entry of the data from a table to a database is time consuming and prone to human error. An automated system of table recognition and understanding would mitigate both of these concerns. This paper focuses on table understanding, recognition being a problem worked on by other members of the BCL group.

Table understanding is the process by which the text information in the table is extracted into a searchable form that preserves the text relations inferred by the layout and syntax of the printed table. This process consists of determining how the blocks of text associate with each other. There are two general types of blocks: data blocks and descriptive blocks. Data blocks are the most numerous and regular in terms of content and spacing relationships within a given table. Descriptive blocks provide context to either kind of block. Descriptive blocks that have the same kind of associativity with data blocks (top down, left to right) tend to share a common boundary with the data they describe. These boundaries may be defined by changes in any of the following characteristics from one region to another:

- Horizontal or vertical white space
- Size and/or type of font
- Justification
- Presence and characteristics of lines
- Content
- Syntax of content

The information for defining a boundary between regions can be generated from the table itself based on the regularity of the features above along one cardinal axis and their irregularity along the other.

It is unnecessary and cumbersome to work at the image level for this type of analysis. For the purposes of this paper, it is assumed that the table has been recognized and extracted from the rest of the document. Rather than working with pixels as the smallest unit, it is preferable to work at the level of a page description language. At this level, the image has been segmented into lines and blocks of text, and the text has passed through an OCR. Within this paper, the term "block" of text refers to a data structure that stores the location, content, tag number, and neighbor relationships for a string of text. Location is stored as the pixel coordinates of the upper left and lower right of the block. The tag number is an arbitrary number which uniquely identifies that particular block. The neighbor relationships indicate which other blocks of text are closest to this block in the cardinal directions (north south, east, and west) by tag number. Vertical and horizontal lines are also tagged with a unique identifying number, and are stored by their upper left and lower right coordinates along with the neighboring blocks.



Pb thickness (in)	m.f.p.	Broder
1	6.01	4.29
2	7.55	5.45

Figure 1: The table image (left) and the block equivalent

The equivalent block file would be similar to this:
block(1, $[x_1,y_1]$, $[x_2,y_2]$, "Pb"{north: nil south: 2 east: nil west: nil}).
block(2, $[x_1,y_1]$, $[x_2,y_2]$, "thickness"{north: 1 south: 6 east: nil west: nil}).
block(7, $[x_1,y_1]$, $[x_2,y_2]$, "m.f.p"{north: nil south: 10 east: 5 west: 6}).
block(5, $[x_1,y_1]$, $[x_2,y_2]$, "Broder"{north: nil south: 3 east: nil west: 7}).
block(6, $[x_1,y_1]$, $[x_2,y_2]$, "(in)"{north: 2 south: 9 east: 7 west: nil}).
block(9, $[x_1,y_1]$, $[x_2,y_2]$, "1"{north: 6 south: 12 east: 10 west: nil}).
block(12, $[x_1,y_1]$, $[x_2,y_2]$, "2"{north: 9 south: nil east: 11 west: nil}).
block(10, $[x_1,y_1]$, $[x_2,y_2]$, "6.01"{north: 7 south: 11 east: 3 west: 9}).
block(11, $[x_1,y_1]$, $[x_2,y_2]$, "7.55"{north: 10 south: nil east: 4 west: 12}).
block(3, $[x_1,y_1]$, $[x_2,y_2]$, "4.29"{north: 5 south: 4 east: nil west: 10}).
block(4, $[x_1,y_1]$, $[x_2,y_2]$, "5.45"{north: 3 south: nil east: nil west: 11}).
line(101, $[x_1,y_1]$, $[x_2,y_2]$, {north: 6, 7, 5 south: 9, 10, 3 east: nil west: nil}).

1.1 Application

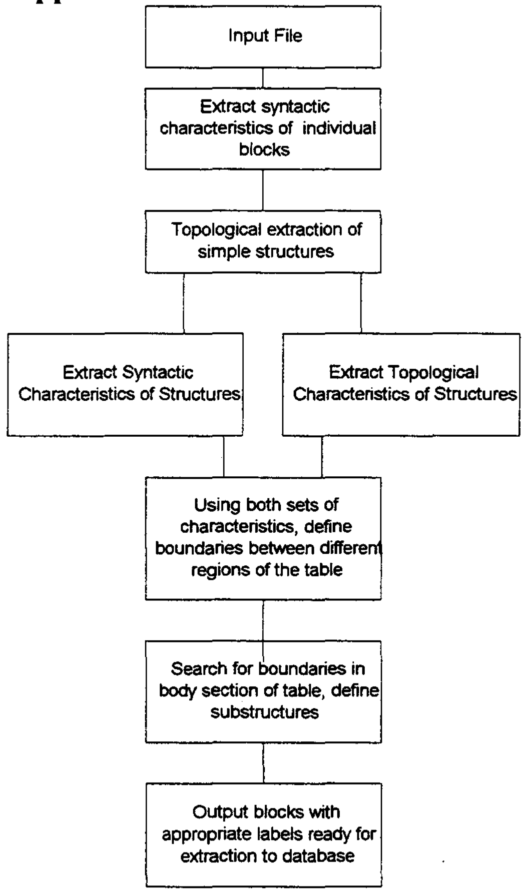

Input File
Extract syntactic characteristics of individual blocks
Topological extraction of simple structures
Extract Syntactic Characteristics of Structures
Extract Topological Characteristics of Structures
Using both sets of characteristics, define boundaries between different regions of the table
Search for boundaries in body section of table, define substructures
Output blocks with appropriate labels ready for extraction to database

Figure 2: Flow Chart of this Table Understanding System

Throughout this paper there are two input assumptions to keep in mind:

1. The input file contains the complete table; there are no missing or extraneous features.
2. The text within the blocks has minimal OCR errors (under 1%, preferably).

It is important to distinguish table understanding from form understanding. Tables represent a mapping of one or more indices to many pieces of data. The data type is usually determined by one of the indices, and is therefore regular along one of the axis. Forms represent only a one to one mapping between indices and data, in which there are no implications of regularity of data. By this fact, forms are ill suited to the methods developed in this paper.

2.0 Regions and Elements of the Generalized Table Model

The regions of the model are defined by how their content associates with itself and with the content of other regions. The generic model for a table considered for analysis in this paper in Figure 2. Viewing the table as a compact layout of a database, the fields of the database are the indices of the table. The data that the index describes is found either under the index or to its right. It is assumed that the data is either vertically (column) or horizontally (row) aligned with the index by which it is described.

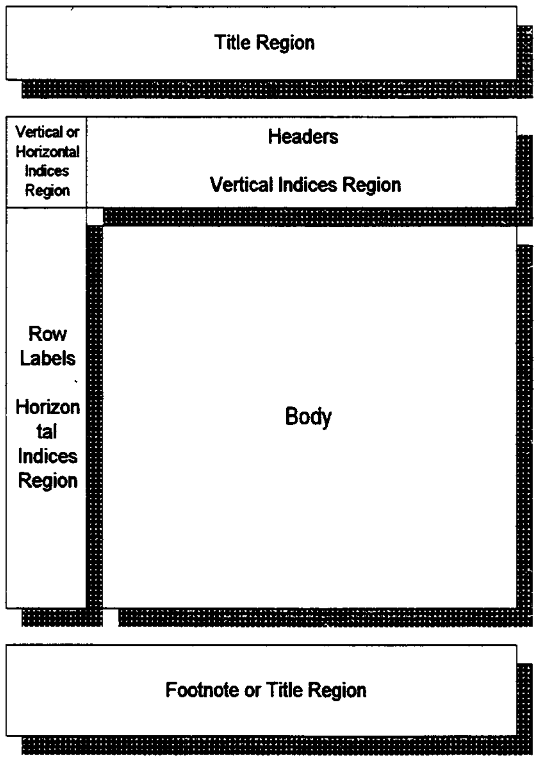


Figure 3: Generic Table Model

All regions in Figure 3 are optional except for the Body. The Body region consists, at its most simple level, of data completely without explicit indices. Understanding of tables that consist only of data without indices requires domain specific knowledge that is usually highly specialized, and is therefore inappropriate for consideration in this model. Such cases are rare. The majority of cases considered have the vertical indices, horizontal indices, and title regions defined as well. The composition of the body region is not limited to simple data cells consisting of one text block. The body region may contain complex data cells consisting of more than one text block, or complex structures such as sub-tables. Few assumptions are made about the contents of the regions.

While the model for the overall table is loosely constrained, the behaviors of the elements of the table are well defined. The elements include structures, regions, and block descriptions. The primary structures are flat rows and columns. A flat row is a collection of blocks that have reciprocal neighbor relations along the horizontal axis. Likewise for a column, except that reciprocal neighbor relations are along the vertical axis. Compound structures, such as combined rows and sub-tables are combinations of simple structures. Combined rows are groupings of flat rows such that the primary index for one of the flat rows in that group is descriptive of all data blocks in those flat rows. Sub-tables are regions of the body in which there exists at least one set of indices and data blocks.

Regions are defined by the way the blocks they include interact with each other and with blocks in other regions. The title region is associated with every other block within the table. Footnotes bind to another specific block, which may either be data or descriptive. Vertical indices are associated with columns while horizontal indices are associated with rows. The body contains the data described by the descriptive regions and may contain sub-regions that function as vertical or horizontal indices.

Within the descriptive regions, blocks are classified as either being descriptive of other blocks within the region or outside of the region. Data blocks are associated with the descriptive blocks of the column, row, and other structures that contain them.

3.0 Topological Models

The topology of a table is the primary source of information about the individual blocks and the significance of their contents. The coarse information provided at the topological level may be sufficient to successfully classify all blocks within a simple table. From the input data the simple structures (rows and columns) are extracted along with their topological characteristics: size, position, the means of spacing between blocks, mean vertical size of blocks, deviations from those means, and justification trends. A figure of significance, α, is attached to each deviation found. The value α is a function of the number of times analogous deviations occur within the column or row versus the total number of possible occurrence for that deviation. The horizontal and vertical lines are tallied independently with considerations for run length, thickness, and number of blocks intervening.

The larger the total number of lines of a particular grouping of characteristics, the lower the significance, α, for those lines.

3.1 Boundaries between Regions

Once the topological data has been generated, it may be parsed to find the boundaries between the different regions: title, header, row label, body, and footnote. (See figure 1.) The following characteristics are used to define such boundaries.

3.1.1 Boundary between Title and Header Regions

- All blocks north of boundary have the same justification and no horizontal neighbors.
- Blocks to the south of the boundary may have horizontal neighbors.
- A horizontal line of α greater than a threshold value may coincide with the boundary.
- The vertical space between the blocks north of the border and south of the border may be smaller than the vertical space between the regions.

3.1.2 Boundary between Headers and Body/Row Labels Regions

- A horizontal line of α greater than a threshold value may coincide with the boundary.
- The vertical space between blocks along the boundary might not coincide with the means calculated for the respective columns of those blocks.
- The boundary may be represented by a change in the justification of the blocks, relative to the column.
- The blocks below the boundary may be uniform in sizing, while those above are irregular.

3.1.3 Boundary between Row Labels and Body/Headers Regions

A baseline assumption is that the leftmost column is the horizontal indices region of the table. Supporting evidence can be found in the following details:

- Irregular horizontal spacing between the rightmost boundary of the blocks to the left of the boundary and the justification defined boundary of the blocks to the right of the boundary.
- Presence of a vertical line of α greater than a threshold value coinciding with the boundary.
- The blocks right of the boundary may be uniform in sizing, while those above are irregular.

3.1.4 Boundary between Row Labels/Body and Footnote/Title Regions

- Presence of a horizontal line of α greater than a threshold value coinciding with the boundary.
- Blocks below boundary have no horizontal neighbors.
- Blocks below the boundary are left or center justified.
- Vertical spacing of boundary is not the same as the means of vertical spacing either above or below it.

3.2 Read Order within Regions

Once the boundaries between regions have been defined, the read orders of the blocks within those regions are determined. The regions themselves are partially defined by which set of topological rules determine their read order. Read orders are assumed to proceed either left to right, top to bottom, or a combination of the two. Non-adjacent blocks may be sequential in read order, as occurs within the indices. Adjacent blocks which may be grouped together to form one atomic descriptive entity (header, row label, title, footnote) are called 'cells.' The term 'super' is used to describe any descriptive block within the indices regions that describes more than one other index within the same region.

The title region has the simplest read order, which proceeds from top to bottom within the region. The index regions, body, and footnotes have more complex rules governing their read order.

3.2.1 Header Read Order

Headers are of two types: plain and super. The description 'super' is applied to any block whose horizontal span covers two or more columns. A super header is a member of each of the columns that it spans (Figure 3). This span may be real, defined by the horizontal length of the block, or implied. A horizontal span greater than the block length can be implied by a line of run length shorter than the width of the table, or by centering one block above the region between the two blocks below it (Figure 4). 'Plain' headers occur within the horizontal bounds of the column of data they describe, and have no visual references to imply membership to any other column. The header text string for an individual column may be obtained by concatenating the header blocks for the column in order from top to bottom.

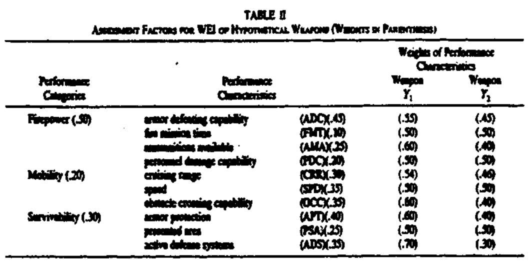


TABLE II
ASSESSMENT FACTORS FOR WEI OF HYPOTHETICAL WEAPONS (WEIGHTS IN PARENTHESES)

Performance Categories	Performance Characteristics		Weights of Performance Characteristics	
			Weapon Y_1	Weapon Y_2
Firepower (.50)	armor defeating capability	(ADC)(.45)	(.55)	(.45)
	fire mission time	(FMT)(.10)	(.50)	(.50)
	ammunitions available	(AMA)(.25)	(.60)	(.40)
	personnel damage capability	(PDC)(.20)	(.50)	(.50)
Mobility (.20)	cruising range	(CRR)(.30)	(.54)	(.46)
	speed	(SPD)(.35)	(.50)	(.50)
	obstacle crossing capability	(OCC)(.35)	(.60)	(.40)
Survivability (.30)	armor protection	(APT)(.40)	(.60)	(.40)
	presented area	(PSA)(.25)	(.50)	(.50)
	active defense systems	(ADS)(.35)	(.70)	(.30)

Figure 3: Super Headers in Rightmost Columns based on Actual Horizontal Span of Block

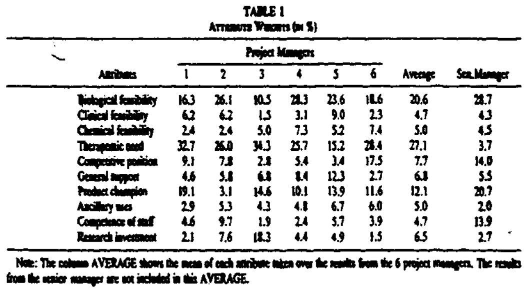


TABLE 1
ATTRIBUTE WEIGHTS (IN %)

	Project Managers							
Attributes	1	2	3	4	5	6	Average	Sen. Manager
Biological feasibility	16.3	26.1	10.5	28.3	23.6	18.6	20.6	28.7
Clinical feasibility	6.2	6.2	1.5	3.1	9.0	2.3	4.7	4.3
Chemical feasibility	2.4	2.4	5.0	7.3	5.2	7.4	5.0	4.5
Therapeutic need	32.7	26.0	34.3	25.7	15.2	28.4	27.1	3.7
Competitive position	9.1	7.8	2.8	5.4	3.4	17.5	7.7	14.0
General support	4.6	5.8	6.8	8.4	12.3	2.7	6.8	5.5
Product champion	19.1	3.1	14.6	10.1	13.9	11.6	12.1	20.7
Ancillary uses	2.9	5.3	4.3	4.8	6.7	6.0	5.0	2.0
Competence of staff	4.6	9.7	1.9	2.4	5.7	3.9	4.7	13.9
Research investment	2.1	7.6	18.3	4.4	4.9	1.5	6.5	2.7

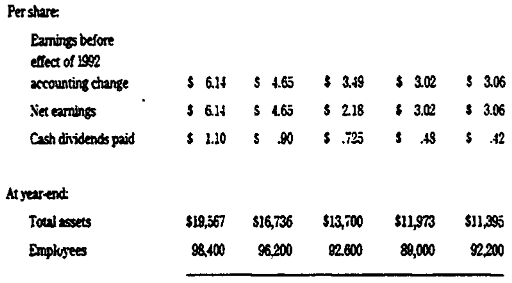
Note: The column AVERAGE shows the mean of each attribute taken over the results from the 6 project managers. The results from the senior manager are not included in this AVERAGE.

Figure 4: Super Header 'Project Managers' with Horizontal Span Implied by Underlying Line

3.2.2 Row Label Read Order

A row label may either be directly or indirectly associated with members of the body region. If a member of the row label region is adjacent to a data block, it is assumed to be directly associated with that data block unless syntactic evidence indicates otherwise. If the block is directly associated, it is said to be anchored, and the read order is from left to right. Without an adjacent data block, a member of the row label region is assumed to be associated with at least one anchored block and possible other unanchored blocks. The formation of these associations of blocks defines the problem of read order in this region.

Indentation is the most common way to discern these associations. A 'super' row label will often have its subordinates indented from it. Association of the super row label to subordinates applies only to blocks that are indented from that super row label that occur directly below the super row label or one of its subordinates. Figure 5 provides a good example of this.



Per share:					
Earnings before effect of 1992 accounting change	$ 6.14	$ 4.65	$ 3.49	$ 3.02	$ 3.06
Net earnings	$ 6.14	$ 4.65	$ 2.18	$ 3.02	$ 3.06
Cash dividends paid	$ 1.10	$ .90	$ .725	$ .48	$ .42
At year-end:					
Total assets	$19,567	$16,736	$13,700	$11,973	$11,395
Employees	98,400	96,200	92.600	89,000	92,200

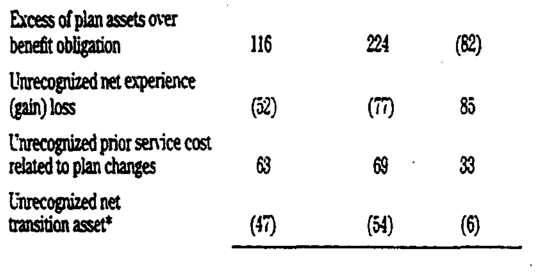
Figure 5: Indentation used to Denote Subordinate Blocks

As stated in Section 3.2, often blocks may be associated with their neighbors to form atomic units. Most commonly, this occurs among row labels when the text string is too long to fit into the column and the label is divided between two or more lines of text. Usually, only one of these lines of text will be anchored. There are two possible cases, that the block is associated with the one above it or the one below it. If associated with the one above it, it is considered a 'continuation' of that row label. If the unanchored block is associated with the one below it, the block is considered to be the '1st part' of that row label.

3.2.2.1 Association of an Unanchored Row Label with its Southern Neighbor

This association is supported by the following topological clues:

- The block above the unanchored block is relatively indented.
- The block is closer to its southern neighbor than it is to its northern neighbor (Figure 6).
- The southern neighbor of the unanchored block is relatively indented, while its southern neighbor is not indented from the unanchored block.



Excess of plan assets over benefit obligation	116	224	(62)
Unrecognized net experience (gain) loss	(52)	(77)	85
Unrecognized prior service cost related to plan changes	63	69	33
Unrecognized net transition asset*	(47)	(54)	(6)

Figure 6: Unanchored Row Labels Associated with Southern Neighbors

3.2.2.2 Association of an Unanchored Row Label with its Northern Neighbor

The label 'continuation' is applied to a block when it is associated with its northern neighbor. This association is supported by these topological clues:

- The unanchored block is closer to its northern neighbor than it is to its southern neighbor.
- It is indented from its northern and southern neighbors.

3.2.3 Defining Sub-regions within the Body

Within the body region, data blocks are assumed to be associatively related to their neighbors unless there exists a sparsely reused or singular boundary condition between the data blocks. Again, other boundary conditions may exist which are based on changes in the syntax of the table.

3.2.3.1 Topological Boundaries within the Body Region

- Horizontal or vertical line of significance greater than a threshold for such objects within the body.
- Vertical space gaps that are horizontally aligned in all columns and are not consistent with the means of vertical space between blocks for the columns.
- Existence of a block that spans multiple columns.

4.0 Syntactic Models

The text of the table provides a wealth of information concerning the hierarchy of blocks in the regions and the boundaries between regions. Word matching between the regions and sub-regions of a table and extraction of syntactic patterns from the table are tools which can be employed to extract and exploit this information. Recent advances in optical character recognition reduce the possibility of miss-recognized characters to levels that are tolerable for this kind of analysis.

4.1 Syntactic Patterns

The data within a table tend to use syntax's that are consistent within the columns or rows. In order to find this syntax, a simplification of the text within each block must occur. This method replaces each individual character with a special character representative of the grouping to which it belongs. Characters not grouped into families are not replaced or omitted. The resulting replacement string is called the reduced regular expression, RRE for short.

Character Family	Replacement Character
[0-9]	η
[A-Z]	β
[a-z]	γ

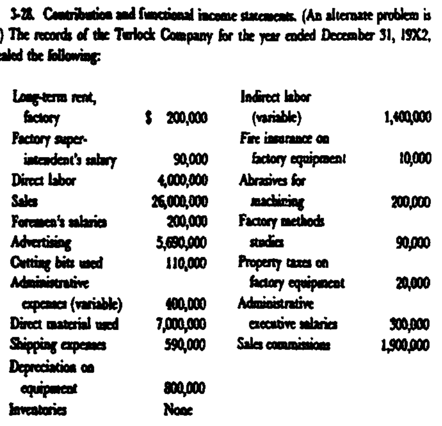
Table 1: Replacement Characters

Original String	Replacement String
Juan	βγ[3]
123.8	η[3].η
Real (20.1 %)	βγ[3] (η[2].η %)

Table 2: Original String versus Replacement String (RRE)

Comparison between strings can now be made that are sensitive to changes in context using variants of the edit distance method developed by Horst Bunke[2]. The most recurring expression, and the types and frequency of variations from that expression, are recorded for each column and row. Neighboring blocks that are within a threshold edit distance of each other are clustered together. The thresholds are based upon consistency of RRE's within the structure. Multiple thresholds may be used to form various strengths of clusters. A syntactic boundary is now defined as the edge of any of these clusters. These boundaries between different syntactic entities can be used to find the boundaries between regions of the table as well as find boundaries between sub-regions within the body region.

If a column is found to consist uniformly of multiple different RRE's, their ordering from top to bottom is scanned for repetition of patterns of RRE's. If a pattern is found, a substructure within the body region of the table may be defined as one set of such a pattern. If a pattern is not found, and the RRE's consists mostly of non-numeric character, then that column is a horizontal indices region candidate. Indentation, RRE content of other columns, and word matching between columns, are the verifying criteria for the candidate. Figure 7 illustrates two tables that are detectable by this method.



3-28. Contribution and functional income statements. (An alternate problem is 3-2.) The records of the Turlock Company for the year ended December 31, 19X2, revealed the following:

Long-term rent, factory	$ 200,000	Indirect labor (variable)	1,400,000
Factory superintendent's salary	90,000	Fire insurance on factory equipment	10,000
Direct labor	4,000,000	Abrasives for machining	200,000
Sales	26,000,000	Factory methods studies	90,000
Foremen's salaries	200,000	Property taxes on factory equipment	20,000
Advertising	5,690,000	Administrative executive salaries	300,000
Cutting bits used	110,000	Sales commissions	1,900,000
Administrative expenses (variable)	400,000		
Direct material used	7,000,000		
Shipping expenses	590,000		
Depreciation on equipment	800,000		
Inventories	None		

Figure 7: Two tables Side by Side

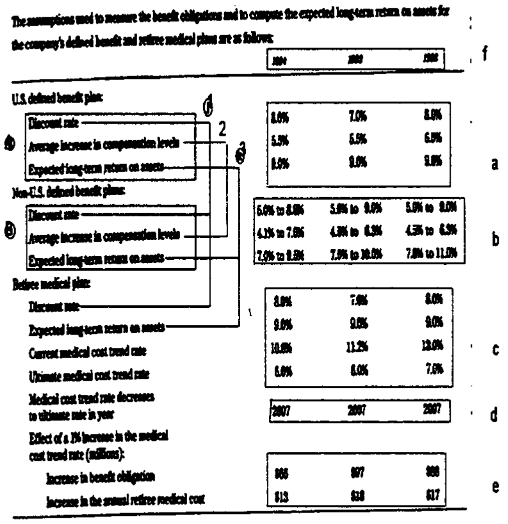
4.2 Word Matching

Direct word matching is also potent tool for table understanding. Quite often two tables will be located side by side. This is especially true if the two tables contain information that is strongly related. At the image level, it is difficult to separate these two entities. At the text level, the second table can be detected by finding recurrence of terms from the horizontal indices region of one table in the column that serves the same role in the other. This can also be accomplished by dividing the vertical indices region in half and comparing the two halves against each other for term recurrence. In order to increase the accuracy of this technique, a stop list of common words is used to filter the words used in this comparison. By using the table itself to generate its own key words, the cost of building and using a lexicon is avoided.

4.3 Sentence Detection

Quite often, there are multiple blocks arranged vertically that should be treated as a cell. These regions, called flow text, should be identified and grouped to prevent mistaken parsing. Usually, these regions form complete sentences or chains of sentences. Using the syntax of a complete sentence, i.e. starting with a capital letter and ending with a punctuation, allows us to cluster these blocks together and treat them as one logical unit.

4.4 Example of the Combined Application of these Techniques



The assumptions used to measure the benefit obligations and to compute the expected long-term return on assets for the company's defined benefit and retiree medical plans are as follows:

	[illegible]	[illegible]	[illegible]
U.S. defined benefit plan:			
Discount rate	8.0%	7.0%	8.0%
Average increase in compensation levels	5.5%	5.5%	6.0%
Expected long-term return on assets	9.0%	9.0%	9.0%
Non-U.S. defined benefit plans:			
Discount rate	6.0% to 8.5%	5.0% to 9.0%	5.0% to 9.0%
Average increase in compensation levels	4.1% to 7.0%	4.0% to 6.3%	4.5% to 6.5%
Expected long-term return on assets	7.0% to 9.5%	7.0% to 10.0%	7.0% to 11.0%
Retiree medical plan:			
Discount rate	8.0%	7.0%	8.0%
Expected long-term return on assets	9.0%	9.0%	9.0%
Current medical cost trend rate	10.0%	11.2%	12.0%
Ultimate medical cost trend rate	6.0%	6.0%	7.0%
Medical cost trend rate decreases to ultimate rate in year	2007	2007	2007
Effect of a 1% increase in the medical cost trend rate (millions):			
Increase in benefit obligation	$88	$97	$98
Increase in the annual retiree medical cost	$13	$18	$17

Figure 8: Indentation and Word Repetition used to Denote Hierarchy of Horizontal Indices

Figure 8 shows how the methods of this paper can be combined. Note that all groups of blocks denoted with lower case are grouping according to RRE. These groupings are also supported by white space. Because of the low number of horizontal lines, the significance of each one is high. The line, combined with the vertical gap coincidental with the topmost line and the change in regular expression, suggests that block f is the header region. Indentation and word repetition denote the hierarchy of horizontal indices. Groups A and B can be defined as sets of strings [1,2,3]. Note that group A is indented from the string "US defined benefit plan:", and therefore subordinate to it.

5.0 Conclusions and Future Work

The current system is realized in PERL running on a Sun Ultra under the Solaris OS. Perl was chosen for its dynamic structure allocation, speed of implementation, and portability. The code for the demonstration version is running under PERL for NT and will also be made available in C++.

The January,1997 test was performed on a corpus of 100 tables. Table style varied within the confines of the general model and included all types of elements discussed in this paper. The results are as follows:

	Total	Missed	Added
Body	3940	13	0
Headers	554	2	29
Row Lbls	1229	15	0
Titles	126	1	2
Footnotes	50	0	0

Table 3: Block Classification

As the results indicate, the header region was most often made to large and encompassed blocks that were not headers, or the headers were included with the title.

	Total	Missed	Added
1st part	58	1	0
Cont.	33	0	0
Anchored	978	10	0
Super	67	4	0

Table 4: Row Label Classification

Missed row labels were due to faulty assignment of the boundary between row label and header regions.

Occured	10
Detected	10

Table 5: Side by Side Tables Detection

The system described within this paper takes advantage of white space and topological techniques and enhances them through the use syntactic knowledge previously unexploited for this problem. By combining exploiting both spheres of knowledge, this system gains sensitivity to complex structures within the table that are not detectable by topological methods. Future improvements include:

- Further refinements to the string comparison algorithms for the detection of substructures.
- Addition of information to the input block file such as font type and style.
- Exploitation of new OCR features to capture superscripts and subscripts so as to preserve footnote relations.

References

[1] Akindele,O., and Belaid, A., "Construction of Generic Models of Document Structures using Inference Tree Grammars" CRIN-CNRS/ INRIA LORRAINE, Batiment LORIA, Campus Scientifque, B.P. 239, 54506 Vandoeuvre-les-Nancy Cedex. France.

[2] Bunke, Horst, "Edit Distance between Regular Languages", Department of Computer Science, University of Bern, Neubruckstr. Bern Switzerland. (bunke@iam.unibe.ch)

[3] Fan, Kuo-Chin, Lu, Jeng-Ming, and Wang, Jing-Yuh,"A Feature Point Clustering Approach to the Segmentation of Form Documents" Institute of Computer Science and Information Engineering, National Central University, Chung-Li, Taiwan, R.O.C.

[4] Farrow, G., Oakley, J., and Xydeas,C.,"Model Matching in Intelligent Document Understanding", *in Proc of the Third ICDAR* Vol I, pp.293-296, August 1995, Montreal Canada.

[5] Green, E.,and Krishnamoorthy,M., "Model Based Analysis of Printed Tables", *in Proc of the Third ICDAR* Vol I, pp.214-217, August 1995, Montreal Canada.

[6] Green, E.,and Krishnamoorthy,M., "Recognition of Tables Using Table Grammars", Department of Computer Science, Rensselaer Polytechnic Institute, Troy, NY.

[7] Ozaki, Masaharo, "Column and White Space Pattern Matching", *in Proc of the Third ICDAR* Vol I, pp.134-137, August 1995, Montreal Canada

BDOC - A Document Representation Method

Annie Dong Scott Tupaj* Dr. C.Hwa Chang**
EECS Department, Tufts University, Medford, MA 02155

*Hassan Alam***
BCL Computers, 2540 Mission College Blvd, Santa Clara, CA 95054

**{adong,stupaj,hchang}@eecs.tufts.edu   **hassana@netcom.com*

Abstract

While more documents are being published on-line, the use of paper based documents is still growing. With proliferation of computer printers and computer based faxes, the paper-less office remains an elusive goal. To incorporate the paper relatively seamlessly into electronic transmission medium, there needs to be a method for capturing the contents of document images and inferring their logical structure and label. This logical labeling and structure inference will help in the ultimate goal converting paper automatically to HTML. BCL along with a number of research groups are attempting to develop algorithms to automate this logical labeling. Algorithms contain deterministic and stochastic pattern recognition techniques and are under constant improvement and revision. To effectively modify algorithms to infer logical labels and structures, an intermediate level of document representation is needed. Ideally this intermediate representation should capture information that humans use to infer logical labels and structure so that algorithms can be rapidly written, tested. This rapid prototyping will assist in testing out hypothesis on document structure inference rapidly. This paper presents BCL's BDOC an intermediate representation of a document that capture much of the information used in structural inference. Built with an object oriented representation, BDOC allow easy access to information about a document's format and content, allow for development and test of different document structural hypothesis.

1 Introduction

Documents contain knowledge. Precisely, they are mediums for transferring knowledge. In fact, much knowledge is acquired from documents such as technical reports, government files, newspapers, books, journals, magazines, letters, bank checks, to name a few. Automatic knowledge acquisition from documents has become an important subject, and many people are working to find new techniques of processing documents. It makes sense to develop systems which can acquire knowledge directly by analyzing and understanding the documents [1]. In order to do that, the structures of a document must be defined first. There exist a variety of definitions of document structures. [2, 3, 4, 5]

As the document is a medium of knowledge, it can be considered not only as a two-dimensional image - a concrete document, but also a conceptual document which corresponds to human thinking. The abstract representation of these two kinds of documents are *conceptual structure* and *concrete structure*. The process of publishing or writing corresponds to encoding the conceptual structure into a concrete structure. Conversely, the concrete structure of the document is decoded into its conceptual one in document processing, i.e. Three main stages are composed in document processing, they are:

(1) Document analysis:
Concrete Document $\xrightarrow{\textit{extracting}}$
Geometric Structure

(2) Document understanding:
Geometric Structure $\xrightarrow{\textit{mapping}}$
Logical Structure

(3) Document decoding:
Logical Structure $\xrightarrow{\textit{decoding}}$
Conceptual Structure

Since 1960s, a great deal of research on the first stage in document processing has been focused on Optical Character Recognition (OCR) . [6, 7] And a lot of vendors such as XEROX and Calera have claimed that they get about 99% percent accuracy.

More efforts are put on the second and third stage now to build a complete system that can work on the three main stages of document processing. As far as all the methods that are concerned, they either employ the geometric information from the image or base on the semantic analysis of the document. But either way must start from the concrete image structure. So if we can develop a mechanism that can store both the geometric and textual information of the document and allow fast and easy access, we can exact as much information as we can by working on the image level only one or two pass, and let the later analysis base on this mechanism and therefore more fast and effective. For this purpose, we develop our BDOC. This paper presents an overview of BDOC specification and reference method. Section 2 describes the algorithms that are related to features extraction for BDOC file storage. Section 3 describes the BDOC specification, Section 4 give some example of BDOC usage and how it benefits later logical and conceptual analysis. Section 5 describes the conclusion and areas of future work.

2 Feature Extraction

When there comes a document, first we scan it in and save it as a TIFF format; second, we use the line extraction and removal module to extract and remove those lines in the image; The third step, is to employ the RLSA algorithm to get a smear image; The fourth step is to use the bounding box algorithm to get the text cells; The fifth step is to post-process the text cells to handle overlapping cells and to merge cells of text into larger units; Then we pass the original image and the text cells coordinates to a OCR engine to get the text information for each text cells.

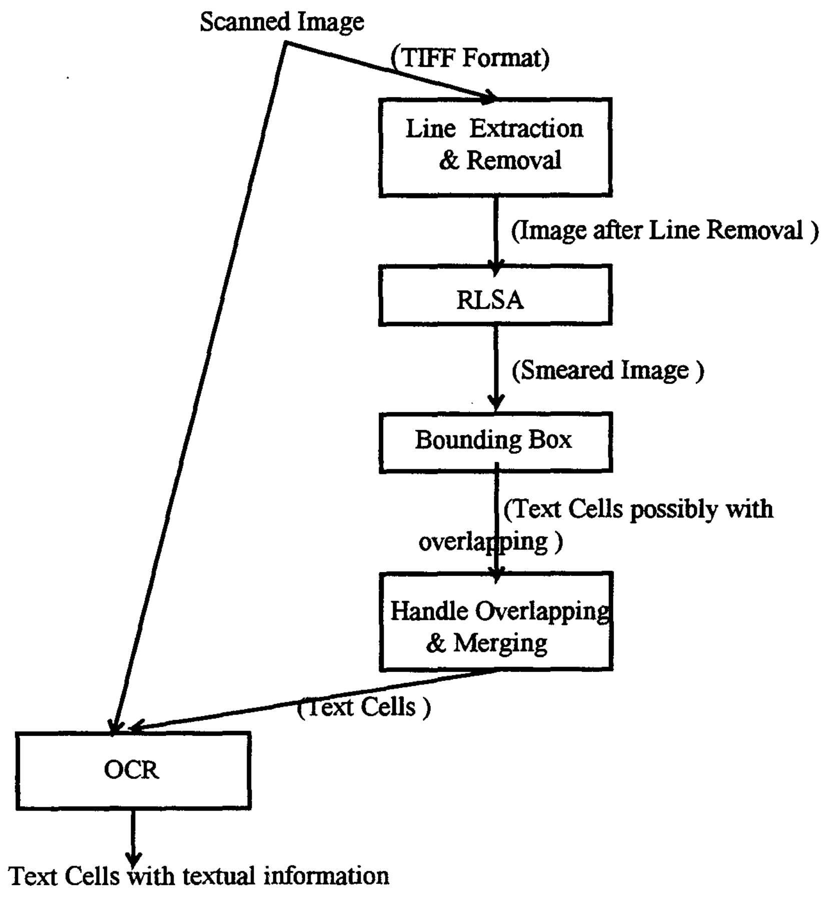

Scanned Image
(TIFF Format)
Line Extraction & Removal
(Image after Line Removal)
RLSA
(Smeared Image)
Bounding Box
(Text Cells possibly with overlapping)
Handle Overlapping & Merging
(Text Cells)
OCR
Text Cells with textual information

Figure 1. Diagram of the File Structure

2.1 Line Extraction and Removal

Technical and financial documents usually have some embedded tables. And many tables have horizontal and vertical lines as defining features. Although we cannot rely on lines to analyze table structure, if present, they can give us valuable clues. Also, line attributes (e.g., thickness, double/single, dashed) can also help in the table analysis. Hence, accurate detection of lines is important in any document processing system. Besides merely recording line location, it is also important to be able to remove lines. This is needed for more accurate text extraction and for white space analysis.

2.2 Run-Length Smoothing (RLSA)

When viewing the image globally on the first pass, we may only be interested in primary features and not fine details. To accomplish this, we smear the image using a run-length smoothing algorithm(RLSA)[8]. For each scan line, if a run white pixels is encountered that is less than a particular threshold, these white pixels are changed to black. This can also be applied to vertical scan lines. The magnitude of the threshold value determines how much smearing takes place in the document image.

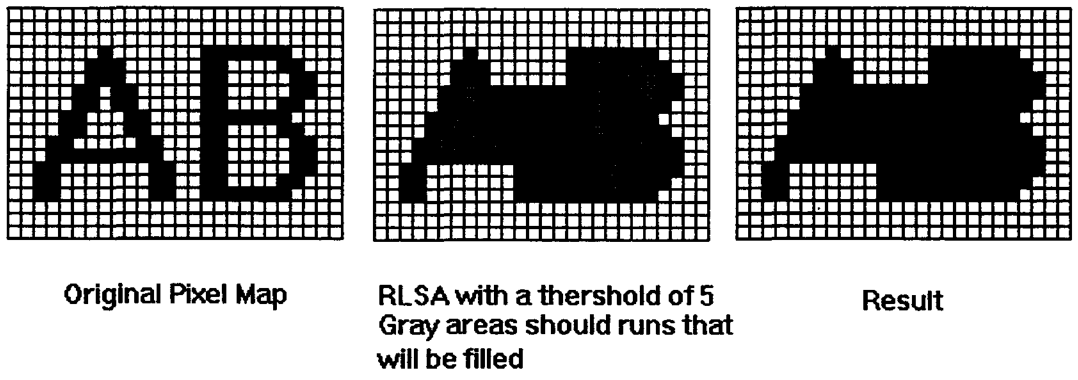
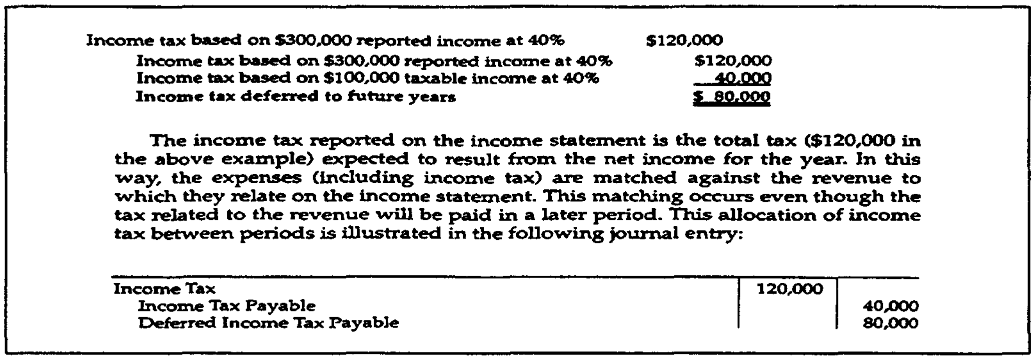


Figure 2. Run-Length Soothing Algorithm

2.3 Bounding Box Algorithm And Text Cell Extraction

The text cell extraction process is based on the well-known bounding box technique[9]. However, to better preserve inter-line spacing characteristics and produce consistent text block baseline coordinates, we attempt to remove descenders from the character bounding Box. Then we create a nearest neighbor graph for those cells that we generate after bounding box process. We further analyze those overlapping cells, based on an algorithm we developed, eliminating or merging some cells to get a non-overlapping text cells graph. In the final step, we merge together those cells which are horizontally close into larger cells. This produces a block graph similar to the one depicted in figure 4.



Income tax based on $300,000 reported income at 40%	$120,000	
Income tax based on $300,000 reported income at 40%		$120,000
Income tax based on $100,000 taxable income at 40%		40,000
Income tax deferred to future years		$ 80,000

The income tax reported on the income statement is the total tax ($120,000 in the above example) expected to result from the net income for the year. In this way, the expenses (including income tax) are matched against the revenue to which they relate on the income statement. This matching occurs even though the tax related to the revenue will be paid in a later period. This allocation of income tax between periods is illustrated in the following journal entry:

Income Tax	120,000	
Income Tax Payable		40,000
Deferred Income Tax Payable		80,000

Figure 3. A Example of a portion of document image

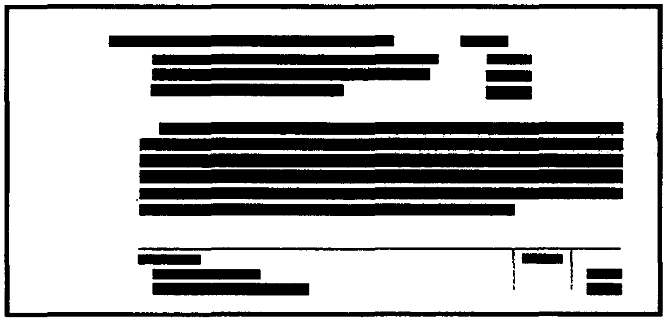


Figure 4. The Result After Line Extraction and Text Cell Extraction of the document shown in figure 3.

2.4 OCR

Once all the text cells have been generated as shown in figure 4, the bitmapped image and the text cells coordinates are passed to an OCR engine to be recognized. After this step, we will get the textual information which include text strings and font information for each text cells. Figure 5, show the result after OCR.

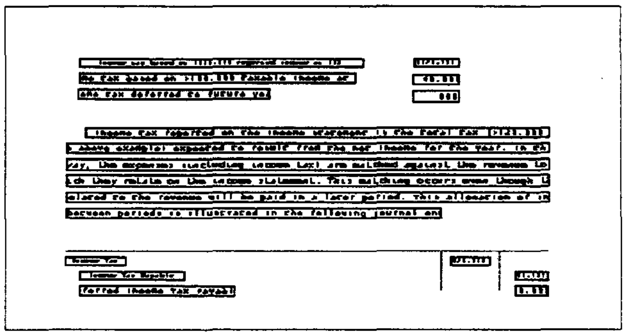


Figure 5. Text Cells after OCR

3 BDOC File Specification

In this section we describe our BDOC 2.0 file specification.

The BCL BDOC file describes the text and/or tables contained in a document. BDOC files can be generated from bi-level (black and white) images, Adobe PDF files, and plain ASCII text files. BDOC files are plain ASCII files. The general format is as follows:

```
<BOF>
<BDOC identification tag>
{
<header information section>
}
{
<line definitions section>
}
{
<phrase definitions section>
}
{
<table definitions section>
}
<EOF>
```

3.1 BDOC Identification Tag

This is a string that uniquely identifies this ASCII file as a BDOC file. Version 2.0 uses **%BDOC%**.

3.2 Header Information Section

The header of a BDOC file contains useful information needed to reconstruct the document. The header contains entries of the form: ***<entry> = <attribute>;***

The following entries are used in BDOC files:

```
Version = <BDOC file version>;
File = <source file>;
FileType = <source file type>;
XYMappingMode = <X-Y mapping mode>;
Origin = <origin of coordinate system>;
Units = <measurement unit>;
ConversionFactor = <conversion factor>;
Resolution = <coordinate points per
inch>;
Width = <width of document>;
Height = <height of document>;
NumberOfLines = <number of line
entries>;
NumberOfPhrases = <number of phrase
entries>;
NumberOfTables = <number of table
entries>;
```

where:

BDOC file version is a floating point number indicating the BDOC file format version.

source file is the full path name of the file used to generate this BDOC file

source file type is the type of document used to generate this BDOC file. This can be any one of the following: IMAGE, PDF, or TEXT.

X-Y mapping mode specifies the coordinate system. It can be any one of the following:

MAP_RIGHT_UP
Positive X extends to the right, positive Y extends up.
MAP_LEFT_UP
Positive X extends to the left, positive Y extends up.
MAP_RIGHT_DOWN
Positive X extends to the right, positive Y extends down.
MAP_LEFT_DOWN
Positive X extends to the left, positive Y extends down.

origin of coordinate system specifies the origin. Can be one of the following:

ORG_TOPLEFT
Origin is at the top left corner.
ORG_TOPRIGHT
Origin is at the top right corner.
ORG_BOTLEFT
Origin is at the bottom left corner.
ORG_BOTRIGHT
Origin is at the bottom right corner.

measurement unit can be one of the following:

INCH	Inches
CM	Centimeters
PIXELS	Pixels

CHAR	Characters (used for text source files)

conversion factor is the scale factor for converting coordinates to measurement units. This field is mostly used with inches and centimeters.

coordinate points per inch is the number of coordinate points per inch. This field is most useful if the measurement unit is pixels.

width of document is the width of the document in coordinate units.

height of document is the height of the document in coordinate units.

number of line entries is the total number of line entries in the line section of this BDOC file.

number of phrase entries is the total number of phrase entries in the phrase section of this BDOC file.

number of table entries is the total number of table entries in the table section of this BDOC file.

3.3 Line Definition Section

A line corresponds to a graphical line appearing on the document. If the document source type is text, this section will be empty. Both horizontal and vertical lines are stored in this section.

A line entry has the following form:

LINE:[*<ID>*, *<x1>*, *<y1>*, *<x2>*, *<y2>*, *<thickness>*];

where:

ID is a unique unsigned integer assigned to this line (integer).

x1, y1, x2, y2 are the endpoint coordinates (integer or float).

thickness is the thickness of the line (must be a floating point number).

3.4 Phrase Definition Section

A phrase corresponds to the bounding box region of text on the page. A phrase can be a partial word, entire word, or series of words that are semantically related. If the document source type is text, the bounding box is simply the line number and first and last character locations.

A phrase entry has the following form:

PHRASE:[*<ID>*, *<x1>*, *<y1>*, *<x2>*, *<y2>*, *<<north>>*, *<<south>>*, *<<east>>*, *<<west>>*, "*<text>*", *<point size>*, "*<font name>*", *<font style>*];

where:

ID is a unique unsigned integer assigned to this phrase (integer).

x1, y1, x2, y2 are the bounding box coordinates (integer or float).

<north> are the IDs of the phrases that are directly above this phrase. (comma delimited integers)

<south> are the IDs of the phrases that are directly below this phrase. (comma delimited integers)

<east> are the IDs of the phrases that are directly to the right this phrase. (comma delimited integers)

<west> are the IDs of the phrases that are directly to the left this phrase. (comma delimited integers)

text is the actual text string.

point size is the point size of the text (not always defined - must be a float).

font name is the name of the font (not always defined).

font style is the font style. Can be one of:

NORMAL	plain text
B	bold
I	italic
U	underlined
BI	bold-italic
BU	bold-underline
IU	italic-underline
BIU	bold-italic-underline

3.5 Table Definition Section

A table definition entry has the following format:

TABLE:[*<ID>*, *<x1>*, *<y1>*, *<x2>*, *<y2>*, *<number of rows>*, *<number of columns>*];
{

```
    CELL:[<_x1>, <_y1>, <_x2>, <_y2>, <t>,
  <l>, <b>, <r>, <# phrases>, <<phrase IDs>>,
  <cell type>];
      .
      .
      .
  }
```

where:

ID is a unique unsigned integer identifier for the table (integer).

x1, y1, x2, y2 are the bounding box coordinates of the entire table (integer or float).

number of rows is the number of rows in the table (integer)

number of columns is the number of columns in the table (integer)

_x1, _y1, _x2, _y2 are the bounding box coordinates of this cell. (integer or float)

t, l, b, r are top, left, bottom, and right (respectively) cell coordinates of this cell. We assume a table has basic row-column indices. Cell coordinates are indices into this basic grid of cells. This allows us to store spanning cells or combined cells. (integers)

# phrases is the number of phrases that this cell contains. (integer)

<phrase IDs> are a set of comma delimited ID numbers that map to the IDs of the phrases defined in the phrase definition section. (integers)

cell type can be one of the following:

C_TITLE	Title block
C_FOOTER	Footer block
C_SUPER_HEADER	Cell is a super column header
C_ROW_LABEL	Cell is a row label
C_COLUMN_HEADING	Cell is a column heading
C_ENTRY	Cell is a basic entry cell (data)

3.6 Text Segment Definition Section

A text segment definition entry has the following format:

```
TEXT_SGM:[<ID>, <x1>, <y1>, <x2>, <y2>];
{
   CELL:[<_x1>, <_y1>, <_x2>, <_y2>, <t>,
<l>, <b>, <r>, <# phrases>, <<phrase IDs>>];
      .
      .
      .
}
```

where:

ID is a unique unsigned integer identifier for the text segments(integer).

x1, y1, x2, y2 are the bounding box coordinates of the entire table (integer or float).

_x1, _y1, _x2, _y2 are the bounding box coordinates of this cell. (integer or float)

t, l, b, r are top, left, bottom, and right (respectively) cell coordinates of this cell. We assume a table has basic row-column indices. Cell coordinates are indices into this basic grid of cells. This allows us to store spanning cells or combined cells. (integers)

# phrases is the number of phrases that this cell contains. (integer)

<phrase IDs> are a set of comma delimited ID numbers that map to the IDs of the phrases defined in the phrase definition section. (integers)

3.7 Layout Definition Section

A layout definition entry has the following format:

```
{
   TopLeft = <Top_leftmost table or text block ID>;

   TopRight = <Top_rightmost table or text block
ID>;
   BottomLeft = <Bottom_ leftmost table or text block
ID>;
   BottomRight = <Bottom_ rightmost table or text
block ID>;

   CELL:[<table or text segment ID>,< <north
neighbors ID>>,<<south neighbors ID>>,
```

<<west neighbors ID>>,<<east neighbors ID>> **];**

```
        .
        .
        .
}
```

*Note: table ID and text segment ID have the same sequence space when we assign them.

3.8 Example BDOC file

```
<BOF>
%BDOC%
{
   Version=2.00;
   File="C:\IMAGES\DOC5.TIF";
   FileType=IMAGE;
   XYMappingMode=MAP_RIGHT_DOWN;
   Origin=ORG_TOP_LEFT;
   Units=PIXELS;
   ConversionFactor=1;
   Resolution=300;
   Width=2560;
   Height=3301;
   NumberOfPhrases=2;
   NumberOfPhrases=4;
   NumberOfTables=1;
}
{
   LINE:[1,10,50,10,100,3.00];
   LINE:[2,200,25,400,25,5.00];
   .
   .           <additional line entries>
   .


}
{
   PHRASE:[1,10,10,20,15,<3,5>,<>,<6>,<2,
8,9>,"Total Cost",6.00,"Helvetica",BI];
   PHRASE:[3,50,260,100,275,<1>,<5>,<>,<4
,7>,"Revenues",0.00,"",NORMAL];
   PHRASE:[4,100,110,143,122,<>,<>,<3>,<>
,"$560.00",8.20,"Courier",U];
   PHRASE:[5,10,10,20,15,<3,4>,<6,7,9>,<1
>,<8>,"1,700.00",0.00,"",NORMAL];
   .
   .           <additional phrase entries>
   .
}
{
   TABLE:[1,33,57,120,200,3,4];
   {
      CELL:[13.00,26.00,45.00,57.00,2,2,2
   ,2,2,<3,4>,C_COLUMN_HEADING];
      CELL:[31.45,63.72,55.00,79.01,4,5,8
   ,8,1,<5>,C_ENTRY];
      .
      .        <additional cell entries>
      .
   }
   .
   .           <additional table entries>
   .
}
{
   TEXT:[11,1120,57,2210,570];
   {
      CELL:[1120,57,1148,130,2,<10,11>];
      .
      .
      .                  <additional text
block entries];
   }
   TEXT:[12,20,2000,2210,3100];
   {
      .
      .
      .
   }
   .
   .
   .
}
{
   TopLeft = 1;
   TopRight = 11;
   BottomLeft = 12;
   BottomRight = 12;

   CELL:[1, <>,<2>,<>,<3,4>];
      .
      .
      .
}

<EOF>
```

4 Examples of Usage of the BDOC file

In this section, we give two examples on how to use the BDOC file on further document analysis and give you a flavor of how the BDOC file allow the easy access of the information that we catch on the first pass.

4.1 Nearest Neighbor

During the logical inference, it's quite often that we want to find the nearest neighbor for a given table, text segment or even the nearest neighbor for a given cell. As we can see, BDOC file keeps two nearest neighbor graphs, one is overall layout graph for text segments and tables, one is for the whole cells inside the given document. With these two graphs, we will be very easy to program functions to access the neighbor for a cell or block from either directions we want. The following codes is an example of how to access the north neighbor for the given table or the text block "pTable1", and it's given in the codes that is under MFC.

```
/* Ctable & Ctext are two classes that
derived from their parents Csegment;
SegList is the type CO pos; ){
pSeg = bList, and it keeps all the tables
and text segments;
return values: NULL, if there exist no
NORTH neighbor;
a point of type Csegment that point to the
first NORTH neighbor in the list */

Csegment* GetNorth(  Csegment* pTable1 )
{
        Csegment*  pSeg=NULL;
        POSITION pos;

        pos=pTable1-
>m_Dir[NORTH].GetHeadPosition();
        if( pos )
        pSeg =  pTable1-
>m_Dir[NORTH].GetNext(pos);
        return( pSeg );
}
```

Figure 6. A sample code of how to access the NORTH neighbor of a given segment

4.2 Reading order assignment

How to retrieve the reading order of a given document after it has been stored in the electric format is the question of how to correctly inference the logical structure of a document's content from its physical structure. Since a BDOC file keeps both the geometric and semantic information of a document, it's very easy for us to program the codes to give the right reading order sequence. For example, with the layout structure shown in figure 7, since no segment(table or text block) has east and west neighbor, it's very clear that we have the reading order starting from the top segment on the page followed by its single SOUTH neighbor till the end of page.

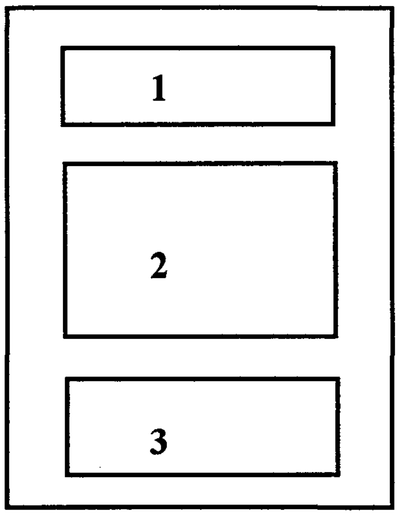
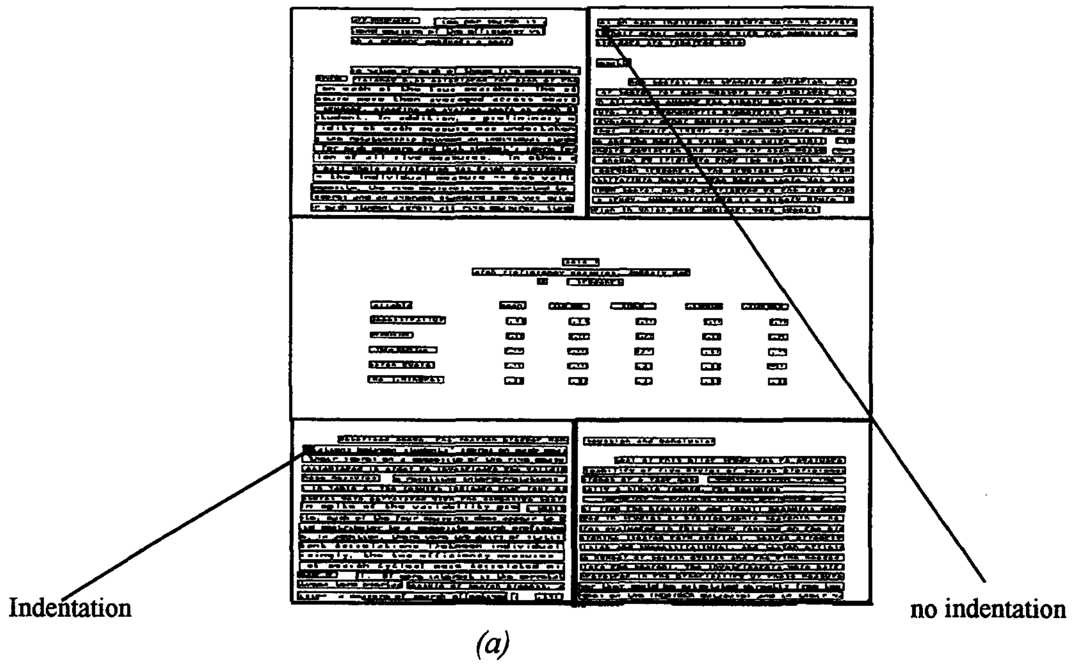


Figure 7. Reading Order Example with segments have no West/East neighbor

Figure 8 is an little complicated example with the reading order has two possible solutions. As shown in figure 8 (b) and (c). But if we give more detail analysis, we will find that indentation happens on the first cell of the segment after the one column table segment and there is no indentation on the first cell of its right neighbor segment, so the right reading order should be that shown in figure 8 (c).



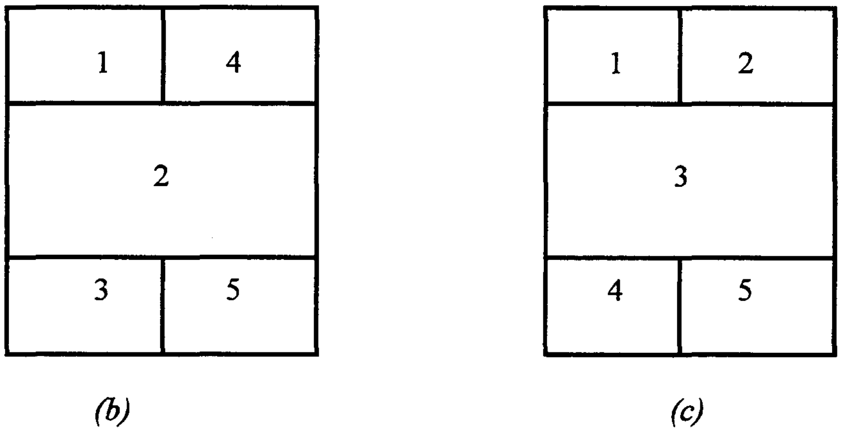


Figure 8. A document and its two possible reading order sequence
(a) the original document's BDOC.
(b) one reading order (c) the other reading order

5 Future Work

Building a complete document processing system still may be decades away. In order to reach this goal, more effort should be devoted to it. New approaches should be developed. Interdisciplinary studies, as well as pattern recognition techniques associated with AI methods(such as natural language processing), information science(such as information retrieval), knowledge engineering, linguistics and other aspects should be explored and employed. BDOC as an intermediate representation of a document that capture much of the information used in structural inference will serve as an good reference for further analysis.

Bibliography

[1] Y. Y. Tang, C. Y. Suen, and C. D. Yan, "Document processing for automatic knowledge aquisition", *IEEE Trans. Knowledge Data Eng.*, 1994(in press).

[2] S. M. Kerpedjiev, "Automatic extraction of information structures from documents", *Proc. First Int. Conf. on Document Analysis and Recognition,* Saint-Malo, France, Sept.-Oct. 1991, pp. 525-532

[3] G. Nagy, "What does a machine need to know to read a document", *Proc. Symp. on Document Analysis and Information Retrieval,* March 1992, pp. 1-10.

[4] Y. Y. Tang, C. D. Yan, M. Cheriet, and C. Y. Suen, "Automatic analysis and understanding of documents", *Handbook of Pattern Recognition and Computer Vision,* S. P. Wang, C. H. Chen, and L. F. Pau, eds., World Scientific Publ. Co. Pte Ltd., 1993, pp. 625-654.

[5] Y. Y. Tang, C. D. Yan, M. Cheriet, and C. Y. Suen, "Document analysis and understanding: A brief survey", *Proc. First Int. Conf. on Document Analysis and Recognition,* Saint-Malo, France, Sept.-Oct. 1991, pp. 17-31

[6] R. N. Ascher, G. M. Koppelman, M. J. Miller, G. Nagy, and G. L. Shelton Jr., "An interactive system for reading unformatted printed text", *IEEE Trans. Computers C-20,* 12(1971)1527-1543.

[7] G. Nagy, "A preliminary investigation of techniques for the automated reading of unformatted text", *Commun. ACM* 11, 7 (1968) 480-487.

[8] Wong, K.Y., *et al.* "Document Analysis Systems", *IBM Journal on Research and Development.* Volume 26, No. 6, pp.647-656, 1982

[9] Mahadevan, U and Nagabushnam, R.C, "Gap metrics for word separation in handwritten lines" *Proc. of the 3rd Intl. Conf. on Document Analysis and Recognition,* Vol. 1, pp. 124-127, 1995

FaxAssist: Inbound Fax Routing Using Document Understanding

Scott Tupaj Horace Dediu Hassan Alam

BCL Computers, 600 West Cummings Park, S1500, Woburn, MA 01801

Abstract

The feasibility of a fax routing method based on document understanding was tested. Initial research results confirmed this feasibility hypothesis. A complete fax server and routing application which met the precision and recall specifications was built. The application was able to receive electronic fax transmissions through a fax modem, perform document analysis on the fax contents, and route the image and text of the fax through an e-mail server to the intended recipient with 95% or greater precision on a large and variable set of sample faxes.

1 Identification of Significance

As faxes have been integrated into organizational and enterprise communications infrastructures, the cycle time, throughput and frequency of fax transmission has increased while the cost of faxing has decreased. Throughout the explosion in Internet usage, the amount of faxing has continued to grow. According to a 1996 Pitney Bowes/Gallup poll, fax is still the preferred method of business communication among both Fortune 500 and mid-sized business users. Davidson Consulting, which estimates that 336 billion pages will be faxed worldwide in 1996, surveyed fax machine end users and dealers and found that fax toner usage on existing machines rose at a 12% annual rate from 1995 to 1996. IDC/Link Resources reports that the computer-based fax market will sustain a 28% compound annual growth rate from 1995 through 1999. Fax servers have been used to increase the efficiency of fax transmission, thus lowering the overall cost of this medium of communication.

Fax servers are shared fax resources installed on Local Area Networks (LAN) and multi-user computer networks. They typically are installed on a gateway PC or file server. They enable network users to:

- Have computer files transmitted as faxes to any fax machine or device
- Receive faxes from any fax machine or device at the fax server (where the fax phone call terminates) either for automatic print-out or to be routed via some mechanism to fax or universal-message mailboxes associated with each individual end user (or department or workgroup) on the LAN. Faxes are received as image files not as computer-editable alpha-numeric files.

This paper discusses the use of document understanding technology pioneered by BCL Computers to develop a new fax routing application that provides unprecedented levels of precision.

2 The Inbound Fax Routing Problem

Typically, faxes received by a fax server remain on the server until manually routed by an operator who reads the cover page and decides who is the intended recipient. This poses several problems: first, the need for human intervention in the inbound fax processing leaves it open to delays and bottlenecks (as when the operator is unavailable). Second, the cost of the time spent routing can often offset the savings of using a server and invalidate its economic value. Finally, the privacy of the faxes cannot be guaranteed as the operator has access to the fax pages.

Ideally the inbound fax should be as transparent to the user as an inbound e-mail. It should be:

- Delivered in a timely fashion (as soon as possible),
- Delivered to any user location (remotely retrievable),

- Integrated into an on-screen viewer,
- The image as well as character content should be available for use in other applications,
- Faxes should be electronically archived and indexed,
- Fax privacy should be controlled.

Besides having a clerk manually route faxes as they arrive to recipients' in-boxes, several automated methods of routing inbound faxes have been proposed:

DTMF: Dual-tone multi-frequency (DTMF) or TouchTone. It involves the sender dialing an extension number so a fax can be routed beyond the fax server to a recipient workstation. Senders must know recipients' fax phone numbers and extensions and, once the receive-end fax system answers the fax call, to enter the extension number at the proper time (this sometimes is voice-prompted). Since DTMF involves manual input, it doesn't work with automated fax transmissions (delayed sends, auto re-dials -- the kind of automated send that fax servers employ). In some areas and countries where TouchTone is not in use, DTMF may not be available or widely used by potential senders.

Modified Handshaking: With this technology, a sub-address is inserted into the fax handshake process. Modified handshaking differs from DTMF in that sub-addresses are detected during the handshake, so it works with delayed send, auto re-dial, etc. But both the sending and receiving fax systems must be compatible with the same modified handshaking system. The first modified handshaking systems were proprietary. In 1993, the ITU-T created a standard method called T.33 sub-addressing. Fewer than 1% of the fax machine installed base supports any form of sub-addressing, putting in question its applicability in anything besides highly controlled environments.

DID: Direct-inward-dialing employs the same telephone technology that allows people in large office buildings to have a "direct line." With DID and LAN fax, each workstation has a direct fax phone number, but calls to all the different users' fax phone numbers arrive over a single phone line (or common bank of multiple lines). To install DID requires a fax server equipped with DID-compatible fax boards or a DID gateway or conversion module. Then the telephone company must install one or more special DID lines (they're receive-only; additional regular phone lines are needed to send outbound faxes). The phone company assigns DID numbers (often in packs of 10, 50 or 100) and then each user gets a fax mailbox corresponding to that user's unique seven-digit fax number. All of the fax phone numbers assigned arrive over the same DID phone line(s) and the DID system in the fax server node matches the seven-digit number to the proper mailbox. Remote senders only need to know recipients' ordinary fax phone numbers. Separate DID lines incur monthly charges from the phone company plus potentially substantial installation fees and delays. Sometimes installing DID is complicated or impossible.

TTI: Transmit terminal identifier (TTI) is the term for the ID message that prints out at the top margin of received faxes. It identifies the sender, usually with a fax phone number and/or company name. Fax servers can be programmed to recognize a TTI and route all faxes with that TTI to a particular workstation and only that workstation, making it suitable for niche applications (e.g., POs routed to users by geographic region).

Line Routing: With multi-line fax servers, faxes received on each line (each with its own phone number) can be routed automatically to a particular department and then be auto printed or manually or automatically routed from there. For example, if a fax server has four lines, each line can receive faxes for a different department (and all four lines can still be used in a shared mode to handle all outbound faxes).

OCR/ICR: Optical and intelligent character recognition (OCR & ICR) may be used for automated inbound routing, with some notable caveats. With some OCR-based systems, senders must type the receiver's name in coded format -- e.g., <<Peter Davidson>> and then a properly equipped fax server tries to recognize the code to route the fax. With ICR, handprint can be used and the fax server can search the entire cover sheet for clues (recipient name, department) via which to route faxes (LAN administrators must input all user names, including nicknames and even common misspellings). In either case, precision is quite poor and the methods are usually considered only expected to reduce the number of received faxes needing to be manually routed.

Table 1. Comparison of existing routing methods

METHOD	APPROACH	RESTRICTIONS
Dual-tone multi-frequency (DTMF) or TouchTone	The sender dials an extension number so a fax can be routed beyond the fax server to a recipient workstation	• Doesn't work with automated fax transmissions • Senders must know recipients' fax phone numbers and extensions • Requires TouchTone • Requires n lines for n users
Modified Handshaking	A sub-address is inserted into the fax handshake process	• Requires T.33 hardware. • Fewer than 1% of the fax machine installed base supports any form of sub-addressing
Direct-inward-dialing (DID)	Each workstation has a direct fax phone number	• Requires a fax server equipped with DID-compatible fax boards or a DID gateway or conversion module • The telephone company must install one or more receive-only DID lines • DID lines incur monthly charges • Remote senders need to know recipients' individual fax phone numbers • Sometimes installing DID is complicated or impossible
TTI: Transmit terminal identifier	An ID message that prints out at the top margin of received faxes identifies the sender, usually with a fax phone number and/or company name	• Requires sending unit to program the TTI routing message. • Sending units are not easily programmable
Line Routing	Faxes are received on separate lines, each with its own phone number	• Routing end-points are limited to the physical number of lines. • Manual routing is necessary for unaddressed faxes.
OCR/ICR	Uses character recognition to find user names	• senders must type the receiver's name in coded format • precision is quite poor

The absence of a satisfactorily precise fully automatic and backward compatible routing technology is the direct consequence of the nature of fax. Faxes are analogous to standard business correspondence. It is because of this analogy that they have been so successful in penetrating the communications infrastructures of business. The ancient and proven business process of written communication via paper was adapted for transmission over telephone lines quickly because the switching costs were low. The process was left essentially unchanged. The envelope which, in paper transmission, was used to ensure privacy and routing, was replaced by the cover page. However, unlike envelopes and their somewhat standard addressing methods, cover pages were never standardized. Furthermore, due to the cost of transmission, cover pages started being used for messaging as well as addressing, further complicating the standards problem.

3 Summary of Objectives

The problem of fax routing becomes the same as that of correspondence routing. The understanding of the fax contents and the identification of the recipient is similar to the problem of understanding business correspondence and identification of addressing features in the image. Once the addressing features are identified, the problem is one of deciding among the possible recipients which is the intended. This is not the same as positively decoding the name from the image, but rather, can be thought of as eliminating all the names which *are not* the intended, thereby maximizing precision (minimizing mis-routing) not recall.

While we planned to identify other blocks, we focused on the address and message blocks because the address block is crucial to routing. The message block is crucial to avoid mis-routing.

Our work addressed the following technical objectives:

- Address identification through feature analysis
- Keyword detection
- Distance measurement between target strings and address blocks
- Demonstrable implementation

The complete solution is under investigation, however initial results have proven a precision of 95% and a recall of 93%. That is, 5% of the faxes were mis-routed, and 7% were not decidable and therefore routed to a manual in-box. These results are highly sensitive to the sample set and therefore this set will be refined to reflect the most accurate approximation of actual operating environments. Our ultimate goal is to achieve 100% precision and above 95% recall.

3.1 Address Block Identification

The need for address block identification is due to the large amount of noise information on the page which is not relevant to the routing decision. The more information on the page, the more time is required to perform analysis.

The methods used in finding the address block consist of rules for image cleanup (de-skewing, shade removal) geometric feature extraction (line identification, block bounding box detection) and feature pattern analysis (attribute-value pair detection). A rules engine combines the result of this feature extraction process and eliminates blocks of the image from further analysis. The result is a set of blocks of text which *most probably* contain the recipient name. The system automatically polls the e-mail server for the name list which shows the complete names of the possible recipients, keeping the list up to date and available at all times.

3.2 Keyword Detection

In order to reduce the confusion between recipient names and sender names on the same page and within the address block areas, keywords can be used to infer the sender/recipient relationship.

The keywords which might help the identification process are found through optical character recognition (OCR). The OCR engine (from Xerox) is tuned for fax analysis. This implies 200x200 resolution and only black-on-white images. The keywords are searched for in the address block sections only. Typical keywords are "TO:", "ATTN.", etc. A large number of keywords was selected including titles, directives, etc. Furthermore, last name, first name and middle initial permutations were devised to better find the name on a page. We also looked into the problem of misspelled names, abbreviated names, and nicknames being used.

3.3 String Distance Measurement

When comparing the result of OCR on a candidate block with the name list, we measure the distance from every name (last name augmented by first name substrings) to substrings of the candidate text block. A distance metric is used to assess degrees of similarity.

3.4 Demonstrable Implementation

The testing environment was chosen as a Microsoft Exchange e-mail environment. The incoming faxes are processed by a robot Exchange client. A routing engine reads the queue and extracts any incoming fax, converting it to a TIFF format. The TIFF cover page is sent to the analysis engine which reads the recipient list and decides the routing. The analysis engine also extracts the text and layout of the fax message. The image, address and text are then encapsulated into an e-mail message and sent through the Exchange server to the correct mailbox. The system requires the use of an NT server for mail, fax receipt, analysis and routing.

Figure 1 shows the system diagram of the implementation of the routing algorithm.

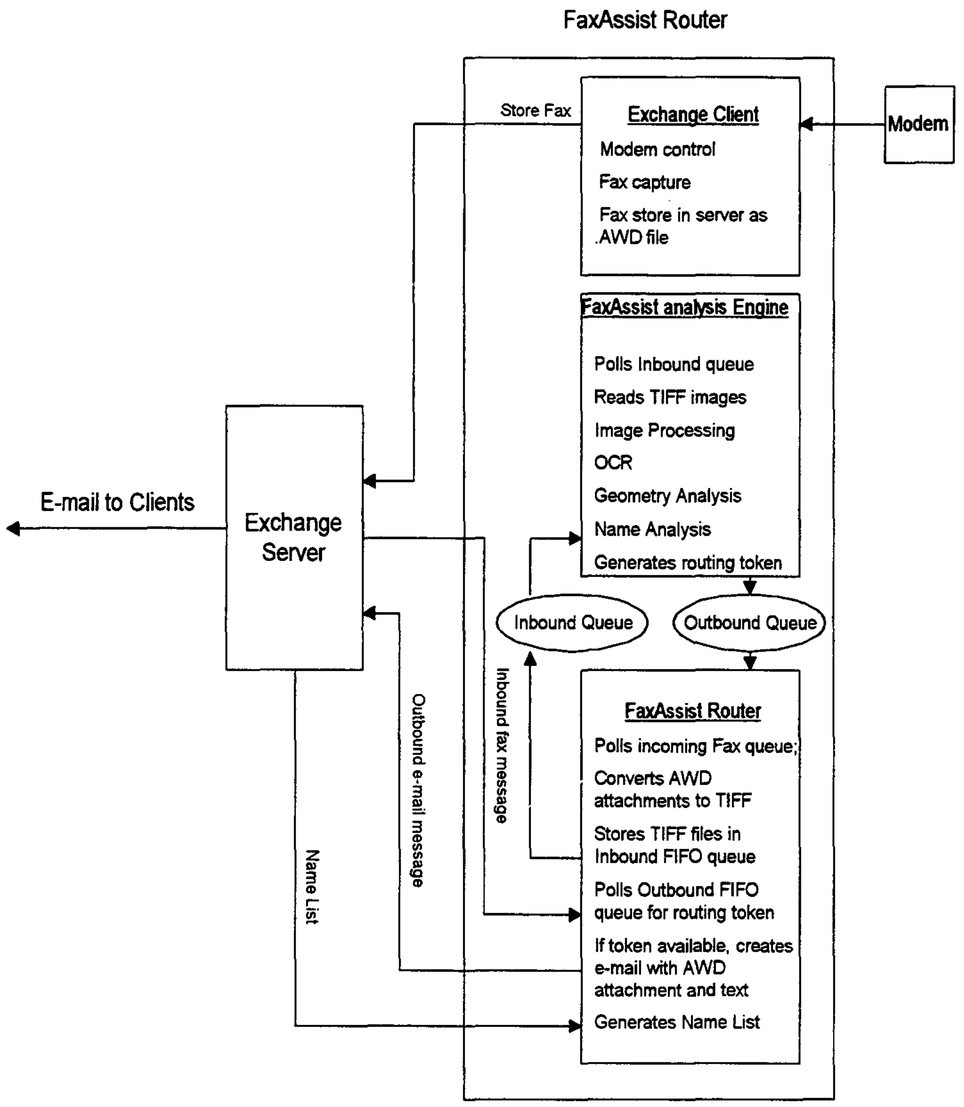


Figure 1. System diagram for the implementation of the Phase I routing algorithm.

4 Implementation

The approach is to build an analysis engine that receives a fax cover page image and maps it to any one of a finite number of recipients based on the contents of the cover page. The analysis uses the results of OCR to search the name list for the correct recipient. However, since OCR accuracy is not high enough for a direct name match, and a single name may have multiple permutations of first name, last name, and initials, a fuzzy string matching technique is used to weight each possible name occurrence on the cover page. To decrease the possibility of false name identifications (e.g., if more than one name appears in the text of the cover page) page layout analysis is used to disregard text blocks that potentially represent body text and fields that do not contain a recipient name. Also, geometric proximity to certain keywords that indicate a recipient field is used to increase accuracy.

The Fax-Assist analysis engine receives as input a fax image in TIFF format and a list of possible recipients. If the image is a low-resolution (200x100 dpi) fax, the resolution is interpolated to a 200x200 dpi image by duplicating each scan line. Thus, the 1:1 image aspect ratio is preserved. The recipient list is an ASCII file containing the full name of each possible recipient as well as their corresponding e-mail address. If the analysis engine isolates the recipient in the list of names, the corresponding e-mail of the recipient is used in the routing process. If the recipient is not in the list, or if the analysis engine could not discern a recipient with a high degree of confidence, the engine returns the e-mail of an account that handles faxes that must be routed manually.

The analysis engine performs the steps shown in Figure 2. Each of the steps is described in detail below.

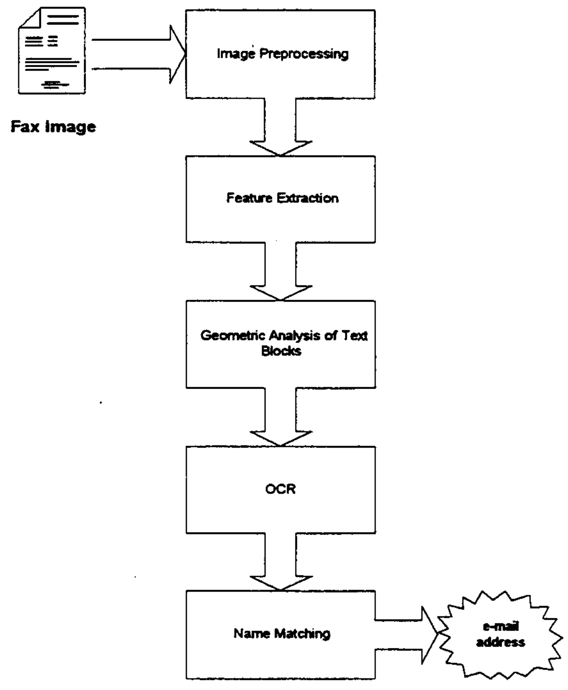


Figure 2: Fax-Assist analysis steps

4.1 Image Preprocessing

Because faxes are often skewed and may contain noise as a result of scanning, the fax cover page image is preprocessed before analysis can be performed. Skew and noise often cause unwanted character recognition errors. The ScanFix™ software package from Sequoia Systems was used for image preprocessing. Parameters for skew, dot shading, inverse text, and speck removal were tuned to optimize the cleaning of fax images.

4.2 Feature Extraction

The location of text and graphical lines are recorded in order to reconstruct the page layout of the fax cover page. Text entities are expressed as rectangular objects that are formed using the character bounding boxes. First, the image is smeared using a global threshold estimated to be the average inter-word spacing gap. This effectively fills in white space between characters in words. Word bounding boxes are formed by calculating the bounding boxes on the smeared image and approximating where the baseline of the text exists. To better approximate the font size of the text blocks, we set the lower coordinate of the word bounding box to the baseline of the text, which removed the descenders of the characters. Word bounding boxes on the same baseline are then merged based on the horizontal gap between the blocks relative to the heights of the blocks. The end result is a set of bounding boxes representing logical phrases of words that are approximately the same font size. Figure 3 shows an example.

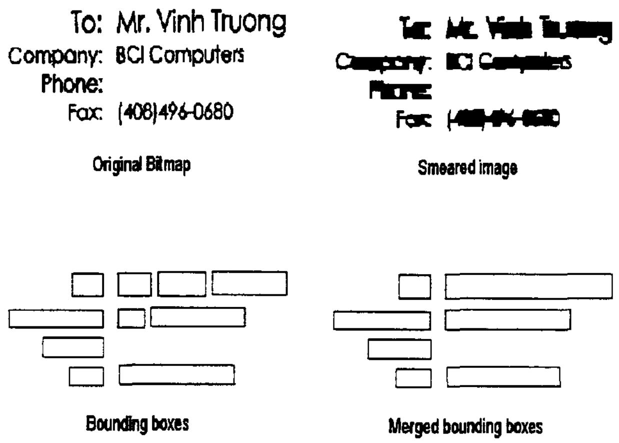


Figure 3 - Text block extraction

In addition to text blocks, horizontal and vertical lines are isolated and removed.

4.3 Geometric Analysis of Text Blocks

Before character recognition is performed on each text block, the positional information of each block can be used to exclude certain text blocks from the analysis. This both increases speed and reduces the probability falsely isolating a recipient field by reducing the search space on the page. Text alignment, vertical spacing, and block length are clues used to determine if body text appears on the cover page.

4.4 OCR

This step returns the characters associated with each block of text. The OCR engine used is TextBridge by Xerox. The OCR engine was tuned for optimum performance on 200 dpi fax-degraded images. Each text block calculated in the feature extraction process is sent to the OCR engine. Since text blocks represent the boundaries of words and phrases without the character descenders, the text blocks are augmented vertically to include any descender information.

4.5 Name Matching

The approach for finding the best name match is the following. For each name in the recipient list, find all occurrences of the name among the candidate text blocks. A name occurrence exists if any permutation of the first and last name of the recipient matches, above a certain threshold, the text or any substring of the text in any block.

When comparing the results of OCR on a candidate block with the name list, we measure the distance from every name (last name augmented by first name substrings) to substrings of the candidate text block. A distance metric is used to assess degrees of similarity. The distance between two strings may be evaluated using the Wagner-Fischer algorithm implementing the Levenshtein string edit distance measure. For two strings of equal length, the Hamming distance between them is the number of symbol positions at which they differ. This is the most common method and it measures the minimum cost of transforming the first string into the second when only substitutions are permitted and are given unit weighting. When not restricted to comparing equal-length strings, insertions and deletions will also be required. When these are given the same weight as a substitution, the minimum total transformation cost is a metric proposed by Levenshtein (1965). For our purposes, we used a *weighted* Levenshtein distance where substitution errors are weighted by the probabilistic confusion matrix of the OCR system in use.

Therefore this method allows us to compare unequal length strings, accounting for the errors OCR is likely to make and measuring the distance of any name string with the substrings of the target blocks. If the minimum distance of some name exceeds some threshold, then the target block most likely contains that name.

When a set of name occurrences has been generated, the following criteria are used to find if a best match is present. If no best match is present,the system returns the e-mail address of the manual router.

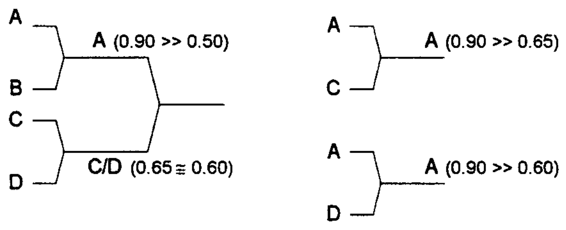
- Degree of string match
- Whether recipient keywords exists
- Distance to recipient field keyword

The following recipient keywords were used in the analysis: TO:, ATTN:, NAME:, ATTENTION:, MR., MRS., MS, DR., DEAR. The FROM: keyword was also used as a negative weight in the analysis. Keyword matching is done using the same string distance method used for the name matching.

Our analysis takes the best eight name occurrences and does a pair-wise comparison between each one to determine which has the higher probability of being the correct recipient. If the comparison between any two name occurrences yields an ambiguous result (i.e., neither name occurrence scores significantly higher than the other using the criteria outlined above), then both name occurrences are considered in the next level of comparisons. Figure 3 gives an example.

Given four name occurrences and their respective probabilities for recipient match:

{ A(0.90), B(0.50), C(0.65), D(0.60) }



In both cases, A has the best probability for match.

Figure 4: Binary comparisons of each name occurrence

If the process discerns one name occurrence as being the best match among the top eight name occurrences, then the e-mail address associated with this name is returned. Otherwise, the system returns the e-mail address of the manual router account, thus classifying this fax as un-routable based on its cover page contents.

5 Challenge Areas

The above summary has clearly demonstrated the feasibility of approach for document analysis-based fax routing. The methods applied proved worthwhile. However, several problem areas have been identified and will be addressed in future effort. These are summarized as follows:

5.1 Adaptive Threshold Assignment

The decision of which name most closely matches the target block must be made given a threshold of minimum distance allowable. This threshold distance is a function of the length of the name string as well as the quality of the image. Image quality can be measured in several ways, but one method we are considering is the ratio of Huffman encoded document image size (total information content of the bitmap) to the number of characters extracted via OCR. This is analogous to a Signal-to-Noise ratio. The S/N measure is used to adapt the threshold of acceptability of any calculated string edit distance.

5.2 Rules Base for Decision

The combination of image, keyword, and OCR edit distance analysis results are used to make two decisions: 1. Which block is most likely to hold the name of the recipient; 2. Which name from the expected recipient list is closest to the name on the page. A rules base combines the results of these analysis and makes the final judgment. The results can be either: name is found with a high degree of certainty, name is not found, and name is uncertain. Two measures of accuracy can then be calculated: precision and recall. Precision measures the effect of misrouting (a name is found with certainty but it is the wrong name), recall measures the effect of not finding the name with certainty. Our rules base must be trained to maximize precision.

5.3 Sample Selection

The question we need to ask is how many samples are sufficient to give credence to our accuracy claims (latest are Precision of 98%, Recall of 93%).

The answer can be arrived to by calculations of the minimum sample size which exhibits statistical properties of mean and variance within a maximum error of the mean for a sample set given that the set is normally distributed, we choose a confidence interval, the sample mean is estimated, the sample standard deviation is estimated.

The first question is: What is the sample's measurable statistic? We chose to use file size as a proxy for the page complexity. This is true because we use Huffman encoding for compression. Huffman is optimized for business communications (printed text) and, since it's a lossless compression method, is a measurement of the information content of the document. Noise, handwriting and graphics and skew all create larger files.

Given that we can use file size as our measure of complexity/variance, we need to test that the samples are normally distributed. A Chi-squared test of goodness of fit confirms this, as does a confidence interval for an estimate of the mean of a normal population.

Therefore, we know the size statistic is representative of the variance in the sample set (from handwritten to printed), the size statistic is normally distributed, we measured the mean and standard deviation of a representative set, the confidence intervals are available from a look-up table and we can assign the maximum error of the mean to 5%.

This means that with complete confidence, a sample set of 1408 samples will result in a mean within 5% of expected mean for a normally distributed set of files with standard deviation-mean ratio of 0.57. We need to find sample sets which fit these criteria in order to tune and test our algorithms.

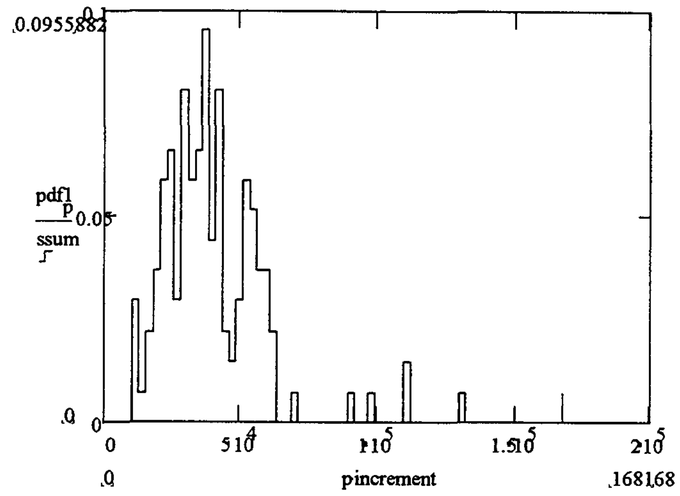


Figure 5. Sample Set "Del" Probability Density Function of File Size Parameter

We generated three sample sets. One had a large variance in the format of the image, the second varied the number of recipients, the third introduced uniform noise in the image.

Table 2. Sample Sets

SET NAME	SOURCE	SAMPLES	NAMES IN TARGET LIST	IMAGE QUALITY	NUMBER OF FORMATS
Del	• Auto Dealer fax-back, • Fax software sample cover pages • Miscellaneous	100	25	Medium	100
BCL_Fax	• Word processor mail merge output	686	229	Good	4
BCL_Fax 2	• Word processor output, low resolution printing with hand-fed fax machine skew	114	229	Poor	4

6 Performance Measurements

Performance was tested with several algorithm configurations and the results are shown in the following table.

Table 3. Sample Sets performance measurements.

Sample Set: **"Del"**
Number of Files: 100

Trial	Correct Routes	False Routes	Rejects	Precision	Recall	Performance (% correctly routed)	Time (sec/page)
1	75	25	0	0.750	1.000	0.750	-
2	87	8	5	0.916	0.946	0.870	18
3	88	5	7	0.946	0.926	0.880	48
4	91	2	7	0.978	0.929	0.910	48
5	80	4	16	0.952	0.833	0.800	

Sample Set: **"BCL_Fax"**
Number of Files: 686

Trial	Correct Routes	False Routes	Rejects	Precision	Recall	Performance (% correctly routed)	Time (sec/page)
1	605	63	18	0.906	0.971	0.882	60
2	650	17	19	0.975	0.972	0.948	50

Sample Set: **"BCL_Fax2"**
Number of Files: 114

Trial	Correct Routes	False Routes	Rejects	Precision	Recall	Performance (% correctly routed)	Time (sec/page)
1	88	7	19	0.926	0.822	0.772	60

7 Conclusion

Each of the research objectives were achieved and the results show the basis for further development of a prototype for universal automated fax routing. Analysis of address block identification, keyword detection, and string distance measurement methods supported this feasibility analysis. Problem areas for performance, rules bases, and sample sensitivity were identified in the course of the effort and have been described. BCL believes that this approach will minimize the difficulty in solving the research problems encountered.

We intend to extend the capabilities of the fax router demonstrated for application in the field of fax management and automation for the Department of Defense. We plan to deploy a prototype system within the offices of DARPA and use the test environment to refine the analysis engine to the point where it can be deployed universally throughout the DOD.

8 Bibliography

[1] Stephen, G.A., *String Searching Algorithms, Lecture Notes Series on Computing – Vol. 3*, World Scientific, 1994.

[2] Wagner, R.A., Fischer, M.J., "The string-to-string correction problem," *Journal of the ACM*, Vol. 21, No.1, p. 168-73, January 1974.

[3] Bayer, T.A. & Walischewski, H. "Experiments on extracting Structural Information from Paper Documents Using Syntactic Pattern Analysis," page 476. *Third International Conference on Document Analysis and Recognition.*

[4] Dengel, A.; Belesinger, R.; Fein, F.; Hoch, R.; Hones, F.;Malburg, M.;. "OfficeMAID - A system for office mail analysis, interpretation and delivery," *International Association for Pattern Recognition on Document Analyses Systems*, 1994 p.p. 253-276.

[5] Doermann, D. Page, "Decomposition and Related Research at the University of Maryland," page 39. 1995 *Symposium on Document Image Understanding Technology.*

[6] Gonczarowski, Jakob; Wilhelm, Reinhard. "Structure in Text Formatting". *IEEE Computer*

Society Office Automation Symposium, Gaithersburg, MD., USA 1987 Apr. 27-29 p 9-14.

[7] Govindaraju, Venu. "Decomposition and Structural Analysis of Documents". *Document Understanding Workshop*, Xerox Palo Alto Research Center, May 6-8 1992.

[8] Haralick, R.M. & Phillips, I. & Chen, S. & Ha, J. "Document Structural Decomposition," page 27. *1995 Symposium on Document Image Understanding Technology.*

[9] Lii, J. & Srihari, S.N. "Location of Name and Address on Fax Cover Pages", page 756. *Third International Conference on Document Analysis and Recognition.*

[10] Lipshutz, Mark & Taylor, Suzanne Liebowitz, "Functional Decomposition of Business Letters", page 435. *Fourth Annual Symposium on Document Analysis and Information Retrieval.*

[11] Lipshutz, Mark; Taylor, Suzanne Liebowitz. "Decomposition and Structural Analysis of Scientific and Technical Documents". *Document Understanding Workshop*, Xerox Palo Alto Research Center, May 6-8 1992.

[12] Niyogi, D. & Srihari, S.N. "Knowledge-Based Derivation of Document Logical Structure", page 472. *Third International Conference on Document Analysis and Recognition.*

[13] Sivaramakrishnan, R. & Phillips, I.T. & Ha, J. & Subramanium, S. & Haralick, R.M., "Zone Classification in a Document Using the Method of Feature Vector Generation", page 541. *Third International Conference on Document Analysis and Recognition.*

[14] Taylor, S.L. & Lipshutz, M. & Nilson, R.W. "Document Classification and Functional Decomposition", page 56. *1995 Symposium on Document Image Understanding Technology.*

Recognition and Architectures

Recognition and Characterization of Handwritten Words

Gregg Wilensky, Richard Crawford
Logicon RDA, P.O. Box 92500, Los Angeles, California 90009
Ronald Riley
Logicon RDA, 2100 Washington Blvd., Arlington, Virginia 22204-5706
E-mail: {gwilensky, rriley, rcrawford}@logicon.com.

Abstract

We present new techniques and results for word recognition and for writer identification from documents containing handwritten words. These problems are complementary and require the development of features which are dependent upon word content but independent of stylistic variations for the first problem and dependent upon style but independent of content for the second. Accordingly, we have developed both a set of style-dependent features and a set of style-independent features. These features have been applied to two handwritten document databases: 1) a handwritten signature database collected from company timecards for word (signature) recognition and 2) a Chinese document database for writer identification. Results are presented for both.

For solving the style-independent problem, we present a new metric for invariant image recognition. This metric offers a solution to a long-standing problem in image recognition, the accurate detection and recognition of objects in spite of the presence of noise, distortions in the shape of the image (either due to natural variation or differences in points of view), and clutter which overlaps and obscures parts of the object.

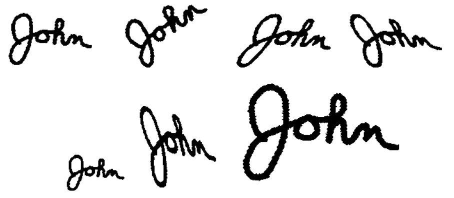
1 Introduction

Two complementary image recognition problems occur in the analysis of handwritten words. One is the recognition of a word regardless of the writer and requires style-independent features. The other is the identification of the writer based on stylistic features, independent of what is being written. In this report we address both of these problems by developing a set of style-dependent and style-independent features. For the former we present a set of measures of the means and the statistical variation among word sizes, shapes and textures. We apply these to identify writers in a database of handwritten Chinese documents. For the style-independent features we develop a new measure of closeness between two images which is insensitive to image distortions, to noise, to overlapping text and to global transformations such as scale, translation, rotation and skew[1].

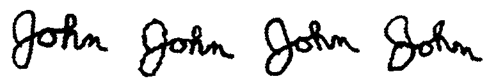
The motivation for developing such a measure of image closeness is to recognize two images as corresponding to the same object even though they may differ because of translation, scale, shear, local distortions or noise. To accomplish this, a measure of distance between two images is needed which is insensitive to small local distortions and is at least approximately invariant to the set of global transformations of interest. Such a measure will be described below.

It is useful to characterize the difference between two images by two separate parameters. The first, d_{inv}, will be referred to as the invariant distance between the images, and is a measure of closeness of the images after compensating for all transformations of interest. The second is a measure of the amount of transformation needed to optimally align the images. This can be characterized by the vector $\bar{\lambda}$, which may include rotation angle, translation, etc. Such a set of transformations is implicit in the definition of d_{inv}.

Fig. 1a illustrates example images, all of which are considered to be distorted versions of the same object.



a. Global transformations of one signature.



b. Local distortions of one signature.

Fig. 1. (a) Different global transformations of a single image. These include rotation, scale and shear. The invariant distance is zero between any pair of images. (b) Images with local distortions in addition to the image shown in the upper left corner in (a). The invariant distance is small between any two images.

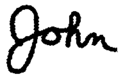
[1] These are transformations which apply equally to each point in an image. The analysis presented in this paper is, however, equally applicable to transformations which are not global in this sense, but may depend upon location.

The dot product, which measures the overlap of two images, would fail as a measure of closeness. On the other hand, the (rotation, scale, shear)-invariant[2] distance between each pair is zero. Fig. 1b illustrates another set of images which differ from those in Fig. 1a by the presence of local distortions. The invariant distance between any image in Fig. 1a and any image in Fig. 1b or between any two images in Fig. 1b is not 0, but is a small number, which measures an average size of the local distortion.

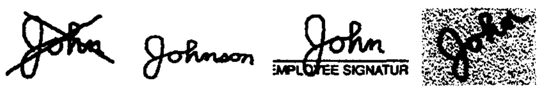
A property of d_{inv} which plays an important role in identifying images in the presence of noise, clutter or camouflage is that it is not symmetric:

$$d_{inv}(I_1, I_2) \neq d_{inv}(I_2, I_1). \tag{1}$$

The distance from image 1 to image 2 is not the same as that from image 2 to image 1; d_{inv} is a directed distance. $d_{inv}(I_1, I_2)$ actually measures the extent to which image 1 is a subset of, or is contained in, image 2 after compensating for the global invariances. While this is not an essential property for an invariant distance measure, we have designed d_{inv} to be asymmetric to enhance its utility for image recognition. For example, Fig. 2 illustrates a prototype image and a set of images which are essentially the same except for the addition of noise, clutter or additional image pieces. For the application of signature spotting, we could use d_{inv}(prototype, image) to measure the degree to which the prototype is contained in the image. In contrast to the use of a symmetric distance, this directed distance measure is zero for all the images in the figure. Note that if one were interested in recognizing an object which may have pieces missing, then one could reverse the direction and calculate d_{inv}(image, prototype), but at the expense of becoming more sensitive to additive noise and overlapping features.



a. Prototype image.



b. Images with clutter, additional images and noise added.

Fig. 2. The directed distance from (a) the prototype to (b) each image is 0 despite the presence of additional noise, clutter or additional word pieces.

The remainder of the paper will develop these concepts and illustrate them with a problem in handwritten signature recognition. Section 2 will develop an approximation to d_{inv} and to $\vec{\lambda}$ which is implemented by a simple algorithm. Results are shown for several transformations. A comparison is made with a complementary approach, based on small distortions of smooth gray-level images, which was developed by Simard, LeCun and Denker in [1]. Their approach is limited to small transformations; insensitivity to global transformations is achieved at the expense of smoothing the image. The larger the smoothing scale length, the larger is the range of insensitivity to the global transformations. However, this comes with a corresponding loss of detail in the image. The approach presented in this paper overcomes these limitations at the expense of the restriction to binary images. For gray-level images the algorithm can be applied to the binary image containing the edges. Section 3 will illustrate the insensitivity of the invariant distance under various image transformations. Section 4 will demonstrate the application to the problem of handwritten signature recognition. This is a problem which requires both the style-independent and the style-dependent features; a person's signature contains both his or her stylistic characteristics as well as the style-independent content of the word. Section 5 presents results on a purely style-dependent problem, the recognition of writer from a set of documents containing handwritten Chinese text.

[2] This notation will be used throughout the paper to indicate in parentheses the transformations for which invariance is designed.

2 Theory and Model Development

The invariant distance will be defined in terms of a measure which is not invariant to global image transformations but is insensitive to small distortions. This distortion insensitive distance between two images will be designated as d_{ins}. The invariant distance can be defined to be the value of d_{ins} obtained after having first optimally transformed one image to best match the other image. Let $T_{\vec{\lambda}}$ represent an operator which acts on an image I and induces a set of transformations parametrized by $\vec{\lambda}$. Then

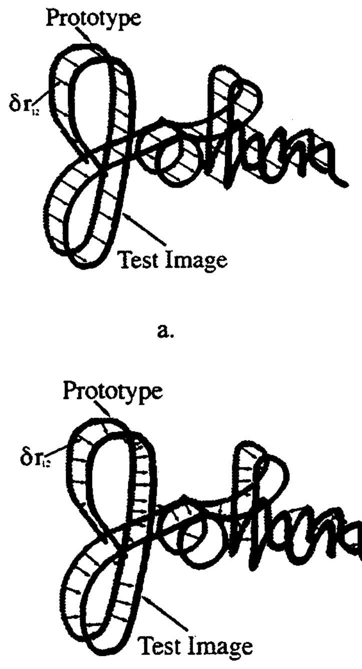
$$d_{inv}(I_1, I_2) = d_{ins}(T_{\vec{\lambda}} I_1, I_2), \tag{2}$$

where $\vec{\lambda}_0$ is the set of transformation parameters which minimize the above distance.

Calculation of the optimum transformation is in general a difficult problem. However, an approximation to the invariant distance will be developed which can be implemented as a simple and fast algorithm. This and the description of the distortion insensitive distance itself are described below.

2.1 The Distortion Insensitive Distance

We begin this section with a definition of a distortion insensitive distance measure which applies to binary (black and white) images. The distance from image I_1 to image I_2 will be designated $d_{ins}(I_1, I_2)$, and is defined to be the root mean square distance of all of the vectors which originate on a black pixel in image 1 and terminate on the corresponding black pixel in image 2. This is illustrated in Fig. 3 below.



b.
Fig. 3. Illustration of the ideal (a) and approximate (b) distortion insensitive distance. Arrows represent vectors which point from the black pixels in the prototype image to corresponding points in the test image. The distance is the root mean square of these vectors.

Let $\vec{r}_1^{\,a}$ represent the coordinate of a black pixel contained in image 1 and $\vec{r}_2^{\,a}$ represent the coordinate of a black pixel in image 2. Then each image can be described by the collection of vectors:

$$I_1 = \left\{\vec{r}_1^{\,a}\right\}_{a=1,2..N_1}, \qquad (3)$$

where N₁ is the number of black pixels in image 1, and similarly for image 2. A global transformation of the image is the set of transformed vectors:

$$T_{\vec{\lambda}} I_1 = \left\{T_{\vec{\lambda}}\, \vec{r}_1^{\,a}\right\}_{a=1,2...N_1}. \qquad (4)$$

If image 2 is identical to the transform of image 1 as induced by the transformation operator $T_{\vec{\lambda}}$, then each point in image 2 has a corresponding point in image 1, as defined through the following equality:

$$T_{\vec{\lambda}}\, \vec{r}_1^{\,a} = \vec{r}_2^{\,a}. \qquad (5)$$

Give such a correspondence, the distortion insensitive distance can then be defined as

$$d_{ins}^{\;2} = \frac{1}{N_1}\sum_{a=1}^{N_1}\left(\vec{r}_1^{\,a} - \vec{r}_2^{\,a}\right)^2 = \frac{1}{N_1}\sum_{a=1}^{N_1}\left(\delta \vec{r}_{12}^{\,a}\right)^2, \qquad (6)$$

where $\delta\vec{r}_{12}^{\,a} = \vec{r}_1^{\,a} - \vec{r}_2^{\,a}$. d_{ins} is the root mean square deviation vector between the two images. It is clear that this distance vanishes when I_2 is identical to I_1. Furthermore, if I_2 is almost identical to I_1 but differs slightly through small distortions of the black pixel locations, then the distance will be a small number which measures the root mean squared deviation of the black pixels from their locations in image 1.

In general, one does not have the convenience of comparing two images for which there is a one-to-one correspondence between black pixels. The general situation is not one in which the image to be compared is obtained from the comparing image solely by a global coordinate transformation. For this situation, we will maintain (6) as the definition of the distortion insensitive distance, but with $\vec{r}_2^{\,a}$ identified as the nearest black pixel[3] to $\vec{r}_1^{\,a}$, the ath pixel in image 1:

$$\vec{r}_2^{\,a} = \text{the point } \vec{r} \text{ in } I_2 \text{ with the smallest value of } \left(\vec{r} - \vec{r}_1^{\,a}\right)^2. \qquad (7)$$

As illustrated in Fig. 3b, this provides an approximation to the distance measure in (6). The approximation is good to the extent that nearest pixels are close in Euclidean distance to the corresponding image pixels. This distance is a variation upon the directed Hausdorff distances [2] and is implemented efficiently using a dynamic programming algorithm [3].

2.2 The Transformation Invariant Distance

In the ideal case, we can find a transformation, $T_{\vec{\lambda}}$, which transforms each vector in I_1 into the corresponding vector in I_2. Applying this transformation to I_1 gives a distortion insensitive distance of zero if one image is a pure transformation of the other. We will define this resulting distance to be the transformation invariant distance:

$$d_{inv}^{\;2} = \frac{1}{N_1}\sum_{a=1}^{N_1}\left(T_{\vec{\lambda}_0}\, \vec{r}_1^{\,a} - \vec{r}_2^{\,a}\right)^2, \qquad (8)$$

where $\vec{\lambda}_0$ is the set of parameters which minimize this distance. As above, in the non-ideal situation for which one does not have a one-to-one correspondence between black pixels in the two images, we will use this definition with the points in image 2 chosen as the closest points to each point in image 1.

[3] If there is more than one point with the same minimum distance, an arbitrary criterion can be used to select one. Note that the word point will be used throughout the paper to mean a black pixel with coordinates x and y.

In general, this distance will differ from zero even when image 2 is obtained from image 1 by a global transformation because closest points are not always corresponding points. But to the extent that a majority of points in image 1 have closest points in image 2 which are nearby to the corresponding points, this is a useful approximation. The utility of this definition is borne out by experiments and will be demonstrated in Sections 3 and 4 below.

Section 2.3 will derive closed form solutions for the optimal transformation parameters based on moments of the separation vectors $\delta \vec{r}_{12}{}^{a}$. Because of the approximation involved in replacing corresponding points by closest points, the transformation found will, in general, only be an approximation. A method for improving the approximation is to iterate the procedure with the following algorithm:

1) $\vec{\lambda}_0$ is calculated using the moments of $\delta \vec{r}_{12}{}^{a}$;
2) all points in image 2 are transformed using the estimate of $\vec{\lambda}_0$.
3) new separation vectors are now calculated and the invariant distance is calculated;
4) repeat steps 1-3) several times.

2.3 Derivation of Transformation Parameters

The utility of the above definition of the transformation invariant distance stems from the simplicity with which the transformation parameters can be calculated. There are a wide range of transformations which yield a simple algebraic formula for the optimum transformation parameters for distorting one image into another. The general problem is to find the set of parameters $\vec{\lambda}_0$ which minimize (8). To simplify the notation, we will write $<\ldots>_1$ to designate the average over black pixels in image 1, with the superscript 'a', over which the average is carried, understood. Then, differentiating (8) with respect to $\vec{\lambda}_0$, we obtain

$$0 = \frac{\partial d^2_{inv}}{\partial \vec{\lambda}_0} = 2\left\langle \left(T_{\vec{\lambda}_0}\vec{r}_1 - \vec{r}_2\right)\cdot \frac{\partial T_{\vec{\lambda}_0}}{\partial \vec{\lambda}_0}\vec{r}_1 \right\rangle_1 . \quad (9)$$

A closed form solution can be found when the transformation is a polynomial function of the coordinates. The simplest situation is illustrated by a linear transformation. Consider first a pure translation:

$$T_{\delta \vec{r}}\, \vec{r}_1^{\,a} = \vec{r}_1^{\,a} - \delta \vec{r}. \quad (10)$$

Equation (9) then gives an estimate of the overall translation as the average over points in image 1 of the vector which points to the nearest point in image 2:

$$\delta \vec{r} = \left\langle \vec{r}_1 - \vec{r}_2 \right\rangle_1 = \left\langle \delta \vec{r}_{12} \right\rangle_1 . \quad (11)$$

This solution can be used to translate image 2, and simply corresponds to aligning the centroids of the two images. The resulting value of (8) is the squared translation invariant distance. After transforming image 2, the points which were originally put in correspondence with points in image 1 will in general change. But if this is ignored, an approximate evaluation of the squared translation invariant distance is obtained as

$$d_{inv}{}^2 = \left\langle \delta \vec{r}_{12}{}^2 \right\rangle_1 - \left\langle \delta \vec{r}_{12} \right\rangle_1{}^2 . \quad (12)$$

To this level of approximation, the translation invariant distance is simply the standard deviation of the nearest point displacement vectors from image 1 to image 2. In other words, for a pure translation all of the displacement vectors should be of the same length and point in the same direction. The invariant distance measures the extent to which this is not true, and hence the extent to which image 2 is not simply a translation of image 1.

As a second example, consider a pure rotation:

$$T_{\theta}\vec{r}_1^{\,a} = \vec{R}(\theta)\cdot \vec{r}_1^{\,a} = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix} \vec{r}_1^{\,a} . \quad (13)$$

The resultant solution to (9) is

$$\tan\theta = \frac{-\left\langle \vec{r}_1 \times \vec{r}_2 \right\rangle_1}{\left\langle \vec{r}_1 \cdot \vec{r}_2 \right\rangle_1}, \quad (14)$$

where we define the cross product in two dimensions as the following scalar:

$$\vec{r}_1 \times \vec{r}_2 \equiv x_1 y_2 - y_1 x_2 . \quad (15)$$

Repeating the procedure which led to (12) we obtain an approximation to the rotation invariant squared distance:

$$d_{inv}{}^2 = \left\langle \vec{r}_1{}^2 \right\rangle_1 - 2\sqrt{\left\langle \vec{r}_1 \cdot \vec{r}_2 \right\rangle_1{}^2 + \left\langle \vec{r}_1 \times \vec{r}_2 \right\rangle_1{}^2} + \left\langle \vec{r}_2{}^2 \right\rangle_1 . \quad (16)$$

This is a distance which vanishes whenever image 2 is a pure rotation of image 1, to the extent that the correspondence approximation is valid.

In general any affine transformation can be incorporated. In addition, many nonlinear transformations can be included that still give rise to analytic solutions for the deformation parameters and the invariant distance. It is instructive to consider a general transformation linear in the distortion parameters, which mây also be the linear approximation to a nonlinear transformation:

$$T_{\vec{\lambda}} = 1 + \delta\vec{\lambda} \cdot \frac{dT}{d\vec{\lambda}}. \tag{17}$$

Equation (9) then has the solution

$$\delta\vec{\lambda} = \left\langle x_{1\vec{\lambda}} x_{1\vec{\lambda}} + y_{1\vec{\lambda}} y_{1\vec{\lambda}} \right\rangle_1^{-1} \cdot \left\langle (x_2 - x_1) x_{1\vec{\lambda}} + (y_2 - y_1) y_{1\vec{\lambda}} \right\rangle_1, \tag{18}$$

where

$$x_{1\vec{\lambda}} = \frac{dT}{d\vec{\lambda}} x_1 \quad , \quad y_{1\vec{\lambda}} = \frac{dT}{d\vec{\lambda}} y_1 , \tag{19}$$

and the matrix inverse and multiplication are with respect to the indices labeled by $\vec{\lambda}$. The invariant distance can be expressed in terms of a projection operator:

$$\vec{\Pi}_{ab} = \vec{1}_{ab} - \frac{1}{N_1} \vec{r}^{\,a}{}_{1\vec{\lambda}} \left\langle x_{1\vec{\lambda}} x_{1\vec{\lambda}} + y_{1\vec{\lambda}} y_{1\vec{\lambda}} \right\rangle_1^{-1} \vec{r}^{\,b}{}_{1\vec{\lambda}} , \tag{20}$$

$$\left(\vec{\Pi} \cdot \vec{\Pi}\right)_{ab} = \vec{\Pi}_{ab}, \quad \left(\vec{\Pi} \cdot \vec{r}_{1\vec{\lambda}}\right)^a = 0 , \tag{21}$$

$$d_{inv}{}^2 = N_1 \left\langle (\vec{r}_1 - \vec{r}_2)^a \cdot \vec{\Pi}_{ab} \cdot (\vec{r}_1 - \vec{r}_2)^b \right\rangle_{1,ab} . \tag{22}$$

The average indicated above is over both indices a and b which label points in image 1. If the sets of coordinates $\left\{\vec{r}^{\,a}{}_{1\vec{\lambda}}\right\}_{a=1 \ldots N_1}$ are thought of as forming template images, one for each value of $\vec{\lambda}$, then the projection operator acts to project an image onto the subspace of images orthogonal to each template image. This enforces the vanishing of the invariant distance when one image is a linear transformation of the other:

$$d_{inv}{}^2 = 0 \quad \text{for}$$

$$x_2 = x_1 + \delta\vec{\lambda} \cdot x_{1\vec{\lambda}} \quad , \quad y_2 = y_1 + \delta\vec{\lambda} \cdot y_{1\vec{\lambda}} . \tag{23}$$

Equation (22) can also be written as

$$d_{inv}{}^2 = \left\langle (\vec{r}_1 - \vec{r}_2)^2 \right\rangle_1 - \left\langle (\vec{r}_1 - \vec{r}_2) \cdot \vec{r}_{1\vec{\lambda}} \right\rangle_1 \cdot I^{-1}{}_{1\vec{\lambda}\vec{\lambda}'} \cdot \left\langle \vec{r}_{1\vec{\lambda}'} \cdot (\vec{r}_1 - \vec{r}_2) \right\rangle_1 \tag{24}$$

with summation understood for the repeated indices λ and λ', and $I_{1\vec{\lambda}\vec{\lambda}'}$ is defined as

$$I_{1\vec{\lambda}\vec{\lambda}'} \equiv \left\langle x_{1\vec{\lambda}} x_{1\vec{\lambda}'} + y_{1\vec{\lambda}} y_{1\vec{\lambda}'} \right\rangle_1 . \tag{25}$$

Each average over black pixels in image 1 can be represented equivalently as a dot product between binary image 1 and an appropriate gray-level image derived from image 2. For example,

$$\left\langle (\vec{r}_1 - \vec{r}_2)^2 \right\rangle_1 = I_1 \cdot I_{2,d^2} \tag{26}$$

is the dot product of image 1 with I_{2,d^2}, the distance-squared transform (as described in [4]) of image 2, normalized by N_1. Defining

$$\left\langle (\vec{r}_1 - \vec{r}_2) \cdot \vec{r}_{1\vec{\lambda}} \right\rangle_1 \equiv I_1 \cdot I_{2,\vec{\lambda}} , \tag{27}$$

the squared invariant distance can be written as a nonlinear function of these dot products:

$$d_{inv}{}^2 = I_1 \cdot I_{2,d^2} - \left(I_1 \cdot I_{2,\vec{\lambda}}\right) \cdot I^{-1}{}_{1\vec{\lambda}\vec{\lambda}'} \cdot \left(I_1 \cdot I_{2,\vec{\lambda}'}\right) . \tag{28}$$

The images $I_{2,\vec{\lambda}}$ can be thought of as moments, which depend upon $\vec{\lambda}$, of the vector distance transform of image 2, where the vector distance transform of an image is the pair of x and y component images whose pixel intensity values are the coordinate displacements along x and y axes to the nearest black pixel in the image.

If we desire invariance under an n-dimensional transformation parametrized by λ_i, $i = 1,2,\ldots n$, then from each image I_2, n+1 new images are derived whose dot products with image I_1 are used to form the directed invariant distance. This distance measures the extent to which image 1 is a subset of image 2. In Section 4 image 1 will be considered as a prototype image and image 2 as the test image. Other applications may require the reverse situation for which image 2 is the prototype and image 1 is the test image.

2.4 Comparison with Tangent Distance

Simard, LeCun and Denker in [1] have developed a complementary approach based on the intensity values of gray-level images rather than on the locations of black pixels in a black and white image. Their analog of the transformation invariant distance is the tangent distance, which is a measure of the overlap between two images after having adjusted for the effect of an infinitesimal transformation. The heart of the approach lies in approximating a transformed image under the transformation described in (17) by the first term in a Taylor series expansion:

$$I(T_\lambda \vec{r}) \cong I(\vec{r}) + \delta\vec{\lambda} \cdot \frac{dT}{d\vec{\lambda}} (r \cdot \nabla) I(\vec{r}) \tag{29}$$

The second term above can be considered to be a linear combination of template images:

$$I'_{\vec{\lambda}}(\vec{r}) = \frac{dT}{d\vec{\lambda}} (r \cdot \nabla) I(\vec{r}) . \tag{30}$$

The approximation is seen to be valid only for images which are sufficiently smooth so that the gradient image exists. This is in general not true, but can be remedied by smoothing the image sufficiently. One must trade off smoothing the image to maintain invariance to small transformations with loss of information from smoothing too much.

The tangent distance between two images has been defined symmetrically. For our purposes, however, it is most instructive to consider a variant of the distance which is asymmetrically defined as

$$d^2{}_{\text{tangent}} = \int (d\vec{r})\left(I_2(\vec{r}) - I_1(\vec{r}) - \delta\vec{\lambda}_0 \cdot I_{1\vec{\lambda}_0}(\vec{r})\right)^2, \tag{31}$$

where $\delta\vec{\lambda}_0$ is the parameter set which minimizes this squared distance. By analogy with the derivation in Section 2.3, the minimization problem has the solution

$$\delta\vec{\lambda}_0 = \left(\int (d\vec{r}) I_{1\vec{\lambda}}(\vec{r}) I_{1\vec{\lambda}}(\vec{r})\right)^{-1} \cdot \int (d\vec{r}\,') I_{1\vec{\lambda}}(\vec{r}\,')\left(I_2(\vec{r}\,') - I_1(\vec{r}\,')\right), \tag{32}$$

and the squared tangent distance can be written as

$$d_{tamgent}{}^2 = (I_2 - I_1)^2 - \left(I_1 \cdot I'_{2,\vec{\lambda}}\right) \cdot I^{-1}{}_{1\vec{\lambda}\vec{\lambda}'} \cdot \left(I_1 \cdot I'_{2,\vec{\lambda}'}\right), \tag{33}$$

with the dot product used to denote spatial integration and the prototype image, $I'_{2,\vec{\lambda}}$, defined by

$$I'_{2,\vec{\lambda}} = -\vec{\nabla} \cdot \left(I_2(\vec{r}) \frac{dT}{d\vec{\lambda}} \vec{r} \right). \tag{34}$$

In spite of the similarities, there are some key differences between the two methods. The invariant distance in (28) is the average distance (in physical coordinate units) that the object or objects in image 1 are from those in image 2. On the other hand, the tangent distance in (33) is a measure of intensity difference between the two images. In addition, the invariant distance approach relies on a coordinate description of an image. As such, it is most suitably adapted to binary images. For gray-level images, it is natural to use a binary image of object boundaries, obtained for example from an edge detection or watershed algorithm. The complementary tangent distance approach compares gray-level intensity images directly but requires the smoothing out of any sharp intensity changes in order to calculate the gradient image. This yields insensitivity to image distortions which are of the same magnitude as the smoothing scale length. Invariance to large transformations is only obtained at the expense of large amounts of smoothing, which can remove important details in an image. In contrast, the coordinate description approach does not require smoothing the image, and is approximately invariant to large transformations, as is shown in Section 3.

2.5 Implementation of the Algorithms

There are several approaches for implementing the invariant distance measure when building invariance to more than one global transformation. For example, invariance with respect to translation, rotation, scale and shear can be built in one step using (22). This is approximately equivalent to using (18) to find the distorting affine transformation parameters, using these values to undistort the image, and calculating the resultant distortion insensitive distance using (6). (22) will only give an approximation of this resultant distance because closest pixel identifications may change after transforming the image. Similarly, the transformation found in (18) will only be approximate. In general, the closer the two images are, the better is the approximation. A useful procedure is to iterate the transformation algorithm: the transformation parameters are calculated and the image is undistorted according to these parameters. Then new transformation parameters are calculated based on the new image, and the image is undistorted again. When iterated, this procedure has been shown to converge, in practice, for a wide range of images from handwritten characters, handwritten words and two-dimensional images of three-dimensional objects. This procedure is demonstrated in Section 3.

An alternate approach to calculating the full set of transformation parameters in one step is to calculate each transformation separately. That is, we could calculate the net translation first and then translate the image. Then we could calculate the rotation angle and rotate the image, .. and so on. It is an open question as to which procedure converges more quickly.

An effective implementation of the algorithm requires overcoming certain difficulties associated with the breakdown of the basic approximation assumed, the equality of closest pixels with corresponding image points. The following are special, somewhat pathological cases which illustrate the extremes of these difficulties:

Suppose the fit is almost exact. Then the computed translation is effectively zero, and convergence to the correct position may be very slow. A trivial example of this is shown in Fig. 4. Here the prototype, shown shifted vertically for clarity, is a single pixel off of the image. Since all pixels in the prototype overlay their nearest neighbors except for the one on the end, the computed translation in pixel units is just 1/N, where N is the number of pixels in the line. But the correct value is 1 pixel. Convergence is slow.

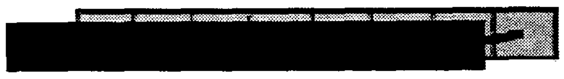


Fig. 4. Translation convergence. The prototype, shown in light gray, almost overlays the image which is shown in dark gray. Each image consists of 8 pixels in a line. The prototype is offset from the test image by 1 pixel.

Our solution is to apply a lower limit to the translation: if it is less than a pixel, we translate a full pixel in the indicated direction.

On the other hand, suppose the prototype does not overlap the image at all, and the image is such that one of its pixels is the nearest neighbor to every point in the prototype. Fig. 5a shows an example of this situation. Here the computed translation will center the prototype on the common nearest neighbor, and the computed scale factors will shrink the prototype to a single pixel. By performing the translation step before the scaling we avoid this problem completely.

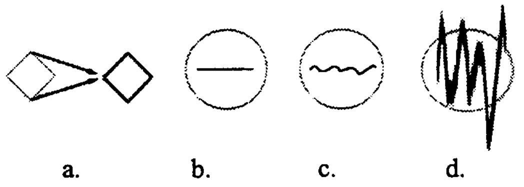


Fig. 5. (a) Degenerate scaling. Every point on the prototype, shown in gray, sees the same nearest neighbor on the image which is shown in black. (b) The prototype will shrink to a line. (c) The prototype will shrink to a flat object. (d) On the scale of the original prototype the resultant distance after shrinking in (c) is large.

However, a related problem occurs if a small object is being compared with a large prototype as in Fig. 5b. If, as in the figure, the object is a perfectly straight line, a vertical scaling which reduces the prototype circle to essentially a straight line will give the smallest resultant distance. This is prevented by applying a lower limit to the scale factors. A slightly more realistic situation is shown if Fig. 5c, for which the small object is not perfectly straight. In this situation, the circle would again be scaled down to be very thin. As it stands, this would result in a small scale invariant distance which is misleading. This is easily remedied by always comparing distances relative to the prototype size, which is equivalent to scaling the object up to the size of the prototype, as shown in Fig. 5d. This rescaled distance is large, and hence there is no confusion that the two objects are of similar shape.

3 Sensitivity Analysis

As pointed out in the Section 2.5, the implementation of the transformation invariant distance depends upon the assumption that closest points are corresponding points. The breakdown of this assumption leads to the approximate nature of the invariance to global transformations. The purpose of this section is to illustrate the extent to which this approximation still leads to a useful measure of distance. We will illustrate that the invariant distance between two images, one of which has been transformed with respect to the other, remains small even for large transformations. The sensitivity to the amount of transformation depends upon the number of transform iterations taken, as was discussed in Section 2.5.

Figs. 6a and b show a prototype signature and a test signature which have been severely sheared with a shear value of 1.5. This distortion corresponds to parts of the image shifting by as much as 85 pixels or roughly half the width of the 166 wide by 113 high prototype image. The result of iterating the shear-invariant distance algorithm is to distort the prototype more and more to match the test image. The signature produces a good match after 10 iterations and an almost perfect match after 30 iterations.

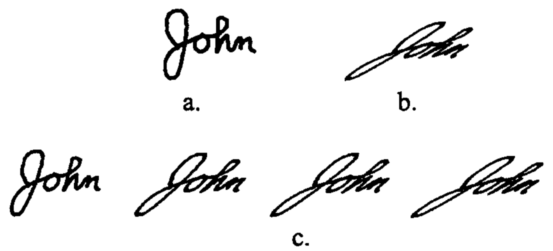


Fig. 6. Illustration of prototype transformation under several iterations of the algorithm. (a) depicts the original prototype signature. (b) is the test signature. In (c) the transformed prototype is shown from left to right in sequence after 1, 5, 10 and 30 iterations.

Fig. 7 shows the range of insensitivity of the shear-invariant distance for a range of shears of the signature depicted in Fig 6.

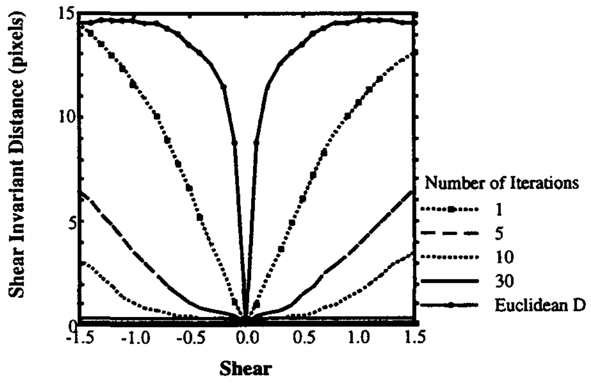


Fig. 7. Insensitivity of the shear-invariant distance to shear for various iterations of the algorithm.

With 10 iterations of the algorithm the distance remains less than 1 pixel for shears up to a magnitude of approximately 1.0. With 5 iterations, the distance remains less than the average width of the lines in the image (5 pixels) over the same range of shear values. The sensitivity decreases as more iterations are used. For comparison, a scaled Euclidean distance is shown to illustrate the high sensitivity of this metric under image transformations.

Results for rotation, with angles ranging from 60 to -60 degrees, and scaling, with scaling factors ranging from 0.5 to 1.5, are similar to the above. Detailed presentation of these results as well as those for combinations of the various distortions will be presented in a future paper [5].

4 Handwritten Signature Recognition

The invariant distance measure has been tested on the problem of offline handwritten signature recognition. The goal is to search a database of documents to identify those documents containing a particular writer's signature, given one or more examples of the signature. We use the invariant distance both to locate and to classify the signature.

4.1 The Database

For this test, we generated a database of signatures by digitizing the employee signature section of the weekly timecards submitted by some of our fellow employees. The database contains the timecards from 45 employees, each with 51 to 54 timecards, signed at weekly intervals. The total sample size is 2384. By spreading the signatures over a large period of time, as opposed to having each person sign his or her name dozens of times in succession, we have produced a set of signatures containing a range of variations such as different pens, different amounts of muscle fatigue and different signing speeds.

A typical timecard document is shown in Fig. 8. The signature has been modified in order to maintain confidentiality of the actual signatures. The timecards have a gray region in the upper left which, when digitized to black and white, produces a speckled pattern. There are also obvious lines and typewritten text present, as well as several other handwritten fields. Each sample was digitized in black and white at 100 dots per inch.

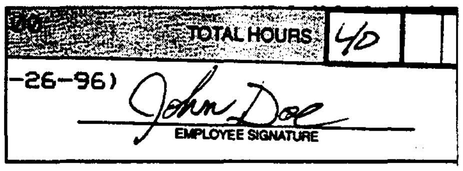

TOTAL HOURS 40
-26-96) John Doe
EMPLOYEE SIGNATURE

Fig. 8. Sample timecard from the database.

Noise removal algorithms were first used to clean up each image. While the directional nature of the feature extraction algorithm was designed to allow recognition in noise, the removal of as much noise as possible eliminates potential regions of the image which must be searched, thus reducing evaluation time as well as the false alarm rate. In addition we apply a baseline adjustment algorithm to initially align the prototype and image principal axes.

4.2 Results

A series of numerical experiments were carried out using a reduced data set of 1857 documents from 35 writers. In the first set of experiments, a single prototype signature from each of seven writers was chosen from the database for evaluation. These signatures are representative of the variation seen across the database, but do not necessarily encompass the full statistical variation expected from an ensemble of writers. These seven signatures were used to search for matching signatures in the total set of 1857 documents. The documents were cleaned by hand to remove non-signature portions, though similar results have been obtained on the raw documents with automated noise removal.

For each search to find matches to the prototype signature image, 12 parameters were calculated. These included the (rotation, scale, shear)-invariant distance from the prototype to the image, the distance between the x profiles (integrated intensity along the y axis as a function of x), the distance between the y profiles, the first moments of the image and several ratios between the prototype and the image. The ratios included aspect ratio and the second, third and fourth moments of the x and y profiles.

A feedforward neural network with 10 hidden layer nodes was trained to determine, for each signature document tested, the probability that it is a signature from the same writer as the prototype signature. Half of the data was used for training and the other half for testing. The two halves were exchanged, the training repeated, and the results averaged over both combinations. Results are presented in Table 1 using several combinations of features all derived from one prototype signature. Combination C in the table used only the invariant distance to identify signatures. B included the x and y profile distances and the ratio of aspects in addition to the invariant distance. Combination A included the first moment and the ratios for the second through fourth moment of the x and y profiles, the x and y profile distances, the ratio of aspect ratios and the invariant distance.

Table 1: Signature classification rate for three choices of feature sets with the reduced dataset of 35 writers/1857 documents.

Feature Combination	**Mean**	**Standard Deviation**
A: all features	94%	5.1%
B : d_{inv} + profiles, aspect	90%	9.8%
C: d_{inv} alone	81%	14%

The total percent correct, which is 100% minus the sum of the false positive and the false negative errors, is given. It is calculated at the threshold probability level which minimizes the sum. A false positive is defined to be an image which is falsely classified as a positive signature match, while a false negative is an image which is falsely classified as a negative signature match. The data is averaged over the seven signatures from different writers. For each signature, four prototypes were selected to run four separate database searches. The results are averaged over all four prototypes. Both the mean and the standard deviation of the 35 values for the individual writer classification rates are shown. The large spread in the results is reflective of the large variation in performance from signature to signature. Several signatures achieved close to perfect performance. Two out of the seven had particularly poor performance.

A more extensive set of experiments was carried out on the full set of 2384 documents. In contrast to the previous experiments in which the feature set involved comparison of the search image with each test image, we developed a set of features for these experiments which are intrinsic properties of each test image. These are obtained by choosing a set of signatures from the database to serve as prototype images. Every other image is then analyzed in reference to the prototype. The prototype images then serve as a basis set for describing any image. For this experiment we have chosen a basis set of six prototypes, one sample from each of six writers. We represent every image by a twenty four dimensional feature vector consisting, for each of the six prototypes, of the invariant distance between the image and each prototype, the root mean squared distance between the two x profiles, the root mean squared distance between the two y profiles, and the ratio of the two aspects ratios.

Having represented every image by twenty four numbers, we then use a nearest neighbor classification scheme to choose the closest matches to a given signature image. We define a correct match to occur when the actual writer is the same as at least one out of the top n closest images, with n ranging in our experiment from 1 to 9. We have used a Euclidean distance measure with a separate weighting constant for each feature dimension. These weights were determined by a genetic algorithm/neural network combination optimized to find the set of weights that minimizes the fraction of misclassification errors.

With n=9, we show the resultant classification rate in Table 2. Results are shown with and without the use of the invariant distance metric.

Table 2: Signature classification rate with the full data set of 54 writers/2384 documents.

Feature Combination	Mean	Standard Deviation
A: all features	96%	6.3%
B : without d_{inv}	93%	7.3%
C: d_{inv} alone	80%	16%

Combination A in the table includes all 24 features. Combination B excludes the 6 invariant distance measures. Combination C uses only the 6 invariant distance measures. These results are similar to the neural network results on the reduced data set reported in Table 1. They also indicate that the incorporation of the invariant distance metric reduces the fraction of errors by roughly 3%. The statistical uncertainty in the classification rates is approximately 0.8%. Clearly the majority of the variation described by the standard deviation arises from the large writer to writer variation.

In Fig. 9 the classification rates are shown for each of the 45 individual writers, sorted from worst to best performer. This sort can be used to rank the writing variability and consistency. The worst performers tend to have highly variable signatures. If one considers several signatures from such a writer, they appear to the human observer to be written by several different writers rather than a single writer. In contrast, the good performers in the figure have handwriting which is very consistent and is easy to identify as belonging to the same writer.

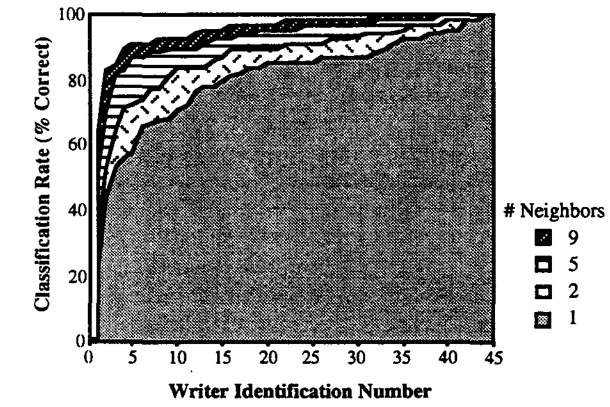


Fig. 9. Classification rates for each of the 45 writers.

Fig. 10 shows a histogram of the classification rates for n equal to 1, 2 and 9, obtained from the above figure by binning the data in classification rate bins, each 10% wide.

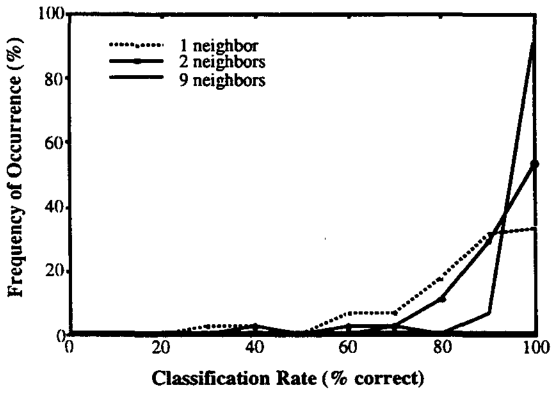


Fig. 10. Histogram of classification rates for selection of the closest 1, 2 and 9 nearest neighbor images.

As the number of closest images used ranges from 1 to 9, the histogram peaks more and more toward the higher classification rates, as expected.

5 Chinese Writer Recognition

The objective of this problem is to separate a large number of handwritten documents according to writer. As a step toward this goal, we seek a set of features that can be computed for each document such that the nearest document in this feature space is written by the same writer. Once the set of features has been chosen, a set of weights for these features is chosen which minimizes the distance between documents by the same writer and maximizes the distance between documents by different writers.

A database of 106 Chinese documents from 15 writers has been generated by the Department of Defense (DoD). Pieces of two example images are shown below in Fig 11. Sample characters of each writer are shown in Fig. 12.

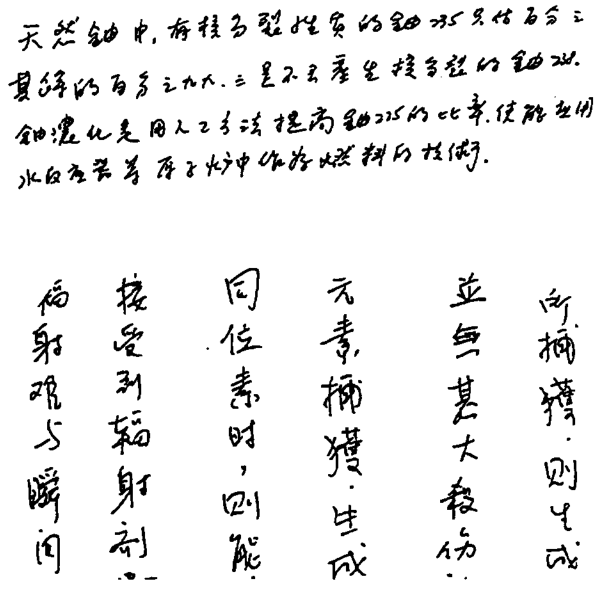


Fig. 11. Example Chinese documents from two different writers in the DoD database.

For each document in the database we group connected components into characters. We then calculate a set of features for each character from which we obtain the mean value and standard deviation of each across the set of characters in the document. The features are used with a nearest neighbor classifier to evaluate the writer identification results. That is, given a new document to test, we evaluate its feature vector and search for the n closest feature vectors. The identify of the writer for the test document is assumed to be that of one of the n closest matching documents. We elaborate these concepts and show these results for the DoD database below.

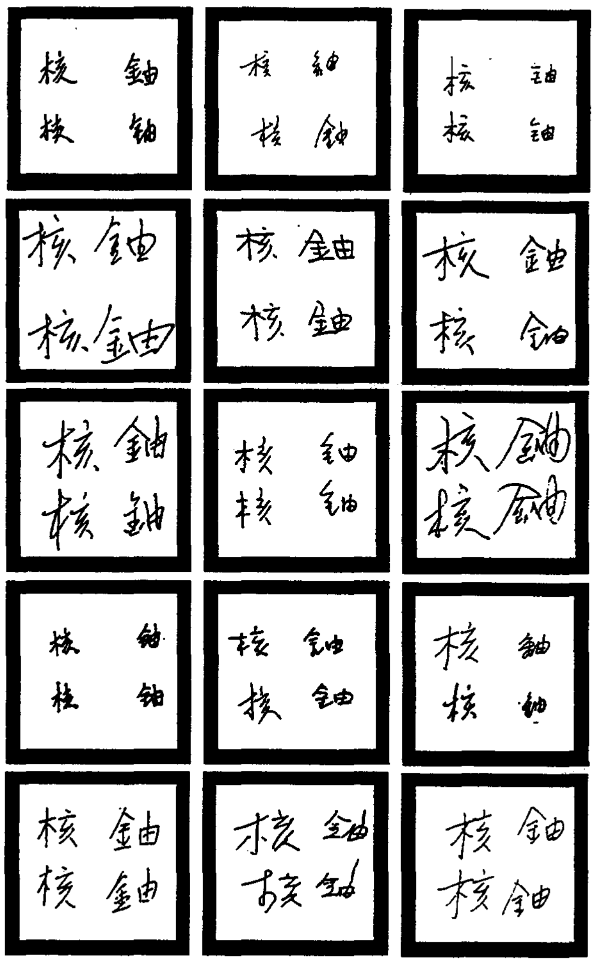


Fig. 12. Sample characters from each of the 15 writers in the Chinese document database. The samples are ordered according to writer identification numbers 1 to 15, with the top row beginning with 1, 2 and 3 from left to right and continuing in the same order to the next rows.

5.1 Character Grouping

Each black and white text image was segmented into a collection of connected black areas or connected components. Any component with fewer than three black pixels was discarded as being due to image noise. The expected height and width of the characters on the page were calculated as the 97th percentile values of the height and width distributions of these components.

The components were then grouped into characters as follows. First, all components smaller than 1/4 the expected character size and more than 1/4 the expected character size away from other components are classified as punctuation marks or noise and removed. A box with the expected character's size is drawn around the center of the largest component not yet tagged as being part of a character.

We define an error measure which describes how well this box fits the character containing this component. The error is defined as a ratio of two component areas, where we define component area as the number of black pixels in the connected component. We define A_1 to be the area of all of the connected components within the box which have not been assigned to another character. A_2 is defined to be the total area of all connected components which have some overlap within the box whether or not they have already been assigned to a different character. Then the error measure is defined to be the ratio A_2/A_1. By shifting the box around so that this error is minimized we obtain a grouping of components that belong to a single character.

All components with more area within this box than outside it are grouped into a character if they have not already been assigned to another character. A second box is drawn around all of the components in the new character. Again, all components with more black pixels inside than outside of this box are tagged as being part of the character.

Sometimes a small component from a neighboring character is erroneously included. If a component in the new character is too isolated from the other components, it is removed.

5.2 Features

A number of features, 40 altogether, were computed for the components and characters. For each document, the ratio of the standard deviation to the mean was computed for the area, height, and width of the components:

$$r_{Area} = \frac{\sigma_{Area}}{\mu_{Area}} \qquad (35)$$

The standard deviation and mean of the aspect ratio (width to height ratio) of the components were also computed.

For the characters in the document, the same features were calculated as well as the standard deviation and mean of the number of components per character, of the horizontal spacing between characters divided by the mean character width, and of the vertical spacing divided by the mean character height. These features were designed to be dimensionless and independent of the resolution of the document image.

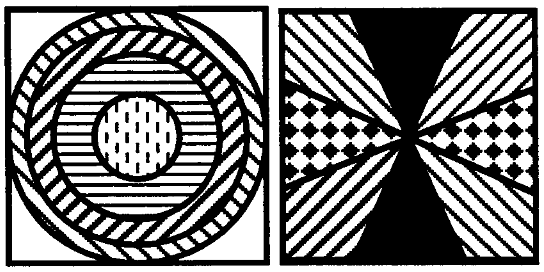


Fig. 12. Ring and wedge shaped regions over which the power spectrum of characters is summed.

In addition, texture features were computed based on the power spectrum of the characters. The power spectrum was computed from the Fourier transform of each character. The power spectrum was then summed over six rings of equal area and over four double-wedges of similar area. The shapes of these regions are illustrated in Fig. 12 above for four rings and four wedges.

The ring sums are insensitive to rotation but are sensitive to changes in scale. The reverse is true of the wedge sums, and so the two sets of texture features complement each other. The wedge regions are symmetric about the origin because the power spectrum is symmetric about the origin.

5.3 Results for Chinese Writer Identification

Classification results are shown in Table 3 for the DoD database of 106 Chinese documents written by 15 writers. Shown are the mean and standard deviation of the 15 individual classification rates. The individual rates are based on small sample sizes ranging from 6 to 30 documents each. The statistical uncertainty in the mean classification rate arising from this small sample size is approximately 1.4%. These results are based upon nearest neighbor classification with from n = 1 to 9 nearest neighbors selected. As with the handwritten signature problem, we identify a test document as correctly identified if any one of the n neighbors has the correct writer identity.

Table 3: Chinese writer classification rate.

Number of Neighbors Used For Classification	Mean	Standard Deviation
1	98%	4.7%
2	98%	4.7%
5	99%	3.7%
9	99%	3.7%

For the case with 1 nearest neighbor classification, there are only two errors. One document from writer 4 is incorrectly assigned to writer 2 and one document from writer 3 is incorrectly assigned to writer 5.

The numbers of documents in the database for each writer are listed in Table 4 below.

Table 4: Tabulation of number of documents for each writer for the Chinese document database.

Writer I.D.	Number of documents
1	10
2	7
3	7
4	8
5	7
6	7
7	6
8	6
9	7
10	5
11	8
12	8
13	7
14	6
15	7

Though the number of documents per writer is small in these experiments, the results are encouraging with average writer classification rates of 98% to 99% as the number of of nearest neighbors considered varies from n=1 to n=9. The variation in the individual writer classification rates is in the range of 4% to 5% (one standard deviation) and varies from a minimum of 86% for the most difficult-to-analyze writer to 100% for the easiest.

6 Conclusions

A new measure has been defined to calculate the distance between a pair of images. This distance is insensitive to small distortions of the image and is approximately invariant to a set of global coordinate transformations which can be selected to best solve a given image comparison problem. The distance has been shown to provide a good measure of image closeness for a wide range of image distortions. These include shear, with shear values in the range of -1.5 to 1.5, rotation, with angles in the range of -60 to 60 degrees, scaling, with scale factors in the range of 0.5 to 1.5 and translations to any distance.

The transformation invariant distance, combined with other features which capture size and shape information, has been applied to a problem in handwritten signature recognition with encouraging results. Results on the Logicon timecard database indicate a classification rate of from 81% to 96% for respectively 1 to 9 nearest neighbor classification, when averaged over 45 writers and a corresponding spread in the rate from writer to writer of 15% to 6%. The transformation distance gives an improvement of 3% over the use of the size and shape features alone.

A set of stylistic features was developed to identify the writer, independent of text content. 40 features were used to identify the writer on a set 106 Chinese documents written by 15 writers. Preliminary results indicate mean writer identification rates of 98% and 99% for 1 and 9 nearest neighbor classification respectively.

Acknowledgments

This work was supported by the Defense Advanced Research Projects Agency and the Office of Naval Research under contract N00014-93-C-0188. We wish to thank the Department of Defense for supplying the Chinese writer database and Narbik Manukian for his comments and assistance in reviewing the paper.

References

[1] P. Simard, Y. LeCun and J. Denker, Efficient pattern recognition using a new transformation distance, *Advances in Neural Information Processing Systems*, (Morgan Kaufman, San Mateo, CA 1993) 50-58.

[2] M.P. Dubuisson and A. K. Jain, A modified Hausdorff distance for object matching, *Proc. 12th Int. Conf. Pattern Recognition*, Jerusalem, Israel, (1994) 566-569.

[3] D. P. Huttenlocher, G. A. Klanderman and W. J. Rucklidge, Comparing images using the Hausdorff distance, *IEEE Trans. on Pattern Analysis and Machine Intelligence* **15**, no. 9 (1993).

[4] P. E. Danielsson, Euclidean distance mapping, *Comput. Graphics Image Processing* **14**, (1980) 227-248.

[5] G. Wilensky and R. Crawford, Development of the Transformation invariant distance for image comparison, preprint submitted to *IEEE Trans. on Pattern Analysis and Machine Intelligence* (1996).

A Language-Independent Methodology for OCR

John Makhoul, Richard Schwartz, Christopher LaPre, Issam Bazzi, Zhidong Lu, and Prem Natarajan

BBN Corporation, Cambridge, MA 02138 USA
makhoul@bbn.com

Abstract

We present a methodology for OCR that exhibits the following properties: language-independent feature extraction, training, and recognition components; no separate segmentation at the character and word levels; and the training is performed automatically on data that is also not presegmented. The methodology is adapted to OCR from continuous speech recognition, which has developed a mature and successful technology based on Hidden Markov Models. The language independence of the methodology is demonstrated using omnifont experiments on the DARPA Arabic OCR Corpus and the University of Washington English Document Image Database I.

1 Introduction

During the last two decades, a revolution has taken place in continuous speech recognition (CSR) technology. The earlier technology struggled to perform recognition using a presegmentation stage, where the speech was segmented into tentative phonetic units, followed by a recognition stage that employed a set of acoustic-phonetic rules that were written by hand. Furthermore, the training data was laboriously hand-segmented and labeled at the phonetic level. The technology was very language-dependent – new segmentation and recognition algorithms had to be written for each new language. In contrast, the new CSR technology, which is based on the use of hidden Markov models (HMM) [1] to model phonetic units, has proven to be language-independent and does not require presegmentation of the data, neither during training nor recognition. The improvements in this model-based CSR technology in the last decade have been truly phenomenal [2].

In this paper, we show how the same CSR technology based on HMMs can be adapted in a straightforward manner to the problem of optical character recognition (OCR) [3]. In fact, after a line-finding stage, followed by a simple feature-extraction stage, the system utilizes the BBN Byblos CSR system [4], with no modification, to perform the training and recognition. The whole system, including the feature extraction, training, and recognition components, are designed to be independent of the script or language used. The language-dependent parts of the system are confined to the lexicon, grammar or language model, and training data. Furthermore, this technology does not require presegmentation of the data at the character and word levels, neither for training nor for recognition.

We demonstrate the power of the new methodology using the DARPA Arabic OCR corpus, which consists of scanned text from a number of sources of varying quality. Arabic text is known to present special challenges to OCR because the characters are connected for the most part and the shape of each character changes depending on the neighboring characters. To demonstrate the script-independence of this methodology, we also report on experiments with the University of Washington (UW) English Document Database I, using the same basic system as the Arabic recognition system.

There have been a number of attempts to use HMMs in OCR [5,6,7,8,9,10,11]. This work presents a number of departures from other studies: (1) This may be the first attempt to use a CSR system directly to perform OCR; (2) it represents the first attempt to use HMMs for Arabic OCR [12]; and (3) the major components of the system (feature extraction, training, and recognition) are intended and designed to be language-independent, and have already been demonstrated in two different script families.

In Section 2, we present the general probabilistic methodology used, including a brief introduction to hidden Markov models and a description of our general OCR system. Section 3 details the various parts of the system, including the preprocessing and feature extraction stages, the HMM model structures used, and the training and recognition components. In Section 4 we describe a series of experiments performed on the Arabic OCR corpus and in Section 5 we present our initial results on English OCR.

2 Probabilistic Paradigm

2.1 Problem Setup

Given the scanned data from a line of text, our basic probabilistic paradigm attempts to find that sequence of characters C that maximizes $P(C|X)$, the probability of the sequence of characters C, given a sequence of feature vectors X that represents the input text. Using Bayes' Rule, we can write:

$$P(C|X) = P(X|C)\ P(C)\ /\ P(X).$$

We call $P(X|C)$ the *feature model* and $P(C)$ the *language model* (or grammar). $P(X|C)$ is a model of the input data for any particular character sequence C; $P(C)$ is the *a priori* probability of the sequence of characters, which describes what is allowable in that language and with what probability; and $P(X)$ is the *a priori* probability of the data. Since $P(X)$ is the same for all C, maximizing $P(C|X)$ can be accomplished by maximizing the product $P(X|C)\ P(C)$.

The feature model $P(X|C)$ is approximated by taking the product of the component probabilities for the different characters, $P(X_i|c_i)$, where X_i is the sequence of feature vectors corresponding to character c_i. The feature model for each character is given by a specific HMM. The language model $P(C)$ is described by a lexicon of allowable characters and words and by a statistical language model that can provide the probability of different sequences of characters and words. The most popular language model used for recognition is an n-gram Markov model, which computes $P(C)$ by multiplying the probabilities of consecutive groups of n words (or characters) in the sequence C. Typically, bigram (n=2) and trigram (n=3) statistical models are used.

2.2 Hidden Markov Models

A hidden Markov model (HMM) has the same structure as a Markov chain, with states and transition probabilities among the states, but with one important difference: associated with each state in a Markov chain is a single "output" symbol, while in a HMM, associated with each state is a probability distribution on all symbols. Thus, given a sequence of symbols produced by a model, we cannot unambiguously determine which state sequence produced that sequence of symbols; we say that the sequence of states is *hidden*. However, we can compute the sequence of states with the highest probability of having produced the observed symbol sequence. If we associate symbols with feature vectors, then the recognition problem can be formulated as finding the sequence of states (or characters or words) that could have produced the sequence of feature vectors with the highest probability. Because of the Markov property of HMMs, the search for the most likely word sequence can be computed very efficiently using the Viterbi algorithm [13]. HMMs have several important properties that make them desirable:

a. HMMs provide a rigorous and flexible mathematical model of variability in feature space as a function of an independent variable (time for speech and position for text).

b. Segmentation and recognition are performed jointly using very efficient training and search techniques.

c. Training is performed automatically without any manual segmentation of data.

d. Higher level constraints (in the form of language models, for example) can be applied as part of the recognition process, if desired, instead of applying them as a postprocess.

e. The techniques are language-independent in principle, requiring only sufficient training data and a lexicon from the new language in order to recognize that language.

Perhaps the most important of these properties is that the HMM parameters can be estimated automatically from training data, *without the need to either pre-segment the data or align it with the text.* The training algorithm requires

- a set of scanned data to be used for training,
- the transcription of the data into a sequence of words of text, and
- a lexicon of the allowable set of characters and words.

The HMM training algorithm automatically estimates the parameters of the models and performs the segmentation and recognition simultaneously, using an iterative scheme that is guaranteed to converge to a local optimum.

2.3 Overall System

Fig. 1 shows a block diagram of the OCR system, which is identical to our speech recognition system, but with the following exceptions: the input here is scanned images instead of speech, characters replace phonemes, and orthographic rules (see below) replace phonological rules. The system depends on the estimation of character models, as well as a lexicon and grammar, from training data.

The training system takes scanned-text data, coupled with ground truth, as input. Ground truth here is given as the sequence of words that correspond to the different lines in the input. Note that the location of the lines on the page is not provided, nor is any segmentation or alignment between the scanned data and ground truth at the word or character level. After a preprocessing stage in which the page is deskewed and lines of text are located, a set of features is extracted. The character modeling component then takes the feature vectors and the corresponding ground truth and estimates the various character models. The character modeling component also makes use of a lexicon and a grammar, which are obtained from a large text corpus using a language modeling component.

The training process also makes use of orthographic rules that depend on the type of script. For example, the rules state that the text consists of text lines and tell whether the lines are horizontal or vertical, and whether the text is read from left-to-right (as in Roman script) or right-to-left (as in Arabic script). The rules may also specify which sets of characters can appear together as ligatures. (A ligature is a combination of two or more letters that looks different from the simple concatenation of the letters.) The rules could also specify other aspects of the writing structure, such as the diacritics in Arabic script, radical decomposition in Chinese script, and syllable structure in Korean Hangul script. Orthographic rules are not always necessary, but they could serve to minimize the recognition search and to reduce the amount of training needed by providing prior information about the form that the models should take.

The recognition system in Fig. 1 has a preprocessing and feature extraction component identical to that used in training. Then, using the output of the feature extraction, the recognition uses the different knowledge sources estimated in the training (character models, lexicon, grammar, and orthographic rules) to find the character sequence that has the highest likelihood.

In our system, all knowledge sources, shown as ellipses in Fig. 1, depend on the particular language or script. However, the whole training and recognition system, shown as rectangular boxes in Fig. 1, is designed to be language-independent. That means that the same basic system can be used to recognize most of the world's languages with little or no modification. This language independence has been demonstrated for speech recognition using HMMs and, in this paper, we demonstrate it for OCR as well.

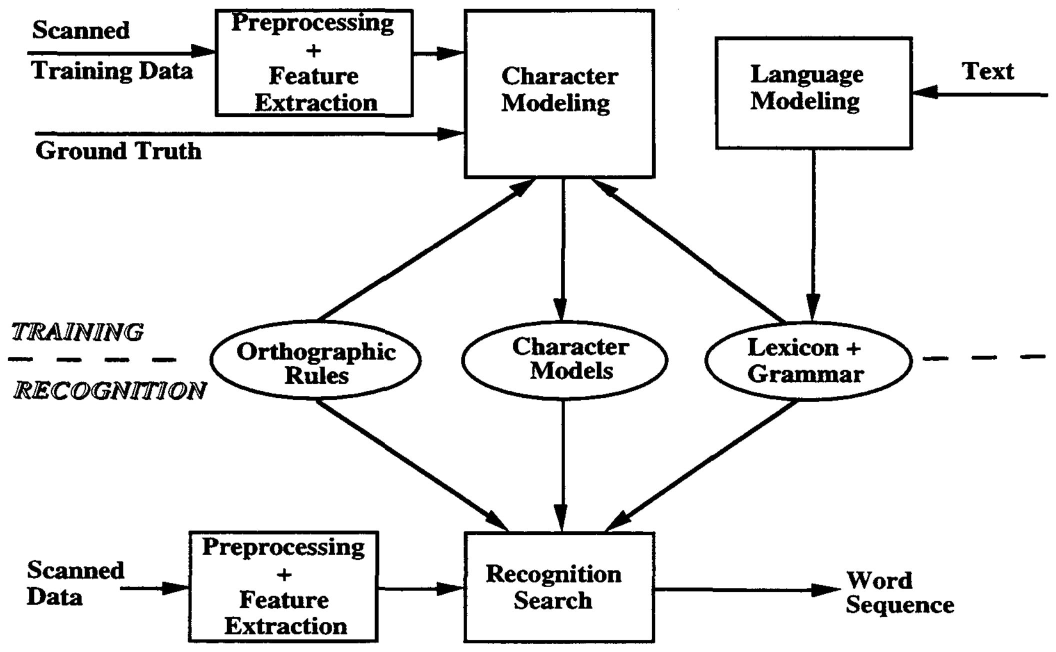


Fig. 1: A block diagram of the OCR system. The training and recognition processes, shown by the rectangular boxes, are language independent.

3 OCR System Detail

3.1 Preprocessing and Feature Extraction

The question of how to apply HMMs to the OCR problem is an interesting one. In order to use HMMs, we need to compute a feature vector as a function of an independent variable. In speech, we divide the speech signal into a sequence of windows (which we call *frames*) and compute a feature vector for each frame; the independent variable then is clearly time. The same method has been applied successfully to on-line handwriting recognition, where a feature vector is computed as a function of time also [14]. However, in OCR, we are usually faced with the problem of recognizing a whole page of text, so there is no obviously natural way of defining a feature vector as a function of some independent variable and, in fact, different approaches have been taken in the literature [5-11]. At this stage in our work, we have chosen a line of text as our major unit for training and recognition. Therefore, we segment a page into a set of lines (which we assume to be horizontal, without loss of generality) and then use horizontal position along the line as the independent variable. Therefore, we scan a line of text from left to right (right to left for Arabic script), and at each horizontal position, we compute a feature vector that represents a narrow vertical strip of the input, which we call a *frame*. The result is a feature vector as a function of horizontal position.

Prior to finding the lines, we find the skew angle of the page and rotate the image so that the lines are horizontal. We then use a horizontal projection of the page to help find the lines. For each line of text, we find the top and bottom of the line. Once each line is located, we are ready to perform feature extraction.

We divide a line into a sequence of overlapping frames. Each frame is a narrow vertical strip with a width that is a small fraction (typically about 1/15) of the height of the line, and the height is normalized to minimize the dependence on font size. The overlap from one frame to the next is a system parameter; currently, the overlap is equal to two-thirds of the frame width (see Fig. 2). Fig. 2 also shows that each frame is divided into 20 equal overlapping cells (again, the cell overlap is a system parameter). The features we compute are simple and language-independent:

- intensity (percentage of black pixels within each cell) as a function of vertical position;
- vertical derivative of intensity (across vertical cells);
- horizontal derivative of intensity (across overlapping frames);
- local slope and correlation across a window of 2 cells square.

Note that we have specifically chosen not to include features that require any form of partial recognition, such as sub-character pieces (e.g., lines, curves, dots), nor did we want to include features that are specific to a particular type of script.

Although the intensity features alone represent the entire image, we include other features, such as vertical

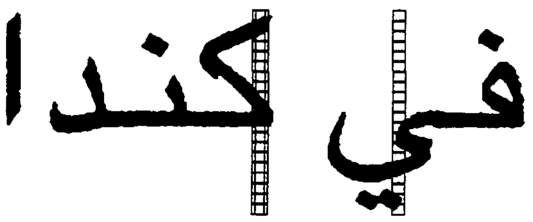


Fig. 2: Dividing a line of text into frames and each frame into cells.

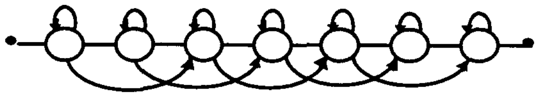


Fig. 3: Hidden Markov model (HMM) for characters. Shown is a 7-state model. For OCR, we have used 14-state models.

and horizontal derivatives, and local slope and correlation, so as to include more global information. We have found in speech recognition that the inclusion of such features helps overcome the limitation imposed by the conditional independence assumption that is inherent in HMM models. The result is a set of 80 simple features. Now that we have a feature vector computed as a function of position, we are ready to put HMMs to good use.

3.2 HMM Character Structure

The central model of the OCR system is the HMM of each character. For each model, we need to specify the number of states and the allowable transitions among the states. Associated with each state is a probability distribution over the features. The model for a word then is a concatenation of the different character models.

Our proposed HMM character structure is a *left-to-right* structure that is similar to the one used in speech (see Fig. 3). The loops and skips in Fig. 3 allow relatively large, nonlinear variations in horizontal position. During training and recognition, for any particular instance of a character, several of the input frames may get mapped to each of the states and other states may not be used at all. In our current system, we are using 14 states for all character HMMs. This structure was chosen subjectively to represent the characters with the greatest number of horizontal transitions. While it would be possible to model different characters with different numbers of states, we find it easier to use the same number of states for all characters.

ت ج ر ب	ك ه ب ة	ل ن م ل أ	ي م ل	a
تجرب	كهبة	لنملأ	يمل	b

Fig. 4: Arabic words, written (a) with the characters isolated and (b) as they normally appear in print.

Context-Dependent Character Models

One of the major advances in speech recognition has been the use of context-dependent phonetic models to account for coarticulation between adjacent phonemes. That is, the parameters (probability distributions and transition probabilities) for each state in each phoneme depend to some extent on the identity of the preceding and following phonemes. In speech recognition, this reduces the error rates by a factor of two [15]. We have found that the same techniques apply equally well to on-line cursive handwriting recognition [14], because the strokes of one letter are affected by the adjacent letters. In OCR, context-dependent models are especially appropriate for languages with cursive script (such as Arabic); for dealing with overlapping printed letters; and for modeling ligatures. Fig. 4 shows examples of a few Arabic words, each written with the characters isolated and as the word appears in normal print. Note that not all characters are connected inside a word and that the shape of a character depends on the neighboring characters. Even though a character can have different shapes, it is important to note that, in our system, we need not provide this information explicitly in the system. The context dependence of the characters is modeled implicitly by including a separate model of each character in the context of all possible left and right characters. Furthermore, the transcription of text in the training data does not include any information about the shape of each character; only the identity of the basic character is given. It is one of the advantages of our system that such detailed information is not necessary for good performance.

Probability Structure

Each state in our HMMs has an associated probability density on the feature vector. We employ what is known in the speech recognition literature as a tied-mixture Gaussian structure [16]. For computational reasons, we divide our 80-dimensional feature vector into 8 separate subvectors of 10 features each, which we model as conditionally independent – thus our probability densities can be expressed as a product of 8 probabilities, one for each subvector. The probability density corresponding to each subvector is modeled using a mixture of a shared pool of 64 Gaussian densities. The parameters for these densities are estimated using a clustering process [17] that is run on a subset of the training data. Thus, each state has, for each of the 8 subvectors, 64 "mixture weights" (one for each Gaussian) that represent the probability density for that subvector.

3.3 Training and Recognition

The training algorithm remains unchanged from that used in our Byblos speech recognition system. We do not require the scanned data to be hand segmented nor aligned, neither at the character level nor at the word level, but only at the line level. We only use the ground truth transcriptions, which specify the sequence of characters to be found on each line. The probability densities corresponding to the states in the various HMMs are all initialized to be uniform densities. We then use the forward-backward training algorithm to derive estimates of the model parameters [1]. The resulting models maximize the likelihood of the training data. However, often there is insufficient training data to estimate robust probability distributions for all of the context-dependent models. Therefore, we cluster the distributions of similar states with insufficient training together and then we retrain the weights of these clustered states.

In addition to the character HMMs, we also compute a lexicon and a language model. These are usually obtained using a large text corpus. The lexicon contains the letters used to spell each word. The language model is usually a bigram or trigram model that contains the probabilities of all pairs or triples of words. Note that only the text is needed for language modeling; it is not necessary to have the corresponding scanned image. In this way, much larger amounts of text can be used to develop more powerful language models, without having to get more scanned training data.

The recognition algorithm is also identical to that used in our speech recognition system. Given the output of the analysis of a line of text, the recognition process consists mainly in a search for the most likely sequence of characters given the sequence of input features, the lexicon, and the language model. Since the Viterbi algorithm would be quite expensive when the state space includes a very large vocabulary and a bigram or trigram language model, we use a multi-pass search algorithm [18].

4 Experiments with Arabic Corpus

4.1 The Corpus

To demonstrate the effectiveness of our approach, we chose to work first on the DARPA Arabic OCR Corpus, collected at SAIC. The corpus consists of scanned Arabic text from a variety of sources of varying quality, including books, magazines, newspapers, and four Apple computer fonts (Geeza, Baghdad, Kufi, and Nadim). Samples of some of the pages in the corpus are shown in Fig. 5. Arabic script provides a rich source of challenges for OCR [12]:

- Connected Letters: Of 31 possible basic isolated letters, 24 can connect to others from the right and the left; six connect only to the right; and one letter does not connect to other letters at all. Since not all letters connect, word boundary location becomes an interesting challenge.
- Context Dependence: The shapes of letters change – often radically – depending on the connectedness of neighboring letters; also, certain character combinations form new ligature shapes which are often font-dependent.
- Dots: A significant number of letters are differentiated only by the number and position of dots in the letter.
- Diacritics: These are optional marks that are placed above or below characters and they mostly represent short vowels and consonant doubling.

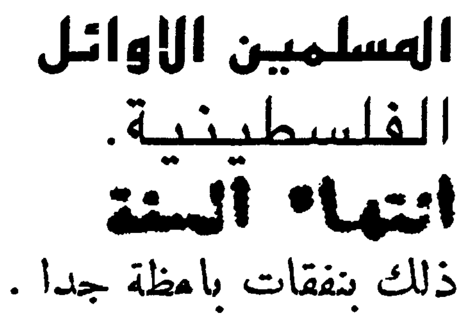

المسلمين الاوائل
الفلسطينية.
انتهاء السنة
ذلك نفقات باهظة جدا .

Fig. 5: Samples from the DARPA Arabic OCR Corpus.

In this effort, we show that our approach deals very effectively and automatically with the connectedness of letters and the context-dependence of their shapes. The system is not given the rules for how shapes change as a function of the adjacent letters nor for how two or three letters might combine to form ligatures. Rather it learns these shapes implicitly by modeling each character in a context-dependent way. (Dealing with diacritics explicitly was beyond the scope of this study.)

The DARPA Arabic Corpus consists of 345 pages of text (~670k characters), scanned and stored at 600 dots per inch. Associated with each image is the text transcription, indicating the sequence of characters on each line. But the location of the lines and the location of the characters within each line are not provided. The corpus transcription contains 80 unique characters, including punctuation and special symbols. Note, however, that the shapes of these characters can vary a great deal, depending on their context. The various shapes, including ligatures and context-dependent forms, were *not* identified in the ground truth transcriptions.

4.2 Experimental Conditions

For our experiments thus far, we removed all pages with untranscribed characters (mostly non-Arabic characters), horizontal straight lines, footnotes, and those for which the simple line finding algorithm we had implemented did not provide the correct number of lines. This left us with 313 pages (~600k characters) of which 68 pages were from the four computer fonts and the remaining pages were from books, magazines, and newspapers, with a large number of unidentified fonts.

We ran experiments under two conditions: (a) omnifont, and (b) unifont. In the omnifont experiment, two-thirds of the 313 pages were chosen randomly as the training set; the remaining one-third of the pages were used as the test set. In the unifont experiment, the system was trained on each of the four computer fonts separately and tested on (different data) from the same font. No other training data was used to train the system. All experiments used a 30,000-word lexicon generated from all 345 pages of the corpus.

Because the amount of training text data in this corpus was not sufficient to estimate word bigram probabilities for most of the two-word sequences, we decided to estimate what amounted to a unigram language model for the words, and a bigram model between the words and neighboring space and punctuation characters.

4.3 Evaluation Procedure

We measured the character recognition error rate following speech recognition conventions (i.e., we added the number of substitutions, deletions, and insertions, and divided by the total number of characters in the transcriptions provided). Each recognized line was aligned with ground truth (using dynamic programming) so as to minimize the error rate just defined. In our initial experiments, we found that about one-third of the errors were due to inserted or missing space characters. After examination of the data, it became clear that the placement of spaces in the printed Arabic texts was often arbitrary (some inter-word spaces were often smaller than intra-word spaces) and the sizes of what was called a space in ground truth varied a great deal. Therefore, we decided to recompute our error rates such that a space error was counted only when it really mattered, namely, when the space error caused a word to be wrong.

4.4 Experimental Results

Table 1 shows the results of the omnifont recognition experiment, broken down according to the source of the data (the error rates for the individual computer fonts are broken down in Table 2 under the omnifont column). The average character error rate for all sources is 2.8%. The error rate was highest for books, where the data had very high variability (almost every page came from a different font) and was generally of lower print quality than the other sources.

Table 1. Omnifont percent character error rates for the DARPA OCR Arabic Corpus.

	omnifont
Books	6.1
Newspapers	2.1
Magazines	2.5
Computer Fonts	2.2
Average	2.8

Table 2. Percent character error rates for four computer fonts under unifont and omnifont training conditions.

	unifont	omnifont
Geeza	0.6	1.2
Baghdad	0.6	1.7
Kufi	0.3	5.4
Nadim	0.4	1.2
Average	0.4	2.2

Table 2 shows unifont and omnifont test results for the four computer fonts. The first column, with the unifont results, shows an average character error rate of 0.4%. The second column shows the omnifont error rates for the same computer font data (these are the results whose average is shown in Table 1). As expected, the error rate increases from the unifont to the omnifont condition for each of the four fonts. However, the increase in error rate is most pronounced for the Kufi font. Even though there are only about 10 pages of data for this font, it was easier to recognize that font in the unifont mode (when compared to the other three fonts) due to its greater degree of regularity (see the top line of Fig. 5). However, the error rate jumped significantly in the omnifont experiment, most likely because this stylized font is so radically different from all the other fonts.

In the tables above, more than 70% of the character errors were errors in the alphabetic characters; the others were due to punctuation, numerals, special symbols, and diacritics. Of the alphabetic errors, there are certain classes of errors that were expected and were easy to discern. For example, Fig. 6 shows three classes of characters that resulted in a significant number of confusions. In Figs. 6a and 6b, the characters are differentiated from each other chiefly by the number and location of dots in the character, while in Fig. 6c the confusion is the presence or absence of a diacritic.

(a) نـ تـ بـ يـ

(b) ي ى

(c) ا أ

Fig. 6: Three classes of characters that resulted in a significant number of confusions.

The conditions of this experiment were unrealistic in one respect: the recognition system employed a closed-vocabulary lexicon (i.e., all the words in the test were in the lexicon). In the future, we plan to remove this constraint, which should increase the error rate somewhat.

5 Experiments with English Corpus

To demonstrate the language independence of our methodology, we used the UW English Document Image Database I [19]. This corpus contains 958 pages scanned from technical articles at 300 dpi. Each page is divided into relatively homogeneous zones, each of which is classified into one of 14 categories (e.g., text, table, text-with-special-symbols, math), and a ground truth transcription is provided. Of a total of 13,238 zones, 10,654 are text zones. Text zones were also classified into one of three styles: plain, italic, or bold. The label was assigned depending on the dominant style in the zone; for example, a plain zone could contain some italic words in it. For these experiments, we used only data from the text zones. The size of the character set is 90.

In order to deal with this data, we made only two changes to our Arabic system: we took account of the fact that English is written left-to-right instead of right-to-left and we parameterized the system to take other sampling rates (in this case 300 dpi instead of 600 dpi). Just like in the Arabic system, we used 14-state HMMs to model characters. However, instead of using context-dependent models, we used context-independent models since, in general, the shapes of characters in English do not depend on their neighbors.

In our first experiments, we trained our system on about 50k characters taken from plain text and tested on 16k characters from plain text taken from 400 different zones and from pages different from training. The results are shown in the first column of Table 3. With a closed-vocabulary lexicon of about 30k words taken from all the words in the corpus, the average character error rate was 1.2%. Clearly, this is an optimistic result since all the words in the test were known. Therefore, in a second experiment, we did not use a lexicon at all, but used a trigram language model on the sequence of characters. The result in this case was a character error rate of 3.2%. But this is an unduly pessimistic result since it should be possible to use a general lexicon to improve performance. Therefore, we implemented a system in which a general lexicon of 30k words was used to help correct errors in the recognition (the lexicon was obtained from all the training data but did not include the test data). The resulting character error rate for the hybrid system was 1.4%. We found it interesting that a relatively modest 30k-word lexicon could have this dramatic effect on recognition accuracy.

A cursory examination of the errors showed that about two-thirds of the errors were words in italics (recall that text labeled as plain can contain italic words). Note that we had only a single context-independent model for every character, and that single model had the task of modeling all fonts in the training data. Since there were very few italic words in the training, it is not surprising that a single model for all fonts had problems with italics.

Table 3. Character error rates for the University of Washington English Document Database I.

	TRAINING DATA	
	Plain	Plain/Italic
30k Lexicon	1.2	0.8
No Lexicon	3.2	2.1
Hybrid System	1.4	1.1

To alleviate the problem of little italic training data, we performed a second set of experiments in which we included another 50k italic characters in the training data. With the 30k closed lexicon, the models obtained by training on the mixed plain/italic data reduced the error rate from 1.2% to 0.8% – a significant reduction in character error rate. With no lexicon, there was a similar reduction in error rate from 3.2% to 2.1%. By using the hybrid system of character recognition combined with a general lexicon, the character error rate diminished from 1.4% to 1.1%. This experiment points to the importance of having enough training data from the different styles that are expected to be encountered in the test.

It is important to emphasize that, even when we included italic text in our training data, we still maintained only a single context-independent model for each character. The fact that such a simple model could do well for italic as well as plain data was quite surprising.

6 Conclusions

In this paper, we demonstrated how speech recognition technology, based on hidden Markov models, can be used effectively for OCR. As an initial study, using a system that was designed for speech recognition, we are very encouraged by the OCR results we obtained for Arabic – a particularly difficult cursive script. It is very gratifying to see that the context-dependent HMMs were able to model the different shapes of the Arabic characters without telling the system the identity of those shapes in the ground truth data.

Since the system and features we chose are not specific to any language, we see no reason why they should not work for other languages as well. To demonstrate the language independence of our system, we performed preliminary experiments on English OCR and obtained very encouraging results. We consider the portability of our system to various scripts and languages within a relatively short period of time to be one of its major advantages.

Model-based approaches, such as HMM, tend to be computationally more compute intensive than rule-based systems but, in general, can lead to superior recognition performance. In addition, there are search strategies that can speed up the recognition process significantly [18].

Much work remains to optimize system performance and to render it more robust to various types of degradation. However, having demonstrated the soundness of the general methodology, we expect that future advances in OCR will mirror the large advances that occurred in speech recognition but, we hope, over a shorter period of time.

Acknowledgments

The authors wish to thank Dr. Ying Zhao and Dr. Christopher Raphael for their many contributions to this project.

References

[1] L. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," *Proc. IEEE* **77**, 257-286, (1989).

[2] J. Makhoul and R. Schwartz, "State of the Art in Continuous Speech Recognition," *Proc. Natl. Acad. Sci. USA* **92**, 9956-9963, (1995).

[3] R. Schwartz, C. LaPre, J. Makhoul, C. Raphael, and Y. Zhao, "Language-Independent OCR Using a Continuous Speech Recognition System," *Int. Conf. Pattern Recognition*, Vienna, Austria, C-99-103, (1996).

[4] Y.L. Chow, M.O. Dunham, O.A. Kimball, M.A. Krasner, G.F. Kubala, J. Makhoul, P.J. Price, and S Roucos, "BYBLOS: The BBN Continuous Speech Recognition System," *IEEE Int. Conf. Acoustics, Speech, Signal Processing*, Dallas, TX, 89-93, (1987).

[5] A. Kundu and P. Bahl, "Recognition of handwritten script: a hidden Markov model based approach," *IEEE Int. Conf. Acoustics, Speech, Signal Processing*, New York, NY, 928-931, (1988).

[6] E. Levin and R. Pieraccini, "Dynamic planar warping for optical character recognition," *IEEE Int. Conf. Acoustics, Speech, Signal Processing*, San Francisco, CA, III-149-152, (1992).

[7] J.C. Anigbogu and A. Belaid, "Performance evaluation of an HMM based OCR system," *Proc. 11th Int. Conf. Pattern Recognition*, The Hague, The Netherlands, 565-568, (1992).

[8] J.A. Vlontzos and S.Y. Kung, "Hidden Markov models for character recognition," *IEEE Trans. Image Processing*, 539-543, (1992).

[9] G. Kopec and P. Chou, "Document image decoding using Markov source models," *IEEE Int. Conf. Acoustics, Speech, Signal Processing*, Minneapolis, MN, V-85-88, (1993).

[10] O.E. Agazzi and S. Kuo, "Hidden Markov model based optical character recognition in the presence of deterministic transformations," *Pattern Recognition* **26**, 1813-1826, (1993).

[11] H.-S. Park and S.-W. Lee, "Off-line recognition of large-set handwritten characters with multiple hidden Markov models," *Pattern Recognition* **29**, pp. 231-244, (1996).

[12] B. Al-Badr and S. Mahmoud, "Survey and bibliography of Arabic optical text recognition," *Signal Processing* **41**, 49-77, (1995).

[13] G.D. Forney, "The Viterbi Algorithm," *Proc. IEEE* **61**, 268-278, (1973).

[14] T. Starner, J. Makhoul, R. Schwartz, and G. Chou, "On-line Cursive Handwriting Recognition Using Speech Recognition Methods," *IEEE Int. Conf. Acoustics, Speech, Signal Processing*, Adelaide, Australia, V-125-128, (1994).

[15] R.M. Schwartz, Y. Chow, S. Roucos, M. Krasner, and J. Makhoul, "Improved Hidden Markov Modeling of Phonemes for Continuous Speech Recognition," *IEEE Int. Conf. Acoustics, Speech, Signal Processing*, San Diego, CA, 35.6.1-4, (1984).

[16] J. Bellegarda and D. Nahamoo, "Tied Mixture Continuous Parameter Models for Large Vocabulary Isolated Speech Recognition," *IEEE Int. Conf. Acoustics, Speech, Signal Processing*, Glasgow, Scotland, **1**,13-16, May 1989.

[17] J. Makhoul, S. Roucos, and H. Gish. "Vector Quantization in Speech Coding," *Proc. IEEE* **73**, 1551-1588, (1985).

[18] R. Schwartz, L. Nguyen, and J. Makhoul, "Multiple-Pass Search Strategies," in *Automatic Speech and Speaker Recognition: Advanced Topics*, C-H. Lee, F.K. Soong, K.K. Paliwal, Eds., Kluwer Academic Publishers, 429-456, (1996).

[19] I.T. Phillips, S. Chen, and R.M. Haralick, "CD-ROM document database standard," *Proc. Int. Conf. Document Analysis and Recognition*, Tsukuba City, Japan, 478-483, (1993).

Application of Blackboard Technology to Allow Non-Linear OCR Processing

Christopher Hood
Lockheed Martin Federal Systems
Gaithersburg, MD
`chris.hood@lmco.com`

Abstract

This paper presents a Non-Linear approach to performing OCR. A Blackboard Architecture is used to construct a system which allows the processing to backtrack paths of deduction that show little promise, and which can utilize disparate, heterogeneous Experts in unison. Judgement of the promise of a path of deduction is made possible through Belief Values, which are computed for each generated hypothesis and are continuously updated during processing. Tests have shown this implementation to achieve improved recognition accuracy over a comparible linear OCR implementation.

1 Introduction

One of the problems faced during the linear processing[1] of any problem is that of a mistake made early in the process, that severely affects the final outcome. Another is processing, performed toward the end of the process, cannot influence processing which took place earlier. Linear OCR (Optical Character Recognition) systems suffer, to some degree, from both of these problems. For example, if a line of text is not properly segmented, there is little or no chance that the individual symbols will be properly identified, and further, the process attempting to identify the symbols has no way to communicate it's difficulties so that the segmentation may be reattempted.

In an attempt to overcome these limitations, we have devised a Blackboard based OCR system, which uses Belief Values to rank both the hypotheses produced at the different OCR stages, as well as the Experts used to create those hypotheses. Use of a Blackboard Architecture allowed us to create a flexible, data driven design in which OCR data and the processes (segmentation, classification, etc.) that operate on the data, are treated as individual and distinct entities. This allows the processes, called Experts, to be applied in any order, and to any current data on which they can operate. It also allows for the Experts to be heterogeneous in nature, i.e., there can be multiple page segmenters, line segmenters, classifiers, etc., each distinctly different in their inner workings, but interchangeable with each other, given that they produce the same type of output. In addition, the Blackboard Architecture made it possible to make the output from the different Experts available, at all times, to all other Experts and it allows new experts to be added to the system with relative ease.

Ranking the hypotheses from an Expert (i.e. the Expert's output), gave us the ability to judge the correctness of each hypothesis in relation to an overall level of acceptance and to other hypotheses of the same type (even if they were created by different Experts). This allowed us to determine which hypotheses were deserving of further processing and which ones should be abandoned (though perhaps only temporarily) in an attempt to create a better hypothesis. The process of abandoning a hypothesis with a low Belief Value in an attempt to create one with a higher Belief is referred to as 'Backtracking', and is the non-linear element of this system which helps to overcome the problems seen in linear systems cited earlier.

The results from this system thus far have shown it to be able to recognize text better than the linear system from which it was derived. We believe it is the ability to judge the correctness of a hypothesis, together with the ability to backtrack paths of low Belief, which give this increase in recognition.

We begin this paper with a brief overview of what comprises a Blackboard system and of the architecture of our system. Building on this foundation, we next discuss the concept of backtacking and the use of Belief Values within our system. We then conclude with a presentation of our test results and with

[1] Here linear processing is defined as a process which flows rigidly from one step to the next, without any means to deviate from the original ordering, or the ability to adapt to current conditions.

a few words about our future work directions.

2 Background: Blackboards

Perhaps the best way to begin an explanation of a Blackboard, is with a real world analogy.

2.1 A Room Full of Scientists

Imagine, if you will, a classroom full of Scientists. Each one is sitting at desk and each is an expert in a particular field of study. At the front of the room is a large, blank Blackboard. We want to get this group of people to work together to solve a general problem. So one person, which we will call the Controller, goes to the front of the room and writes the problem to be solved on the Blackboard, along with any initially known data. At this point, with any luck, one or more of the Scientist will realize that using their particular expertise, they can perform an operation on the initial data which will help to solve the problem. These enlightened Scientist then raise their hands to indicate they have something to contribute. The Controller will choose the Scientist (or Scientists) they believe can contribute the best answers and call upon them one at a time (or in parallel, if the Controller has extra pieces of chalk). When a Scientist is called upon, they go to the Blackboard and write down the new data they are able to deduce and then sit back down. This new information, together with information from others and the original data, will hopefully cause other Scientists to get new ideas and to raise their hands. This cycle of Scientists processing current data and the Controller calling on them to add new data to the Blackboard continues until an answer is produced.

2.2 General Blackboard Architecture

So, in essence a Blackboard is a *data driven* hierarchical data base. But, as we can see from the analogy, it is different from a normal data base in that a Blackboard is a data base with an active purpose, that being a problem to solve. A general Blackboard system can be separated into three components: Data, the Experts, and the Controller.

The *Data* is of course the information used to solve a problem, and is composed of both the original data and all data deduced during the problem solving process. This data is generally organized into *blackboards*. These are the units within a Blackboard system, which encapsulate and organize data that share some kind of relationship making it easier to access.

The *Experts* are a group of specialized functions or programs which perform operations on the data (in the classroom example these would be the Scientists). These Expert can be any kind of processing. They can be complete COTS systems, or pieces of a COTS system which have been pulled out and encapsulated so they can be called as stand alone processes. Experts can also be functions specially written for the Blackboard itself. The principles that the different Experts operate on can be similar or they can be completely different. Even the languages used to implement the Experts can vary. In other words, the Experts can be completely heterogeneous, which is one of the main strengths of the Blackboard Architecture, its ability to bring together disparate types of processing in order to solve a common problem.

The *Controller* is made up of code which takes care of choosing which Experts are to be called upon and of scheduling and executing those Experts. In our case it makes these decisions based on the input requirements of the Experts, what data is available and the reliability of the different Experts. The problem solving process is therefore driven by the Controller based purely upon the data available and the resources of the Experts.

3 System Architecture Overview

The OCR system presented in this paper has the same general architecture described in Section 2. This section gives a brief overview of each of the three main components of our Blackboard system.

3.1 Data

The Data component of our Blackboard is separated into six classes:

- Subject Data (images, paragraphs, characters, etc.)
- Belief Data
- Expert Call Records
- Expert Characteristics Data
- Correlation Data
- Goal Data

The data is organized by encapsulating each of the six classes in its own *blackboard* or set of *blackboards*. The data is then linked together across these *blackboards* to make it convenient to access and to reflect certain desired relationships, such as parent/child relationships within the Subject Data.

3.1.1 Subject Data

The Subject Data is composed of six different types of objects: images, pages, paragraphs, lines, characters and codepoints. Each of these types is encapsulated in its own *blackboard*. This allows operations to be performed over entire object types,

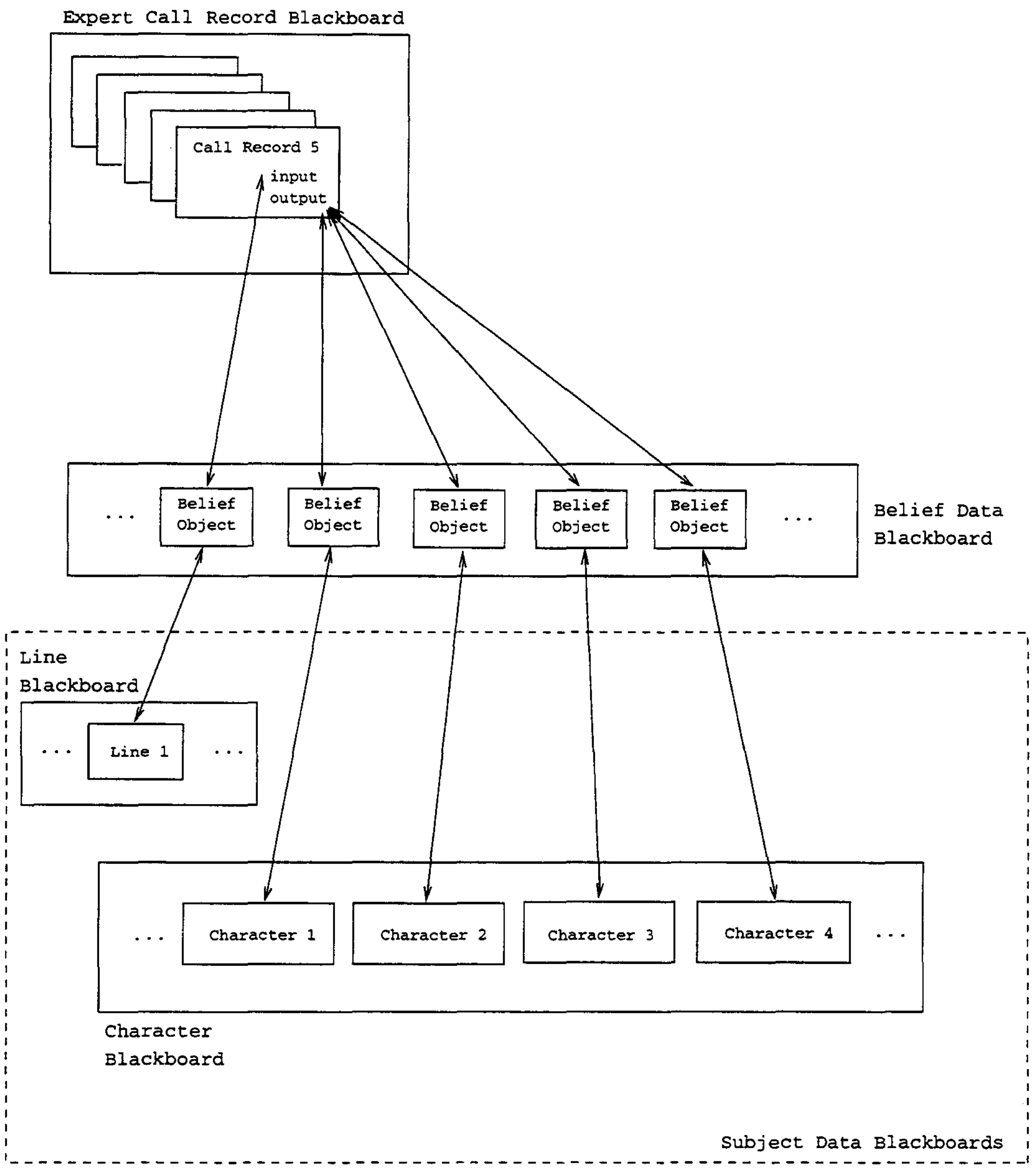


Figure 1: Structure of links between Data Objects.

e.g. when a new character is created a search can be performed over the entire character *blackboard* to see if the new character matches an existing character instantiated by another expert. Information such as this will play a vital role in determining the overall Belief Value of any given Subject Data Object. Additionally, each *blackboard* is indexed to allow efficient searches to be performed. For objects other than codepoints, this indexing is accomplished by using the coordinates of the polygon which bounds each object. This gives the indexing a spatial quality that can be used to determine whether the bounding polygons of different objects overlap, which again, will play a role in determining final Belief Values. Codepoint objects are indexed simply by the value of the codepoint which they store.

3.1.2 Belief Data

Each object which exists in one of the six Subject Data *blackboards* will have a single and unique Belief Data Object created and maintained for it. These objects encapsulate all pertinent information about the Belief Value of the Subject Data Object for which they were created. A link is established between these Belief Objects and their Subject Data Objects, see Figure 1. (Note that this link spans from the Belief Data *blackboard* to a Subject Data *blackboard*. It will be true that any links discussed in this paper will always span from one *blackboard* to another.) This link provides the ability to find any Subject Data Object given its Belief Object and *vice versa*.

The Belief Object is updated continuously during processing to reflect any changes in the Belief Value of the Subject Data Object to which it is linked. The Belief Value is initially determined in various different ways depending on the type of Subject Data Object.

3.1.3 Expert Call Records

Records of which Experts were called, along with which Subject Data Objects were used as inputs and which were produced as outputs, are maintained in Expert Call Records. Each Call Record will contain links to the Belief Objects associated with the different input and output elements of an Expert. It was decided to store the Belief Objects instead of the Subject Data itself because all decisions made by the Controller Code to determine which Experts to call, what input data to pass them and which outputs are acceptable, are based on Belief Values. And furthermore, since a Subject Data Object and its Belief Object are linked, any needed information stored in the Subject Data Object can be accessed through its Belief Object.

The Expert Call Records are also used to imply a parent/child relationship between Subject Data Objects. For example, referring again to Figure 1, say the parent of Character 4 is needed. It can be determined by following the link from Character 4 to its Belief Object and from there to the Expert Call Record which lists Character 4 as an output value, in this case Call Record 5. Once at Call Record 5, the links to the Belief Objects of the input data will provide the objects from which Character 4 was created. In this case Line 1 is the parent of Character 4. If further ancestry is needed, all that is required is to repeat this process starting with Line 1. In this way a Subject Data Object's ancestry can be traced as far back as needed. This ability is essential when backtracking a Subject Data Object with a low Belief Value.

Another property of the relationship between Call Records and Belief Objects worth noting is that a Belief Object will appear in one and only one Call Record as output data. However, a Belief Object can appear in any number of Call Records as input data.

3.1.4 Expert Characteristics Data

Expert Characteristics Data Objects describe the different properties, requirements and resources of individual Experts in the Blackboard system. An Expert Characteristics Object will contain all the information needed to be able to utilize an Expert. Included in this information is the exact input requirements of the Expert and its output types. Also stored in Expert Characteristics is an *a priori* Belief Value, that indicates the amount of faith the Blackboard system has that an Expert will return correct answers. This information is stored so that it is possible to search across all the Experts that take a specific type of input data, or return a specific type of output data to decide the best one to call.

3.1.5 Correlation Data

Correlation Data Objects are used to indicate significant relationships between two or more Subject Data Objects. Like Subject Data Objects, each Correlation Object is linked to its own Belief Object which reflects the Belief Value associated with the correlation. The Correlation Object itself encapsulates two pieces of data: a list of correlated objects and the type of correlation. The list of correlated objects is a series of two or more links to Belief Objects. At the time of this paper, the only type of correlation that is being represented is that of identical Subject Data Objects being created by different Experts, however the Correlation Objects are intended to be capable of representing other types of relationships if needed.

3.1.6 Goal Data

Goal Data Objects represent the intermediate and final goals of the Blackboard system. These simple objects each store the data type of a goal and whether or not it is an intermediate or final goal. This gives us the ability to take different actions, if needed, when particular types of data are produced. For example, our system has a Goal Object that identifies codepoints to be the final goal of the processing. Therefore, when a codepoints object is produced with an acceptable Belief Value, the system knows that it is a final goal and that a search for an Expert that operates on it need not be performed.

3.2 Controller Code

The Controller Code coordinates all of the processing within the Blackboard. It is responsible for determining which Experts are valid processing options based on the current data, choosing which Expert is the "best" at any given step, gathering the input for that Expert, scheduling the Expert to be executed and for creating a Call Record to record the Expert's execution. It is also responsible for determining when backtracking is necessary and how far the backtracking should go. Figure 2 shows the general flow of control in the Controller Code.

3.3 Expert Code

The Expert code is responsible for actually performing the processing on the Subject Data to produce a final answer. Each Expert is defined through an

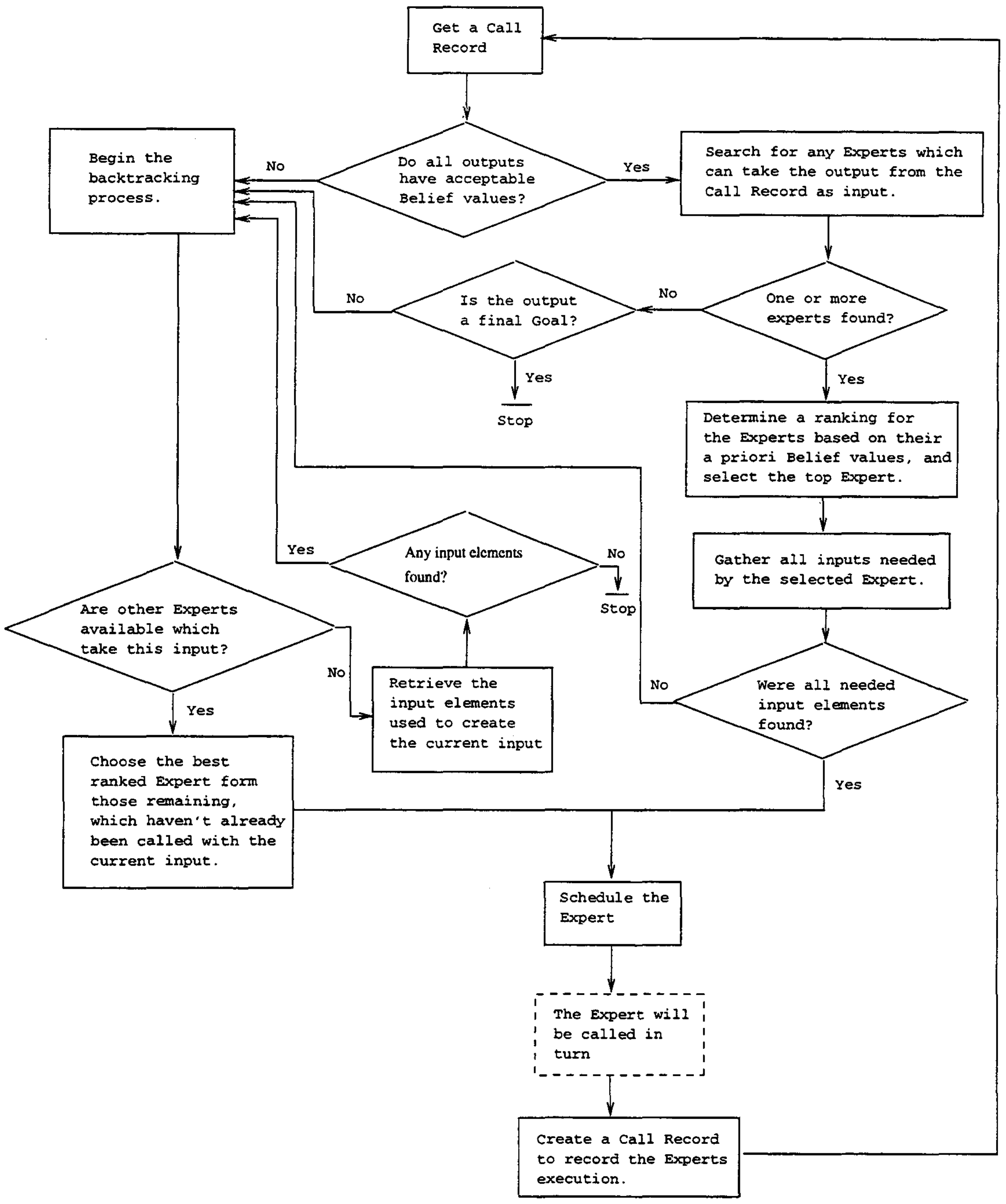

Get a Call Record
Do all outputs have acceptable Belief values?
No
Yes
Begin the backtracking process.
Search for any Experts which can take the output from the Call Record as input.
One or more experts found?
No
Is the output a final Goal?
No
Yes
Stop
Yes
Determine a ranking for the Experts based on their a priori Belief values, and select the top Expert.
Gather all inputs needed by the selected Expert.
Were all needed input elements found?
No
Yes
Any input elements found?
Yes
No
Stop
Are other Experts available which take this input?
No
Yes
Retrieve the input elements used to create the current input
Choose the best ranked Expert form those remaining, which haven't already been called with the current input.
Schedule the Expert
The Expert will be called in turn
Create a Call Record to record the Experts execution.

Figure 2: Flow of Control within Controller Code.

Expert Characteristics Data Object, which specifies what kind of data needs to exist before the Expert can be called and what kind of data the Expert will produce, along with other information about the Expert, such as its *a priori* Belief Value.

There are only four requirements that a process must meet in order to be added to the Blackboard.

- The input data the Expert needs must be available or must be creatable by another Expert in the Blackboard.
- The input Subject Data, as it is stored on the Blackboard, must be convertible into the input data structures needed by the Expert.
- The output from the Expert must convertible back into a form that can be stored in the Blackboard.
- The Expert must return some kind of measurement (or some data that can be measured) which can be used to determine a Belief Value for the Expert's output.

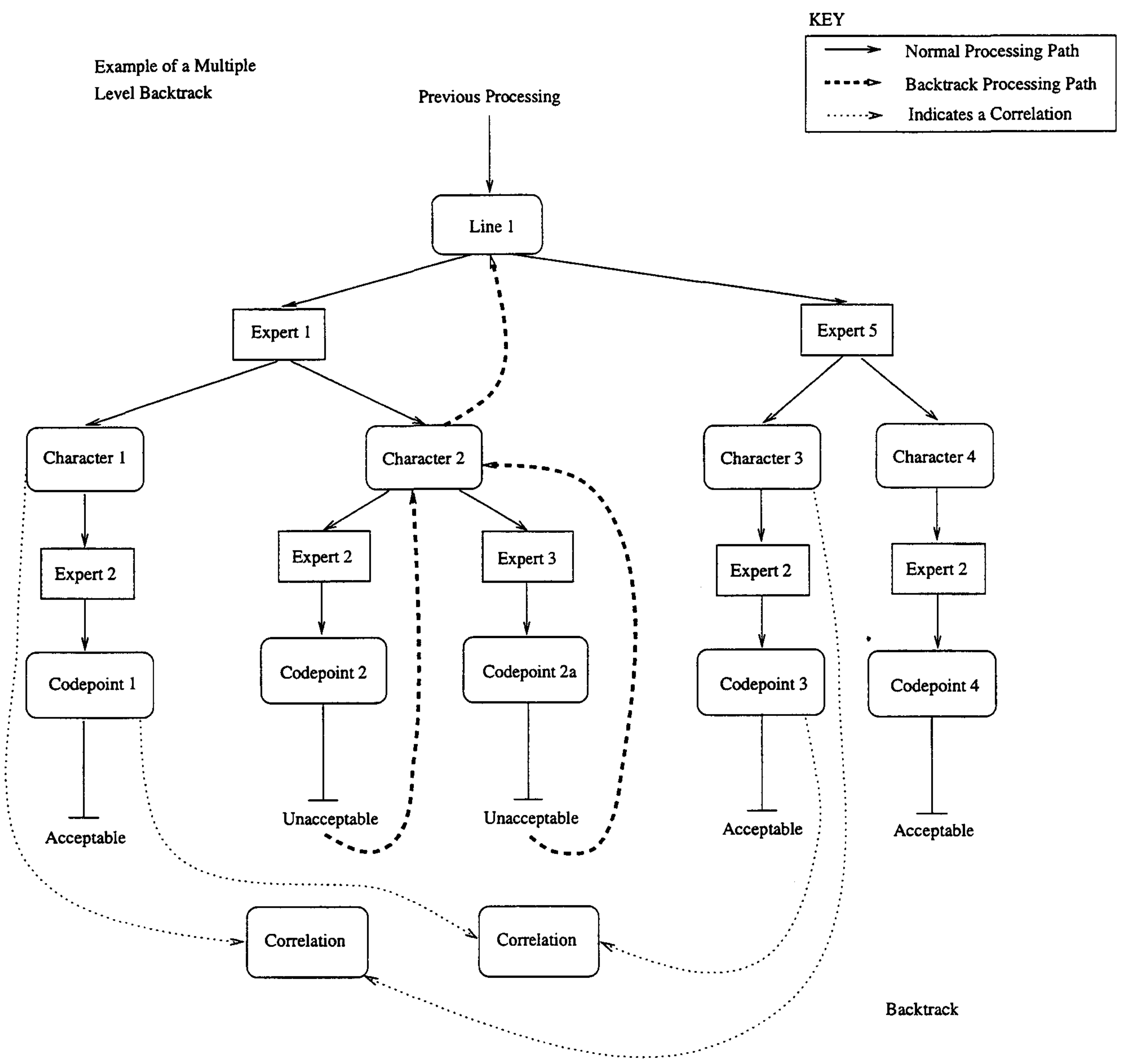


Figure 3: A multilevel backtrack.

4 Backtracking and Belief Values

Of the three main concepts that make our approach successful, we have already mentioned one, the ability to use disparate Experts. Now we will discuss the other two, Backtracking and the use of Belief Values.

4.1 Why Take Two Steps Forward and One Step Back?

Backtracking's importance lies in the fact that it allows one to overcome the problem of making mistakes early on in processing. In a linear system, mistakes made early on cannot be recovered, not so in this Blackboard system. When the Blackboard determines that a path of deduction is producing results that are below a general acceptance level, it can halt that path and attempt to find another, better path of deduction, using a different set of Experts. The determination of whether or not a path of deduction is acceptable is accomplished by analyzing the Belief Values of the various hypotheses produced along that path. The methods used to form these Belief Values and to analyze them will be dealt with later in this section.

First, let's look briefly at what happens when a backtrack occurs. Referring to Figure 3, Line 1 is given to Expert 1 for segmentation. This produces Characters 1 and 2. Each Character is given to Expert 2 for classification. In the case of Character 1 the Codepoint produced was acceptable, so processing on that path stops. Codepoint 2, on the other hand is deemed unacceptable. This will initiate a backtrack to Character 2, which is then given to Expert 3 for reclassification. The new Codepoint, 2a, is again deemed unacceptable so again we backtrack to Character 2. At this point, however, there are no other classifiers to try, so the algorithm steps back one more level, to Line 1 where the processing continues forward again. Line 1 is sent to Expert 5, another line segmenter, which produces Characters 3 and 4. These Characters are passed to Expert 2, which this time returns acceptable Codepoints for both Characters, ending the processing.

If either (or both) of the Characters produced by Expert 5 had not been acceptable and there were no other line segmenters, the backtracking algorithm would have stepped back to the level above Line 1. This process of stepping back one level stops when the original image is encountered. On the other hand, if there had been no other segmenters of any kind left to try, the codepoint with the highest Belief Value is chosen from those produced so far and processing continues. Also, the Call Records produced by each backtrack are used to assure that no backtrack is ever duplicated during processing.

The last thing to note in Figure 3 is the Correlation boxes. Since the backtracking caused Line 1 to be resegmented, Character 1 was recreated by Expert 5 as Character 3, and therefore Codepoint 3 is a recreation of Codepoint 1. The Correlation Objects indicate this duplication and help to reinforce the Belief Values of the Objects by combining their Belief Values. This reinforcement is the method through which the output from multiple Experts is combined. For instance, if Codepoint 2a, which by itself was deemed unacceptable, were a recreation of another Codepoint, then the combination of the Belief Value of Codepoint 2a and the other Codepoint may result in a combined Belief Value high enough to be deemed acceptable. This makes it possible for a correct hypothesis to be identified, even if it were never given a completely acceptable Belief Value by a single Expert.

4.2 Belief Values

The ability to backtrack is fine and good, but it would not be possible if there was no way to judge the 'goodness' of one hypothesis against another, or to reinforce the Belief Values of identical Subject Data Objects. The ability to perform these operations is made possible by associating a Belief Value with each hypothesis. The term 'Belief Value' is taken from Dempster-Shafer Belief Network theory. In the strictest sense it is not a true probability, even though it has a domain of 0 to 1, but as in Belief Networks, it serves to give a 'feel' for how good a hypothesis is as compared to other hypotheses.

Initially the Belief Value of a hypothesis is determined in a variety of ways. For a codepoint it is determined by a mapping of measurements returned by the various Classifiers into a Belief Space. In the case of character segments, it may be the amount of overlap of the bounding polygon of a particular segment with the bounding polygons of its neighbors. However, the exact method used to create the initial Belief Value is not as important as the fact that it must be meaningful across all hypotheses of the same type. Still, these methods, although interesting, are not a main focus of this paper. What is of interest to us here is the methods that were used to determine whether or not these Belief Values indicate acceptable or unacceptable hypotheses. To date two approaches have been applied: Serial Probability Ratio Tests (SPRTs) and Dempster-Shafer Belief Networks. We will not attempt a detailed explanation of either of these methods here, the interested reader should see [1] and [2] for more details. We will give a brief overview of each method and then discuss some of the strengths and weaknesses observed in each.

4.2.1 Serial Probability Ratio Test (SPRT)

The SPRT method operates on three key values: the Maximum False Alarm Rate (MFAR), the Maximum Miss Rate (MMR) and the Certainty of a hypothesis. The MFAR and MMR are *a priori* values defined for each Expert used by the Blackboard. The MFAR describes the percentage of times an Expert will call a hypothesis correct with a high certainty when it is incorrect. The MMR describes the percentage of times an Expert will simply return a wrong hypothesis. The Certainty for each hypothesis is essentially its initial Belief Value.

Using the MFAR and MMR, an Acceptance Threshold and a Rejection Threshold are calculated for each Expert. The Certainty is then used to calculate a Certainty Ratio for each hypothesis produced. The comparison of this Certainty Ratio to the Acceptance and Rejection Thresholds then determines whether or not the hypothesis is deemed accepted, rejected or undecided. From the standpoint of the Blackboard, rejected and undecided hypotheses are treated the same, they are backtracked.

On a hypothesis by hypothesis basis, this approach worked very well. As a matter of fact, as will be seen in the Section 5, this approach still gives slightly better recognition than the Dempster-Shafer Belief Network. However, the initial implementation of the SPRT had trouble combining Belief Values from correlated (identical) Subject Data Objects. The combination was accomplished by summing the Acceptance and Rejection Thresholds, and

Certainty Ratio from each Expert-Hypothesis pair in the correlation. If the resulting summed Certainty Ratio was beyond the summed Acceptance Threshold, the hypothesis was deemed acceptable. However, the result of the summing tended to keep the Certainty Ratio from converging on either the Acceptance or Rejection Thresholds, leaving the hypothesis in an undecided state most of the time.

4.2.2 Dempster-Shafer Belief Network

In an attempt to find a way of better combining the Belief Values of correlated Subject Data Objects, we turned to Dempster-Shafer Belief Networks. In a Belief Network there is no concept of calculated Acceptance or Rejection Thresholds, as there was in the SPRT. There is simply a general level of acceptance that is applied to all the hypotheses created. If the final Belief Value of a hypothesis is equal to or above this acceptance level it is deemed acceptable. It's the calculation of the final Belief Value of a hypothesis within the Belief Network, that is of primary interest here.

In a Belief Network, the final Belief Value of a hypothesis at any given node is dependent ultimately on the Belief Values of all the hypotheses directly in the path between that node and the root of the Network (which in our case is the original image), as well as any hypotheses which correlate to those hypotheses. For example, say Character A has a final Belief Value of A. When this character is passed to a classifier, the classifier will return a codepoint, say Codepoint B, with an initial Belief Value of B (this initial Belief Value is still calculated using the mapping to a Belief Space mentioned in Section 4.2). The final Belief value for Codepoint B will be a combination of A, the final Belief Value of Character A, and B, Codepoint B's own initial Belief Value. Thus, the final Belief Value of each hypothesis in the Network reflects the quality of all the hypotheses from which it was created.

This ability to combine the Belief Values of different hypotheses makes the reenforcement of a hypothesis due to correlation very intuitive. The final Belief Values of correlated hypotheses are found by combining the final Belief Values of each individual hypothesis. In other words, if Character A, from above, was to be correlated to a Character C (with a final Belief Value of C), then the new final Belief Values of both hypotheses would be the combination of A and C. Further more, this change in the final Belief Value of Character A would propagate to Codepoint B, thereby raising (or lowering) its final Belief Value. These changes would also propagate to any objects derived from Character C as well.

The test results from our implementation of the Belief Network produced final recognition results slightly below the SPRT. Currently this is thought to be because the mapping of the output of our classifier into a Belief Space is lacking. It is hoped that when we have developed a generic method of mapping the output of classifiers, recognition results will improve, both because the mapping will be more accurate and because we will then be able to add additional classifiers to the Blackboard and combine their output.

5 Test Results

For these results, all of the segmenters and classifiers used as Experts in the Blackboard came from an existing linear OCR system. They were first separated into individual units, so they could act as stand alone procedures. Then different behavioral constants within each were detunned, this allowed us to treat multiple instances of the same segmenter or classifier, with different behavioral constants, as if they were different processes. It should be noted that the behavioral constants in one set of segmenters and one classifier, were given the same values as those in the linear OCR system.

5.1 Recognition Accuracy

Table 1 shows the recognition results from our original linear OCR, the SPRT driven Blackboard and the Belief Network driven Blackboard, over nine sample documents of different sizes.

Table 1: Recognition results from the original linear OCR versus an SPRT driven Blackboard and a Belief Network driven Blackboard.

Image	Glyphs in Truth Model	Correct Matches from Linear OCR	Correct Matches from SPRT Blackboard	Correct Matches from Belief Network Blackboard
Image 1	264	207	212	209
Image 2	271	198	203	213
Image 3	275	211	215	215
Image 4	256	187	197	195
Image 5	1016	776	785	787
Image 6	1299	881	899	900
Image 7	754	600	591	573
Image 8	779	640	640	633
Image 9	588	434	449	446

As Table 1 shows, both the SPRT and the Belief

Network driven Blackboards attained better recognition in almost every case, the exceptions being Image 7 and Image 8. The linear OCR's average recognition over all nine images was 75.6%, where the SPRT driven and the Belief Network driven Blackboards averaged 76.9% and 76.7%, respectively. Although this increase in recognition is very small, it should be stressed that these results were obtained using the same segmentation and classification algorithms in the Blackboards as were used in the linear OCR system. We expect further increases in recognition when segmenters and classifiers from disparate OCRs are added to the Blackboard system.

5.2 Processing Time

The improvement seen above in recognition accuracy came as a result of a trade-off in processing time. Table 2 shows the processing time required by the linear OCR compared to the Blackboard over the same nine images used in the previous section. (Note: Only processing times for the SPRT Blackboard are presented in this section because any run times collected for the Belief Network Blackboard would be skewed due to the current lack of a good mapping from classifier measurements to a Belief Space.)

Table 2: Processing time of the original linear OCR versus the SPRT driven Blackboard.

Image	Linear OCR Processing Time	SPRT Blackboard Processing Time
Image 1	146	742
Image 2	139	599
Image 3	155	686
Image 4	145	704
Image 5	599	4965
Image 6	737	6309
Image 7	453	2799
Image 8	611	2987
Image 9	346	2724

Note: All processing times are in seconds.

Although the processing times required by the Blackboard are significantly longer than the linear OCR system, there are reasons to account for this. Since one of the main ideas of the Blackboard is that it can backtrack and attempt to reprocess paths of deduction that have low Belief Values, this means that the Blackboard could end up performing several times as many segmentations and classifies as a linear OCR system. This reason alone pretty much guarantees that the Blackboard system will always be somewhat slower than a straight linear implementation. But beyond this, there is the overhead required to move data from the Blackboard, to the Experts and back onto the Blackboard. Processing time is also lost while the Blackboard's Controller Code makes decisions as to which Expert should be called next in the processing.

The amount of time the Blackboard spends executing Expert Code as opposed to doing overhead processing can be seen in Table 3.

Table 3: Breakdown of the SPRT processing time: Time spent in Expert versus Blackboard overhead processing.

Total Processing Time	Time Spent Executing Expert Code	Time Spent Executing Overhead Code
742	346	396
599	272	327
686	269	417
704	287	417
4965	1650	3315
6309	1736	4573
2799	1080	1719
2987	1054	1933
2724	796	1928

Note: All processing times are in seconds.

As this shows, at least as much, and usually much more, time is now being spent taking care of overhead processing as is spent actually executing Experts. There is no question in our minds that this overhead processing time can be reduced in magnitude to be near the time spent executing Expert Code. This would still leave the Blackboard slower than a straight linear implementation, but it would bring its execution time down to within more acceptable limits.

6 Future Direction

As this is an ongoing project, there are several areas from which future improvements are expected. The most immediate is the development of a generic method of mapping the output from different classifiers into a uniform Belief Space. This will give us the ability to combine the output of different classifiers during recognition.

In preparation some preliminary tests have been done in adding classifiers other than those derived from our original linear OCR system to the Blackboard. Although their output cannot currently be combined with the output of the current classifiers, examination of the recognition results shows that these new classifiers were able to recognize characters that the original classifiers did not. This is encouraging evidence that, when the output of the classifiers can be combined, overall recognition will be increased.

There are, of course, also plans to add additional segmenters to the Blackboard. However, these additions would require a generic method of acquiring Belief Values for each type of segment they produce, which is an area we have yet to look into with great detail.

Finally, another improvement which has shown promise in initial testing is parallelization of the processing. This will help to bring the overall processing time of the Blackboard down. We have discovered that if an image is first segmented into paragraphs, and then those paragraphs are processed in parallel, the overall processing time is measurably reduced.

7 Conclusions

The Blackboard Architecture, when coupled with Belief Values provides an environment which allows much greater flexibility and much greater intelligence in processing. Thus providing measurably better, and potentially, significantly better, recognition accuracy with no changes to the underlying OCR algorithms. This improved accuracy, however, comes at the cost of longer processing times. We do believe that the processing time can be kept within reasonable limits, both through the reduction of redundant processing and through improvements such as parallelization, making this approach overall superior to a straight linear architecture.

Acknowledgements

Ideas and assistance in many forms from the following kind souls has been essential (and greatly appreciated!):

Dave English, Ivan Bella, Jack Carmody, Trung Nguyen, and Dan Corkill of Blackboard Technology Group, Inc.

References

[1] J. Carmody, Serial Probability Ratio Testing: Sensor and Fusion Models, *Internal Report* June 1996

[2] G. Shafer and R. Srivastva, The Bayesian and Belief-Function Formalisms A General Perspective for Auditing, in *Uncertain Reasoning*, G. Shafer and J. Pearl eds. (Morgan Kaufmann Publishers, Inc. 1990) 482-521.

A Software Framework for Document Image Processing

David S. Brightwell
Department of Defense
9800 Savage Road
Suite 6518
Fort Meade, MD 20755-6518
daveb@last.ncsc.mil

Abstract

A software framework which incorporates many independent image analysis products is described. This framework allows for the rapid development of scalable, extensible, flexible completely automated document image processing systems, and is currently being used successfully by the Department of Defense. The requirements of this framework are described in detail as well as some of the design and implementation details. Some properties of image analysis products which make them easier to integrate into and use in this framework are also presented.

1 Introduction

A complete document image analysis system requires many independent image analysis technologies to be used in conjunction. This is especially true of a system that is expected to operate in a completely automated fashion on documents of varying quality, orientation, and content which may include handwriting, pictures, drawings, many different typewritten language scripts, etc.

It isn't expected that all this capability could be integrated into a single image analysis product. Instead, a framework is being developed that allows for many independent products to be easily incorporated into a single, extensible, scalable system. The requirements for this framework are presented in section 2, and the design is presented in section 3. Some of the implementation details are presented in section 4.

While developing this framework, it became apparent that some image analysis products are better suited than others for integration into such a system. This is mainly due to the way the developers of each image analysis product had envisioned its usage. Some guidelines have been formed for developing image analysis products which should help researchers develop software that can more readily be integrated into this system or similar systems, and ease the transition from a research model to an end product. These guidelines are presented in section 5.

This framework is currently being used in a number of systems, and has incorporated a fairly large number of image analysis products. The current status of the framework and the future plans are presented in section 6.

2 System Requirements

The Department of Defense has requirements for several automated document image processing systems. These systems vary widely in characteristics of the images being processed, and in the throughput they are expected to achieve in terms of image pages per unit time. Some document sets contain multiple languages and script types, handwriting, logos, pictures, maps, drawings, and other information. But some document sets are more limited, and it is known, for example, that all the important information is contained in a single typewritten language.

In addition to differing characteristics in the image data, there also exists differing requirements for ways in which the documents will be identified for retrieval based on the information they contain. Some systems only require a capability to use typical text information retrieval methods on the text resulting from Optical Character Recognition (OCR), but others require capabilities to identify documents based on other information such as the presence of certain logos, signatures, maps, or forms.

In order to provide a foundation for meeting the needs of all these systems, a software framework was developed. This framework allows for the development of systems which are scalable, extensible, and flexible.

2.1 Software Framework

The term *software framework* is being used in this context to mean a collection of software components that are focused on a specific problem domain. The components are either complete programs (UNIX processes), software libraries, or other associated data. The interaction and relationships between the components are well-defined, and to a certain extent dictate much of the high-level software design. The advantage of using a framework is that a single collection of software can be used to produce systems with differing requirements within the same application domain. So software reuse is maximized.

2.2 Scalability

Producing a software framework for image analysis systems requires scalability. The throughput requirements of systems range from being interactive, user initiated analysis of a single image page to fully automated processing of thousands of image pages per hour. The cost of the systems should be fairly proportional to the throughput and functionality requirements. A large scale high throughput system would be expected to employ a much larger amount of hardware and software than a smaller, low throughput, limited capability system.

2.3 Extensibility

The image analysis research community is continually producing new products which are useful, and will help to satisfy the requirements of the users of document image processing systems. As a result, the software framework has a requirement to be extensible to allow for easy integration of these new capabilities. Easy integration means that a minimal (if any) amount of existing software needs to be modified, and currently existing systems can choose whether or not to take advantage of the new capability.

2.4 Flexibility

The most complex document image processing system will need to provide a variety of different processing flows based on the type and content of an image. If it is expected to produce valuable information about the image content regardless of page orientation (landscape, portrait, upside down), language, presence of logos of interest, etc., it must be flexible enough to allow an image page to *choose* which image analysis technologies are appropriate, and ignore those that aren't. For example, it would be a waste of time to run Arabic language OCR on an English language document, or to attempt any kind of OCR on a page containing only a picture or drawing.

3 System Design

The requirements for scalability, extensibility, and flexibility point towards a distributed, service-based architecture. Each of the image analysis technologies, such as page segmentation, language identification, and a variety of OCR for different languages are implemented as services. Each image page can be passed to that service if it is required. The service produces results, which are interpreted to determine which service(s) the image page should be passed to next.

Because each image page is unique, and the results of completed image analysis functions determine the flow of processing for that page, each page needs to remain associated with information about the image analysis functions that have already been performed. This suggests that a compound document structure would be necessary in order to contain this information.

Each document processing system may have different requirements for which services will be run in the presence of certain conditions. For example, one system may want to always perform English OCR while another may want to do language ID initially, and only do English OCR if the language ID service determined that the page contained mostly English text. So there is a need to define the control flow differently for each system being developed. The solution to this problem is to specify processing rules that are read and interpreted at run time. These processing rules can be easily modified to suit the requirements of each system.

3.1 Distributed Services

Incorporating the image analysis technologies as distributed services is the key to scalability. Many of the products being used are very processor intensive, so the use of multiple processors is necessary. One option would be to use large Symmetric Multiprocessing (SMP) systems, and add more processors for those systems with higher throughput requirements; however, SMPs only scale up to a few dozen processors, and can be too expensive for the low-end systems. Another option might be to use a distributed memory multiprocessor system with a high speed low latency interconnection network. However, this really ignores the low-end systems, and it turns out that a standard network of workstations can provide the necessary throughput. Using this model, the systems can scale from a single workstation to a virtually unlimited number of workstations connected via a standard Local Area Network (LAN).

In order to keep the complexity of a system in line with its capabilities and throughput requirements, the services can be either separate processes or can be linked directly with application specific code to be part of a single process. This way the low-end applications can be a single process, but use the same software as the high-end systems which can consist of tens or even hundreds of processes. The decision to link directly with a service, or to include the service as a separate process is made at compile/link time. It is possible to actually do both, and make the decision at run time to use the separate process or not.

All image analysis services in this framework are derived from the same base class, IAService. The RemoteIAService is also derived from the IAService

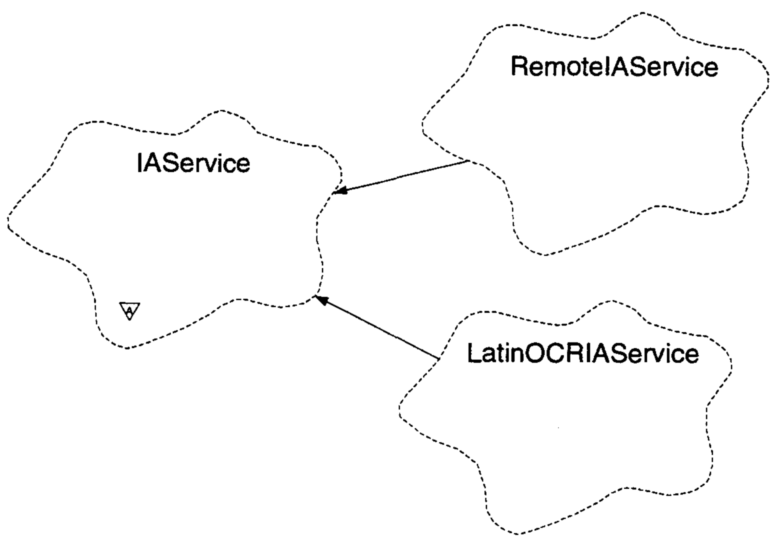


Figure 1: Service Relationships

class, and provides the ability to include a service as a separate process. Figure 1 is a simplified diagram (Booch '94 notation) showing these relationships for the Latin OCR Service [3].

Below is a C++ source code example showing how these different classes may be used. Some of the detail (parameters) are left out, but it demonstrates that the difference between a remote service and a local (linked-in) service is simply a single line of C++ code. The only difference is in the construction of the *ocrService* object. In the first case, it is a local service, and in the second case it is a remote service. All subsequent interaction with the *ocrService* object remains the same. The two lines that construct the *ocrService* object can be either conditionally compiled (with #ifdef), or can both be compiled, and the decision to be remote or local can be made at run-time.

```
IAService *ocrService;
...
ocrService=new LatinOCRIAService(...);
   or
ocrService=new RemoteIAService(...);
...
ocrService->putDocument(...);
```

The distributed service approach also provides both extensibility and flexibility. Extending the framework to include a new technology means developing an additional, completely independent service. In terms of flexibility, the services which will be used by each system can be chosen specifically to match the requirements of that system. If Arabic OCR is not a requirement of a system, then the Arabic OCR service can be left out without affecting any other components of the framework.

3.2 Compound Documents

A well-designed compound document is the key to the extensibility of the software framework. Each image analysis service requires certain information about an image page in order to perform its service. In most cases, this is the image bitmap at a minimum, formatted in some way that is understood by the service. Typically, Tagged Image File Format (TIFF) is used [1]. But in many cases, additional information about the image, produced by other services, can be useful. One example is page segmentation information. A page segmentation service may be able to reliably identify those regions on a page which contain typewritten Latin text. If the English OCR service were able to take advantage of this, then this information should be passed to it. So the compound document acts like a container for information about an image page. When the document first enters the system, it only contains a bitmap representation of the image page. When an image analysis service is invoked, it adds additional information to the document, which subsequent services may be able to take advantage of.

One of the primary difficulties in designing a compound document for this framework was figuring out how to add new services, which produce uniquely formatted data, without affecting any of the existing services. If the compound document was modified to include Unicode characters for Arabic OCR, then this shouldn't require changing the English OCR service, or even recompiling it to use this "new" document [2].

It turns out that the best approach is to use an object-oriented design, and rely on the power of inheritance, polymorphism, and dynamic binding to allow for an extremely flexible compound document [3]. The

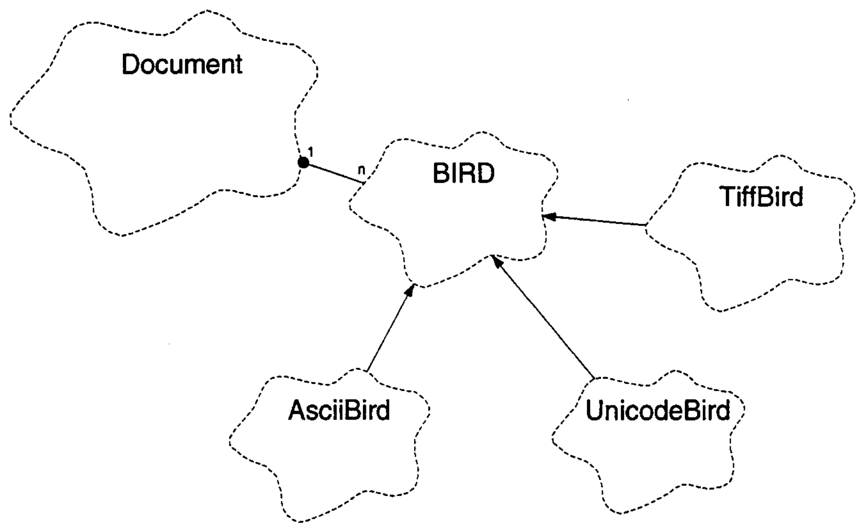


Figure 2: The Compound Document

compound document is a container for a single, general type of data with common methods for identifying and extracting this data. Each specific type of data provided by a service is derived from this more general type. The specifics of this data type are only known to those components which actually need to know about it. Otherwise, it is simply accessed via the general interface it was derived from.

This single, general data type that all image related information is derived from is called *Bitonal Image Related Data* (BIRD). The name had to be very general because it could contain any kind of information related to the original image, and the acronym made it easy to work with. The name for the compound document itself is simply *Document*. Each of the classes derived from *BIRD* has a *Flavor* associated with it that uniquely identifies its type. Figure 1 shows some of the class relationships in Booch '94 notation [3]. It shows that the *Document* contains from 1 to n *BIRDs*, and that *AsciiBird*, *UnicodeBird*, and *TiffBird* are all derived from *BIRD*.

The *Document* class has methods to put *BIRDs* into the document, get *BIRDs* from the document, and test if a *BIRD* of a specific *Flavor* currently exists in the *Document*. In addition, the *Document* has methods called emitStream and absorbStream which allow the *Document* to turn itself into a stream of bytes or restore itself from a stream of bytes. This is necessary for the *Document* to be sent across the network, or stored temporarily on disk.

3.3 Rule-Based Processing Flow

The key to making the software framework flexible enough to be used for many different applications is the rule-based processing scheme being used. There is a program (process) that is central to this called the Executive. The Executive has many responsibilities, including service brokering, rule interpretation, and control flow (based on the rules). The Executive is typically the primary interface into the software framework from an application builder's point of view.

In a large system, using many instances of many services, the Executive is used as a service broker. This is similar to an Object Request Broker (ORB) in the Common Object Request Broker Architecture (CORBA) specification produced by the Object Management Group (OMG) [4]. The Executive keeps all the information about where (on the network) services are located, and how to communicate with each service. The communication between the Executive and the services is typically done via Remote Procedure Calls (RPC) [5]. Services can be added on the fly, and the Executive can also be used to automatically start services, either initially or when they fail.

The Executive reads in a configuration file at startup, and can be signalled to re-read it at any time if it changes. This configuration file provides much information to the executive, including the rules that are to be used for determining which services will be performed on an image in what order. This control flow can be dynamic, so that results from completed services can determine which subsequent services will be performed.

The rules are primarily based on the existence of BIRD flavors in the *Document* or on values of the attributes of the BIRDs. For example, a rule can state that English OCR can be done when Page Segmentation information is available in the *Document*. This would cause a *Document* to be sent to Page Segmentation prior to English OCR. A rule could also state that English OCR should only be done if the LanguageID BIRD is present, and the value of the primaryLanguage attribute of the LanguageID BIRD is Latin.

The Executive learns about BIRD flavors from the configuration file, and needs no specific information about each BIRD flavor (or type). Attributes are available via the general interface, and are made up of standard data types. This means that new BIRD flavors can be added to a system when a new service is deployed, and the Executive doesn't need to be recompiled. In fact, since it can be signalled to re-read a configuration file, it doesn't even need to be restarted in order to add a new service which uses new BIRD flavors.

4 System Implementation

The goal of having an extensible, reusable framework pointed towards an object oriented implementation, and the scalability requirement suggested distributed objects. At the time this framework was first being built, mature CORBA products were not yet available, so this option was not chosen. Open Network Consortium (ONC) RPC was chosen as the primary interprocess communication mechanism, and multithreading was chosen as the way to make the RPCs non-blocking so that the *Executive* could communicate with multiple services simultaneously. C++ was chosen as the implementation language because it allows for easy integration with legacy C code, and it also allows for efficient implementation of many of the object oriented features of interest (encapsulation, information hiding, inheritance, polymorphism, and dynamic binding).

Sun Solaris version 2.x was chosen as the operating system of choice for a number of reasons. The availability of development tools, support for ONC/RPC and multithreading, and availability of image analysis services currently supported or planned for support were among the major reasons. Windows NT has recently been suggested as an alternative platform that has a number of the desired features. Planning is currently underway to possibly provide support for it in the future.

5 Image Analysis Services

Image analysis technologies such as Page Segmentation, Language Identification, OCR, and Image Enhancement are typically provided by third party companies or researchers, and integrated into the framework as services. Although it is possible to integrate almost any capability regardless of how it was intended to be used, there are certain ways in which a service can be packaged, and certain ways a product behaves which can enhance or hinder its usability in this framework.

The most important thing to keep in mind when developing an image analysis product for this framework is that it will be used as part of a larger application, and will be expected to continuously process image pages and provide the results. Many of the other requirements stem from this basic perception. When it comes to designing the product, it is best to think in terms of an Application Programming Interface (API), and what functions or methods will be available as part of a public interface. There are also implementation issues to consider, especially in the areas of resource usage and error handling.

5.1 Requirements

The image analysis services are expected to continually process image pages without interruption or manual intervention. In UNIX terms, they are daemon processes. They respond to requests from a client program, perform a service, return the result, and wait for another request. There is no direct interaction with a user.

Image analysis functions tend to be very processor intensive, and it is usually important to make them run as fast as possible. In a continuously running process, there are sometimes some simple steps that can be taken to help maximize the performance. Separating out functionality that only needs to be done once at process start-up time, and not for each page is an example. These functions can be moved into a set of initialization routines that are only done when the process first starts up. Examples of this are reading a classifier or lexicon into memory, or populating lookup tables.

Typically, multiple copies (invocations) of a service will be running on a single workstation. This is usually the case with some of the higher throughput systems because they are composed of a network of multiprocessor (SMP) workstations. Even if multiple copies of the same service aren't running on the same workstation, many are often run from a shared Network File System(NFS). This will be a problem if the service relies on exclusive access to any resources or uses non-unique temporary filenames. So a service is much more useful in this framework if multiple copies of it can be run on the same workstation using the same executable file.

5.2 Design Issues

When designing a capability to potentially be used as a service in this framework, it's best to think in terms of supplying an Application Programming Interface (API) rather than a complete executable program or set of interacting programs driven by user interaction. An end user program is nice to have for testing and demonstration purposes, but any such program should be considered a driver for the API. Each of the services in the framework are derived from a base *Service* class, and therefore have identical programming interfaces. Rather than requiring each service developer to conform to a certain interface, the framework developers can use the API supplied by the service developer to effectively put a wrapper around it to conform to the base *Service* interface.

One typical inefficiency often observed in APIs can easily be avoided. At the time a *Document* is received by a service, it exists in memory. This means that data buffer based interfaces are much more efficient than file based interfaces. If the primary way to pass an image to the image analysis software via the API is by specifying a filename, then the image data which is already in memory must be written out to a temporary file. The image analysis software will most likely read the file into memory as its first step. If the API allows for a data buffer to be passed instead, then the file I/O is avoided.

One difficult problem in designing image analysis software for this framework is the handling of errors, and the communication of status information. The general guidelines for dealing with these problems are to recover from errors whenever possible, report error information back through the API, and provide a means to log the non-critical status information. The primary behavior to avoid is termination of the process through an *exit()* call, or something similar. An inability to process a particular image page successfully isn't necessarily a problem as long as the error condition or the inability to produce results is communicated back to the calling program through the API.

5.3 Implementation Issues

Since the framework has been implemented almost entirely in C++, it is best for the service providers to provide the API in terms of C function calls or C++ classes (and public member functions).

Memory leaks can be a big problem for server processes that run continuously. They occur when memory is allocated, and not properly deallocated. A memory leak of just a few bytes per invocation (or per image page processed) can cause a daemon process to use up all the system memory in a relatively short period of time. Many tools are available now which detect and report run-time memory leaks, and it is suggested that one of these be used to ensure that memory leaks don't occur.

There are other similar resource issues that come up with server processes as well. Processes have limitations on the number of file descriptors that can be open simultaneously, as well as limitations on interprocess communication resources. Proper allocation and deallocation of these resources needs to be carefully controlled.

Most services developed for this framework have some degree of flexibility in terms of different run-time options they support, different modes they can run in, different classifiers they can use, etc. Setting these options through configuration files or through the API is preferred to environment variables because environment variables are much harder to manage in a large system across a number of remote workstations.

6 Status

The software framework described here is currently being used in at least six distinct application programs with varying requirements. It is being developed in an iterative fashion, and continues to be modified and extended. Most of the current work is focusing on better error detection, recovery, and reporting mechanisms. The current available set of image analysis services numbers about twenty, and continues to grow.

References

[1] *TIFF 6.0 Specification* (Aldus Corporation, Seattle, WA,1992).

[2] The Unicode Consortium, *The Unicode Standard, Version 2.0* (Addison-Wesley, 1996).

[3] G. Booch, *Object-Oriented Analysis and Design with Applications* (Benjamin/Cummings, Redwood City, California, 1994).

[4] R. Otte, P. Patrick, M. Roy, *Understanding CORBA,* (Prentice Hall, Upper Saddle River, NJ, 1996).

[5] J. Bloomer, *Power Programming with RPC,* (O'Reilly & Associates, Inc., 1992).

Foreign Languages and Document Quality

Cherry Blossom: A System for Japanese Character Recognition

Sargur N. Srihari, Tao Hong and Zhixin Shi
Center of Excellence for Document Analysis and Recognition
State University of New York at Buffalo
Buffalo, New York 14228
email: `srihari@cedar.buffalo.edu`

Abstract

A general purpose Japanese character recognition system, Cherry Blossom, has been developed at CEDAR in past years. It is designed to recognize Japanese document images in low resolution or with poor print quality. The system includes modules for page skew correction, document segmentation, text segmentation, character recognition and postprocessing. The API code for each module has been developed so that each module can be tested as a standalone program or can be called in large application systems such as a document indexing and retrieval system.

In the character recognition module, two classification methods, the nearest-neighbor classifier and the subspace method, have been integrated in an efficient way. The speed of character recognition is about six characters per second from the original one character per second. New techniques, including dynamic feature selection, incremental nearest prototype search and visual similarity analysis, have been developed to speed up character classification while keeping high accuracy. A linguistic postprocessing module improves the accuracy of character recognition by exploiting linguistic context. It was trained on a Japanese text corpus with above 70 million characters.

The JOCR system has also been adapted to recognize Chinese documents. Because the similarity between Chinese and Japanese character sets, the Chinese recognizer can be developed rapidly by simply training on font images and by sharing the information of the well-trained Japanese classifier. A Unicode-based OCR for Far East Languages is being developed based on the JOCR system.

We also conducted experiments on representing Japanese OCR results in HTML files for evaluation, error correction and reading purposes. A real-time GUI program which demonstrates different aspects of the system for Japanese and Chinese document recognition is now available on SunOS.

1 Introduction

The Japanese optical character recognition project at CEDAR is to develop a prototype system for machine-printed Japanese document recognition [33]. Intensive development and research have been conducted to improve accuracy of document recognition on degraded documents such as facsimile pages or images scanned at low resolution.

The Japanese character set is a combination of several scripts: Kanji (Chinese characters), Kana (Japanese consonant and syllabary characters), Roman characters and Arabic numerals. Kanji are Chinese ideographs which were introduced into Japanese. In the Japanese standard, JISx0208-1990 [25], there are $6,879$ characters. Among them, $6,355$ are Kanji, divided into two distinct sections – JIS Level I and Level II; where Level I spans $2,965$ of the most frequently used Kanji characters.

Comparing with the task of developing an English document recognition system, there are more difficulties on developing a Japanese OCR system. Some of those difficulties are:

1. Large variety of page layouts in Japanese documents; For example, the direction of a text line can be horizontal or vertical.
2. Large number of character categories; As described above, there are more than 3,000 character categories which are needed to be recognized.
3. Structural complexity of character patterns; Most Japanese characters are Kanji. The stroke structure of a Kanji usually is complex. Many characters have very similar stroke structure.
4. Large variety of character shapes due to different typeface and image quality in machine-printed documents.

As in systems for English document recognition, a Japanese document recognition system has three major components: layout analysis, character segmentation/recognition, and postprocessing. Layout analysis techniques for Japanese documents [2, 26] and multi-lingual documents [18, 35] have been developed. Given an image of a document page, layout analysis will locate text blocks and further segment them into text lines which can be either vertically or horizontally oriented. Because word boundaries are usually not visible, character images are directly segmented from the text lines. Connected-component analysis, projection profile analysis and feedback of character recognition result can be used for character segmentation[2, 3, 27].

Because Kanji characters are complex in their stroke structure and many characters share structural similarities, feature descriptors are typically with large dimensions in order to effectively discriminate so many Kanji characters. Stroke features (such as the direction contribution features [2]) and background features (such as surrounding area features [32, 30]) are two types of statistical features which have been proved very useful. Many sophisticated similarity/distance measures have been proposed to achieve more reliable performance of character recognition. Methods such as the multiple similarity analysis[17] and the modified quadratic discriminant functions [20] have been applied to hand-printed and machine-printed character recognition.

Two classifiers have been designed for character recognition in the system [14]. One is the Minimum Error Subspace method; another is the Nearest Neighbor classifier [11]. The features are called "Direction Contributivity" features [2, 28]. The NN classifier has been generalized to use the distance measurement defined in the Minimum Error Subspace method which has been proven to be more accurate than Euclidean distance. Because its confidence calculation is trustworthy, for a character image, the NN classifier provides only the first decision if its confidence score is high; otherwise, the top 5 choice list is output. Classifiers were trained on 200 ppi character images from CEDAR's Japanese character image database [16], and were tested on 200, 300 and 400 ppi images. On 200 ppi character images from CEDAR's test set, the accuracy of the NN classifier is 95.9%. The speed of character recognition varies from 3 to 6 characters per second, depending on image quality.

A linguistic postprocessing module has been added to the system in order to further improve the accuracy of character recognition by exploiting linguistic context. It was trained on a Japanese text corpus with above 70 million characters.

The JOCR system has also been adapted to recognize Chinese documents. Because the similarity between Chinese and Japanese character sets, the Chinese recognizer can be developed rapidly by simply training on font images and by sharing the information of the well-trained Japanese classifier. A Unicode-based OCR for Far East Languages is being developed based on the JOCR system.

In the system, we also developed methods for visual context analysis, such as image-based character image clustering and inter-character similarity analysis [12, 9]. The results of visual context analysis can be used for OCR error correction [10].

The system is implemented on a SPARC Workstation under SunOS. Given a TIFF image which is a text block or a document page, the system will execute its modules and save the details of analysis and recognition into a file in a special format.

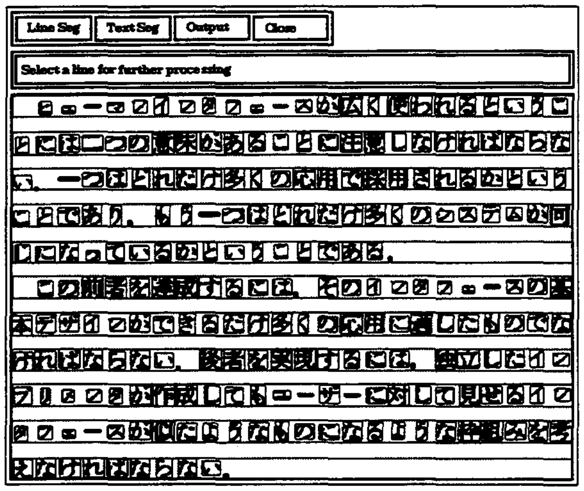
Two X-Window-based GUI programs for system evaluation have been designed using C/C++ and Motif. By using those programs, deskewing, text segmentation, line segmentation and character segmentation can be performed either manually or automatically. For each segmented character image, the top 5 choices are provided by a character classifier. The programs allow a user to use a mouse to manually correct mistakes in segmentation and recognition.

Although these GUI programs are very useful for system evaluation, much work is needed to modify them for the purpose of evaluating different aspects of our OCR system. In order to rapidly prototype visualization and evaluation tools which are more flexible and less platform-dependent, we recently developed a framework to convert the recognition results into a set of HTML files. HTML handles symbols and images uniformly and allows hyperlinks between them. A Web browser such as Netscape Navigator comes with multi-lingual support and can be run on many platforms. Scripts can be easily written to convert recognition results in any level into HTML files so that different aspects of the system can be examined within a local network (Intranet) or on the Internet. Much effort in GUI design, multi-lingual and multi-platform support can therefore be avoided.

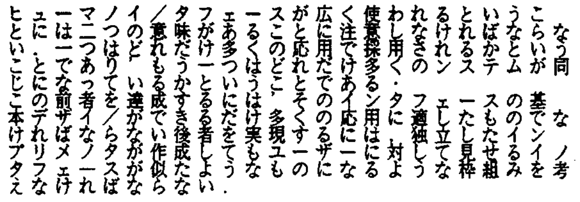
Section 2 gives an overview of the JOCR system design, including its modules for page deskewing, page segmentation, text alignment detection, line segmentation and character segmentation. Section 3 describes the design of two classifiers for character recognitions. Section 4 discusses extending the JOCR system to be multi-lingual recognition system. In section 5, a postprocessing module is described and in section 6, we present an experiment on evaluating Japanese document recognition using HTML. Finally, conclusions and future directions of this project are briefly discussed in section 7.

2 The Overview of the System Design

Cherry Blossom is a prototype system for Japanese document recognition developed at CEDAR in past years. This system is designed to meet several challenges of machine-printed Japanese document recognition: the wide variations in data quality (facsimile, photocopy, newspaper etc.), low scan resolution (e.g., 200ppi), large character set (about 3,000 Kanji characters in the first level JIS code) and the variety of font and point sizes. Given a Japanese document page, it can deskew the image, segment the page into blocks, discriminate text/non-text blocks, determine the alignment of a text block, segment text regions into lines and further into character images, and recognize character images. The system consists of API modules including deskewing, page segmentation, text segmentation, character recognition and postprocessing. In the system, OCR results can be encoded in JIS, Shift-JIS, EUC and Unicode.



(a)



なう同　な　ノ考
こらいが　基でンイを
うなとム　ののイるみ
いぼかテ　スもたせ組
とれるス　一たし見枠
るけれン　ェし立てな
れなさの　フ適独しう
わし用く・タに　対よ
使意採多るン用はにる
く注でけあイ応に一な
広に用だでののるザに
がと応れとそくす一の
スこのどこ，多現ユも
一るくはうはけ実もな
ェあ多ついにだをてう
フがけ一とるる者しよい．
タ味だうかすき後成たな
／意れもる成でい作似ら
イのど　い違がながな
ノつはりてを／らタスば
マ二つあっ者イなノ一れ
一は一でな前ザぼメ ェけ
ュに .とにのデれリフな
ヒといこじこ本けプタえ

(b)

Figure 1: Character segmentation and recognition. (a) shows line and character segmentation result of a text block. (b) is the recognition result displayed in a kterm window.

The design of the modules for deskewing, page segmentation and text segmentation are described as follows. The character recognition module will be presented in the next section.

To detect and correct skew in a scanned document image, a fast directional profile analysis method is used[23]. The range of skew angle handled by the method is about ±30 degree. After detecting skew angle, a simple affine transformation is applied for skew correction. The skew detection program works well on document images which often contains graphics, tables and figures.

The skew detector was tested with 467 artificially skewed images, with varying amounts of noise, fragmentation, black pixel density and ratio of text to graphics. The document analyzer achieves 97% accuracy within ±1 degree of the actual skew on good quality document images, and 86% on noisy or complex documents. The average time required for skew detection is 2 cpu seconds on a SPARC-10.

The objective of page segmentation is to decompose a scanned document image into regions which contain homogeneous entities, such as text, graphics and half-tones. A ***local-to-global segmentation algorithm*** has been designed in *Cherry Blossom* [21]. The page segmentation module was tested with 200 Japanese and English documents. The performance of the document segmentation ranges between 92% completely correct segmentations and 94% partially correct segmentations. On average this module requires 27 cpu seconds on a SPARC-10.

New methods for page segmentation are being investigated. They include smoothed-run-length analysis, background pattern analysis and minimum spinning tree analysis methods. A hybrid method which can achieve better performance on different types of Japanese document pages is being developed.

The text segmentation module is designed with the objective of segmenting a given text block into line images and further into character images [22]. In the system, a five-stage procedure – image quality estimation, character size estimation, text alignment detection, line segmentation and character segmentation, are implemented to fulfill the task of text segmentation.

At the stage of image quality estimation, the system determines aspects of image quality, which will be useful for line and character segmentation. At the stage of character size estimation, character size is estimated globally based on histogram plots of connected components. In clean images, the highest peak in the component histogram corresponds to the width of full pitch characters. In noisy images, this component histogram is smoothed and the mean width of the histogram is evaluated. At the stage of text alignment detection, methods from two different approaches are implemented to determine whether text is vertically or horizontally aligned. The first approach is based on the analysis of a minimum spanning tree of connected components [18]; and the second one is based on projection profile analysis.

Projection profile analysis is also used at the stage of line segmentation. For character segmentation, character components are initially located by performing a projection profile analysis of black pixels and identifying white spaces as component separators. After local character size estimation, a rule-based approach is then used to partition components into three groups. These groups identify single component characters, partial character components and touching characters (or multi-character components). The sub-character components are merged based on a few simple rules to form complete characters. Touching characters are split based on further analysis and heuristics. At this stage a character recognizer has also been used to validate proposed splitting points.

The text and character segmentation is evaluated with respect to several criteria. Text segmentation performance varies between 98.5% for good quality images and 94% for degraded images. Simple connected component analysis is used for high quality printed text, while more complex analysis is applied for low quality document images.

3 Japanese Character Recognition

Two independent character recognition methods have been developed to recognize more than 3,300 Japanese characters. One classifier uses the minimal error subspace method; another is a fast nearest-neighbor classifier. After implementing and testing many types of feature sets reported in Japanese OCR literature, two feature sets were chosen for further testing because their performance on low quality images was encouraging. These are the Local Stroke Direction (LSD) and Gradient, Structural & Concavity (GSC) feature sets. These feature sets are described below, followed by discussing the design of the minimal error subspace classifier and the nearest neighbor classifier. Results of both classifiers as well as their integration will also be reported.

3.1 Local Stroke Direction Features

Local stroke direction (LSD) features are based on direction contributivity analysis [2]. When given a character image, its LSD feature vector can be computed as follows [28] (see Figure 2 for example): first, for each black pixel, compute its directional run-length for each of four directions in Figure 2 (b) and normalize it as a ratio to the total run-length in all directions; second, partition the image into $n \times n$ areas and compute the directional run-length of each area as an average of the pixels in the area. Here, we choose that n equals 8. Therefore, the size of LSD feature vector is $4 \times 8 \times 8 = 256$. In Figure 2 (c), the values in the LSD feature vector are scaled to integers at the range of 0 and 255 so that they can be visualized as a grayscale image.

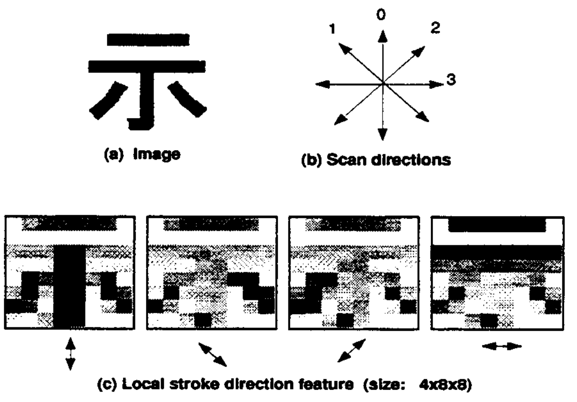


Figure 2: A Kanji character image and its *local stroke direction* (LSD) feature vector

3.2 Gradient, Structural and Concavity Features

The gradient and structural features encode local structure, while the concavity features are global descriptors extracted from binarized images. A gradient map is constructed from the normalized image, by estimating gradient value and direction at each pixel. Histograms of directional features are recorded in each region – indicating the presence (or absence) of a small number of oriented ranges of gradients in the region. Directional histograms from each region are concatenated into a fixed-length *gradient feature* vector. *Structural features* are computed from the gradient map by examining similarities in gradient direction in a local neighborhood of each pixel. These features record curvature in an approximate fashion. Curvature histograms indicating presence of changing orientation of character contours are estimated in each region. These histograms are concatenated from each region in a fixed-length structural feature vector. Gradient and structural features have been described in more detail in [34]. *Concavity features* are coarse global descriptors and are of three kinds: pixel density, large stroke, and true concavity. The large stroke features encode horizontal and vertical strokes in the image. Concave regions enclosed with a character are also recorded.

Concavity features have been described in detail in [5].

3.3 Minimum Error Subspace Classifier

Subspace methods have been a major field of study in pattern recognition [29, 6], particularly for classification and feature subset selection. The minimum error subspace classifier is a discriminant function derived from the Karhunen-Loeve expansion. It is given by:

$$g_j(X) = |X - M_j|^2 - \sum_{i=1,k} \{\phi_{ij}^T (X - M_j)\}^2.$$

Here, $g_j(X)$ defines the discriminant classifier for the j^{th} class, given an input measurement/feature vector X; M_j is the mean vector and ϕ_{ij}^T is the transpose of the i^{th} eigenvector of the covariance matrix for the j^{th} class; k is chosen as the dimension of the subspace defined by the dominant k eigenvectors. Based on our experiences, k is set to 5 in our experiments. An unknown test feature vector is evaluated with these discriminant functions for each class, that is, $g_j()$, for $j = 1, 2, ..., C$, where C is the number of classes. The class of X is determined as that of the discriminant function for which the residual error is least, that is: $g_J(X) \leq g_j(X)$, $\forall j \in \{1, 2, .., C\}, j \neq J$.

Projections of an unknown test vector on the subspaces defined by the dominant eigenvectors of each class are subtracted from the projections on the entire eigen-space. The subspace of the class which best represents the unknown feature vector, results in the least residual error (and results in the least difference in projections between the whole space and the subspace of the principal eigenvectors). The class corresponding to the subspace with least residual error is chosen as the class-identity of the unknown test vector.

This classifier can also be derived from the modified quadratic discriminant function (MQDF)[19], and can be interpreted as a quadratic discriminant classifier.

The training phase of this classifier requires the computation of the covariance matrices and their eigen-decomposition. For each class, the eigenvectors are then sorted in decreasing order of the corresponding eigenvalues. The mean vector M_j is evaluated for each class. Only the dominant eigenvectors (k, in this case) are used in the subsequent processing. During testing, the unknown or test feature vector is projected onto the subspace defined by these dominant vectors, for each class. The term $|X - M_j|^2$ defines the projection of the test vector onto the whole eigen-space. Hence, the class of an unknown vector is determined by the best subspace projection, given by the least residual error.

The classifier had been trained and tested on the CEDAR dataset. Using the LSD feature and the GSC feature, results of the ME (Minimum Error) Subspace Classifier are listed in Table 1 (a).

(a) ME Subspace Classifier ($k = 5$)					
LSD			GSC		
Top 1	Top 5	Top 10	Top 1	Top 5	Top 10
93.48%	97.86%	98.45%	93.93%	97.63%	98.19%

(b) NN Classifier (using Euclidean distance distance)					
LSD			GSC		
Top 1	Top 5	Top 10	Top 1	Top 5	Top 10
92.73%	97.33%	98.01%	93.06%	97.88%	98.60%

Table 1: Character recognition performance of different classifiers on CEDAR's 200 *ppi* dataset (57,571 images from 3,319 classes).

3.4 A Nearest-Neighbor Classifier

A NN (nearest neighbor) classifier has been designed for Japanese character recognition. New algorithms for prototype reduction, hierarchical prototype organization and fast NN search are implemented in the classifier [11, 14]. In these algorithms, k-nearest/farthest neighbor lists are used to estimate the distribution of samples in the feature space. Given a set of samples, $P = \{p_0, p_1, ..., p_{n-1}\}$, for each sample p_i, its k-nearest neighbor list, $N_i = \{n_{io}, n_{i1}, ..., n_{i,k-1}\}$, and its k-farthest neighbor list, $F_i = \{f_{io}, f_{i1}, ..., f_{i,k-1}\}$, have to be pre-computed.

Given a training set, prototype reduction leads to the selection of a subset of samples which can represent the whole training set. Figure 3 illustrates the process of prototype selection for a class. The subset can be generated by deleting redundant samples from the training set. By checking the pre-computed nearest neighbor list of each training sample, the prototype reduction algorithm decides whether a training sample can be deleted without negative effect on the correct classification of itself and other samples. Previous methods, such as Hart's *condensed nearest neighbor rule* (CNN) and Gates' *reduced nearest neighbor rule* (RNN), have to call the procedure of classification iteratively to test whether the deletion(or adding) of a prototype affects the correct classification of the other prototypes[8, 7]. The advantage of the method here is to avoid such an iterative process.

The NN classifier using LSD feature and Euclidean distance measure was trained and tested on the data sets from CEDAR's CDROM. There are

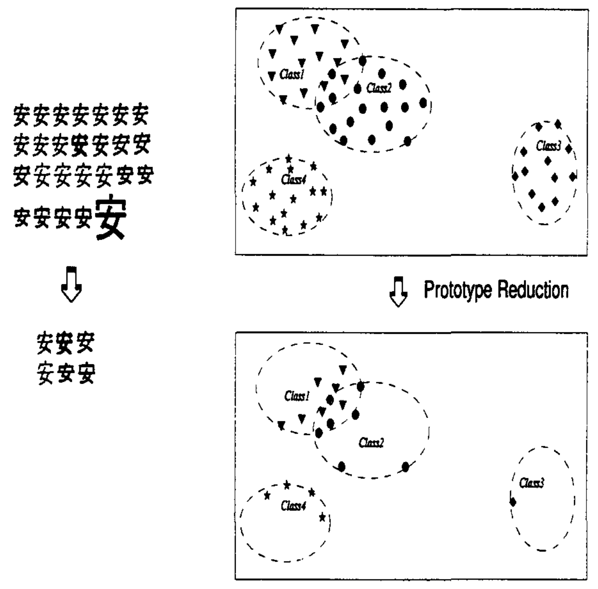


Figure 3: Prototype reduction

118,417 character samples from 3,354 classes in the training set. The number of prototypes derived by prototype reduction is 24,273.

A brute-force NN algorithm suffers from its computational complexity because every prototype has to be matched with the input pattern to see whether it is close to the input pattern. To speed up NN classification, several methods are proposed. An inter-character visual similarity constraints is firstly used. The preliminary experiment shows very promising. Given a text block with 1,721 segmented character images, the system takes less than 17 minutes to recognize it (originally it needs about 29 minutes). In another word, we can improve our classifier to 0.58 second per character from originally 1.0 second per character with the only change.

The idea behind the new method is simple. In a Japanese document page, many characters occur more than one time. An image clustering procedure can be used to group those different image entries of the same character. After image clustering, the NN classifier will recognize characters in a group. After recognizing the first entry of a group by matching with all prototypes, the system will recognize the rest of group members by matching with a small number of prototypes which are in the best match list of the first entry. The size of the reduced prototype set can be less than one tenth of the number of all prototypes. Sometimes the clustering procedure may cause errors so that visually similar characters from different JIS categories are grouped together because in Japanese there are many characters which are very similar in shape. The method can tolerate this kind of clustering errors very well. On the example text block with 1,721 segmented characters, 508 clusters were generated. While it takes 0.95 second to recognize the first entry of each group, it needs only 0.25 second to recognize each of the other entries in a group. On average, 0.58 second is needed for each character. The accuracy of classifier is almost the same as the original classifier.

A dynamic feature selection for feature matching has been developed. In our implementation of the LSD feature, there are 256 inter elements. The size of the feature vector is fairly large. The NN classifier runs slowly because it has to do comparison on each of those elements between the test patter and a prototype. It will be much faster if only a small subset of feature elements can be selected for matching. Given a character image, after extracting its LSD feature vector, we can dynamically select such a subset of element according to the pivotal directions in the feature extraction algorithm.

we have also developed a fast NN search algorithm, FNN. The basic idea is to avoid unnecessary comparisons. Suppose there is a test sample X to be classified (see the triangle in Figure 4). It has to compare with prototypes from the prototype set. After the comparison between the input pattern X and a prototype P_i, we know that the distance between them is very small. For those prototypes in P_i's k-farthest neighbor list, such as F_{i1},F_{i2} and F_{i3} shown in Figure 4, without going further to compare each of them with the input sample X, we know the distance will be very large and therefore they can not be in X's k-NN list. Similarly, after the comparison between X and a prototype P_j, we know that the distance between them is very large. For those prototypes in P_j's k-nearest neighbor list, such as N_{j1},N_{j2},N_{j3} and N_{j4} shown in Figure 4, without going further to compare each of them with the input sample X, we know the distance will be very large and therefore they can not be in X's k-NN list.

An incremental feature match algorithm was also designed which firstly dynamically select a small number of feature elements (about one sixth of the feature size) and do NN search, then use a larger number of feature elements to refine the NN search result on a small number of prototypes with small match distance.

In order to further speed up the search process, The prototype set can be organized into a hierarchical representation. The prototype set is divided into two levels. The first level is the so-called "*centroid set.*" For a centroid, there may be a set of prototypes which are stored in the second level. The set for a centroid is called as its "*package.*" A centroid can approximately represent those prototypes in its

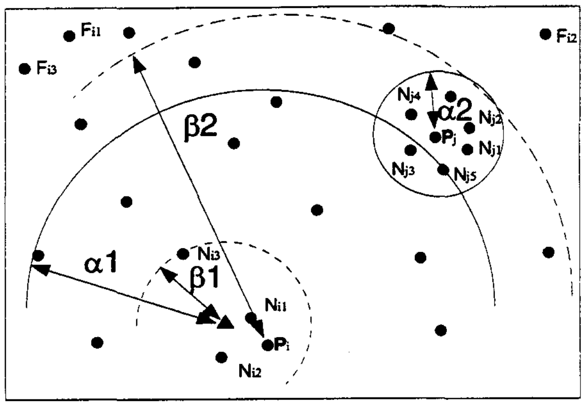


Figure 4: Speed up nearest neighbor search by avoiding redundant comparisons

package. The process to create such a hierarchy is to pack prototypes for selected centroids.

After packing prototypes, the k-nearest neighbor list and k-farthest neighbor list of each prototype in the centroid set C will be computed. The NN classifier can be described as a two-step procedure. Given a test sample X, its approximate k-nearest neighbors $\overline{N}_X$ in the centroid set C can be calculated using the search algorithm described above. Then the prototypes in the packages of those centroids in $\overline{N}_X$ will be compared with X to generate the final k-nearest neighbors N_X for the sample.

In the NN classifier, distance between a test pattern and a prototype can be measured in many different ways, from simple metrics to sophisticated ones. We first tried the Euclidean distance, then changed to use both the Euclidean distance and the distance defined by the *ME Subspace* method.

Using the Euclidean distance, the performance of the *FNN* classifier on the test set is shown in Table 1 (b). The prototype set used in our NN classifier has 24,273 prototypes. The number of centroids is 1,919 if we define the package size as 50. By comparing with 2,195 prototypes on average the system can achieve 92.7% top 1 accuracy. Performance of the classifier using the GSC feature set is also reported in the table.

By using the Euclidean distance, the accuracy of the NN classifier is lower than that of the subspace classifier although there are multiple prototypes per class in the NN classifier while there is only one prototype for each class in the subspace method. It seems to us that the distance measurement defined in the subspace method is much more accurate than the Euclidean distance. Therefore, we designed a new NN classifier to use the subspace distance. Because the subspace method is time-consuming, we proposed to use both the Euclidean distance and the subspace distance in the classifier. Given a test pattern P and n prototypes, steps to recognize P are:

- Step 1 – find first n_1 closest prototypes of P among n prototypes based on Euclidean distance;
- Step 2 – refine the distance scores of the pattern P with n_1 prototypes by using the subspace distance;
- Step 3 – re-rank n_1 prototypes by their refined distance scores and output the top 10 choices.

In our experiments, n is 24,273 and n_1 is set to 100. In the design, instead of computing the subspace distance between the test pattern and each of 3,300 classes, only n_1 subspace distances have to be computed. The Euclidean-distance-based NN algorithm for *Step*1 can be a brute-force NN algorithm or the fast NN search algorithm described above.

Originally each class in *ME Subspace* method has a set of eigen vectors and a mean vector. In the modified NN classifier, we also generalized the subspace method to use multiple prototypes for each class. In the new classifier, each class has a set of eigen vectors and a set of prototypes. This extension gives the capability of using the subspace method in NN classifier so that reliable decisions can be made based on majority voting.

The new NN classifier was designed in order to integrate the NN search method with the *ME Subspace* method. It has also been trained and tested on 200 ppi datasets from CEDAR's character image database. Table 2 lists results of the classifier in which *Step*1 was implemented as a brute-force NN classifier and the FNN algorithm described above. By using the FNN algorithm, each test image was compared with about 6,203 prototypes out of 23,449 prototypes. The LSD feature is used in the classifier here.

To generate reliable confidence information for OCR decision, we proposed a method to compute the confidence score for each choice from the NN classifier [14]. In the method, confidence scores are calculated based on both majority voting result and distance information.

Given a test pattern t, a NN classifier can provide two types of candidate lists: one is the nearest prototype list

$$P = p_1, p_2, ..., p_i, ..., p_n$$

where p_i is a prototype. The identity of p_i is $JIS(p_i)$ and the distance between p_i and t is $DIST(p_i)$. In

	Number of comparisons	Accuracy top1	top5	top10
Brute-force NN search	23,449	95.92% (55,221/57,566)	98.09% (56,471/57,566)	98.57% (56,748/57,566)
Fast NN search	6,203	95.83% (55,166/57,566)	97.97% (56,401/57,566)	98.43% (56,667/57,566)

Table 2: Performance of NN classifiers using Euclidean distance and Subspace method. The only difference between two classifiers is that in *Brute-force NN* search the test pattern has to compare with every prototype while only a portion of prototypes are compared in the *Fast NN* search algorithm.

P, prototypes may have the same identity. Another is the choice list

$$C = c_1, c_2, ..., c_j, ..., c_m$$

where c_j is a JIS code. In C, choices have to be distinct. For each c_j, $DIST(c_j)$ is defined as $DIST(p_i)$, where p_i is the first prototype in P so that $JIS(p_i) = c_j$.

For each c_j, we can compute its confidence as

$$CONF(c_1) = k/3.0$$

where k satisfies

$$JIS(p_1) = ... = JIS(p_k) \neq JIS(p_{k+1})$$

and otherwise

$$CONF(c_j) = DIST(c_1)/DIST(c_j)$$

Based on above experiments, a new system configuration is implemented for the character recognition. Significant improvement on classification speed has been achieved. The improvement is the result of the integration of a flexible control structure, image clustering algorithm, prototype packing method, dynamic sub-feature selection method, incremental nearest-neighbor searching algorithm and the modified quadratic discriminant function. This progress makes our system a step closer to a real-time system. Figure 6 is the flowchart of the character recognition module in the current system. As illustrated in the figure, although there are seven steps for character recognition, character images do not have to go through all of them. For a character passes all seven steps, it takes about 0.4 second. Many character images in large clusters only have to pass Steps 1, 2 and 6 and only have to compare their feature vectors with 500 prototypes. This takes less than 0.1 second.

Figure 5 shows character recognition result of the NN classifier on images of different quality. In the figure, only the first choice is provided if its confidence is high; otherwise, the top five choice list is output and further postprocessing is needed to make a decision.

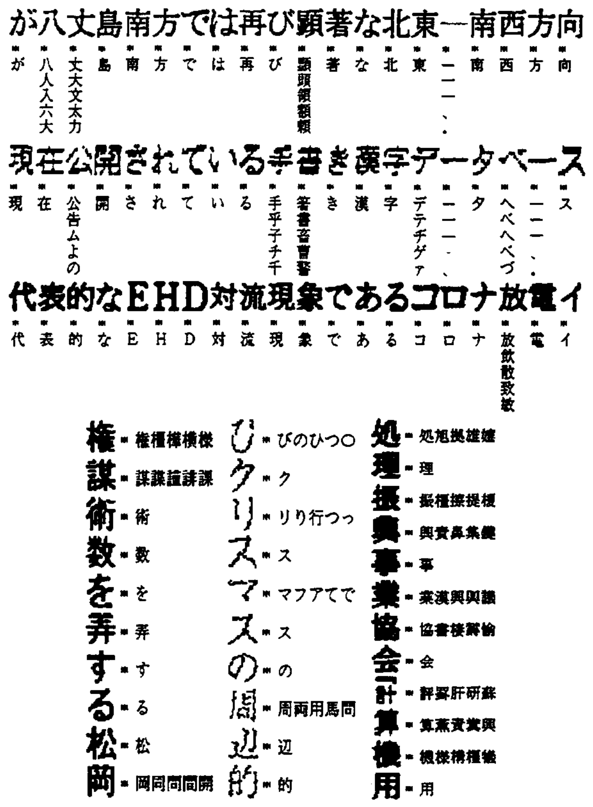

が八丈島南方では再び顕著な北東一南西方向
現在公開されている手書き漢字データベース
代表的なEHD対流現象であるコロナ放電イ

Figure 5: Character recognition. The *NN* classifier is used. For each character image, the classifier gives a candidate list. Because the confidence information from the *NN* classifier is reliable, the classifier provides only top first candidate if its confidence is high.

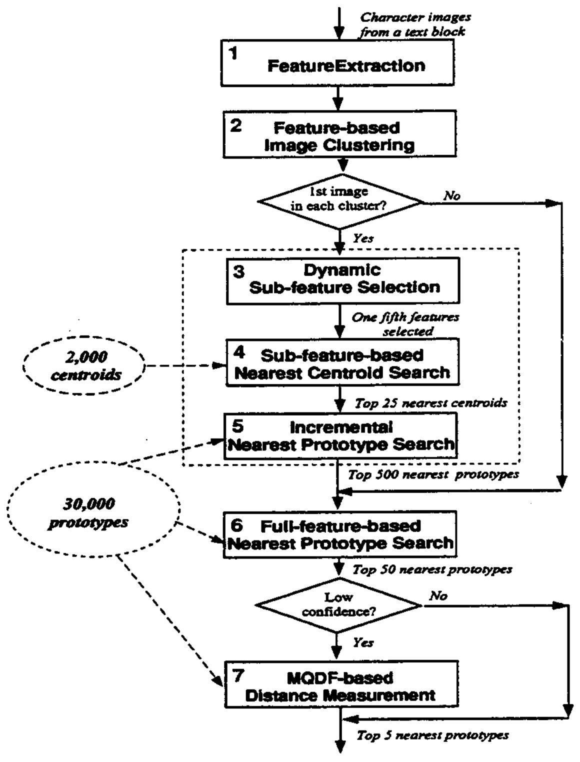

Character images from a text block
1 FeatureExtraction
2 Feature-based Image Clustering
1st image in each cluster?
No
Yes
3 Dynamic Sub-feature Selection
One fifth features selected
2,000 centroids
4 Sub-feature-based Nearest Centroid Search
Top 25 nearest centroids
5 Incremental Nearest Prototype Search
Top 500 nearest prototypes
30,000 prototypes
6 Full-feature-based Nearest Prototype Search
Top 50 nearest prototypes
Low confidence?
No
Yes
7 MQDF-based Distance Measurement
Top 5 nearest prototypes

Figure 6: The framework of character recognition in the JOCR system

4 Multi-Lingual Document Recognition

Based on our developed Japanese document recognition system, we also investigated recognition for documents in other languages such as Chinese and Korean.

A new method is proposed for rapidly prototyping a recognition system for documents written in a new language (such as Chinese) based on the well-trained JOCR system. The new recognition system uses only the font samples and does not need to be trained on real world image samples. As a natural result, a Unicode-based OCR for Oriental languages (such as Chinese, Japanese and Korean) has been developed based on our JOCR system.

It usually takes long time and high cost to develop a high accuracy OCR for documents in a different language. Much effort is usually on training the OCR on many scanned document images. Although font images are easier to be collected than real world document samples, OCR using only font images usually can not achieve high accuracy because the number of font samples is very limited. An accurate distance measurement such as MQDF needs more than 100 character samples for each class.

Can we adapt our JOCR for other Oriental languages such as Chinese by using only the available font images? We did some experiments to answer the question. We collected 4 different Chinese fonts from Internet. They are Song, FangSong, Hei and Kai. By using them as prototypes, the accuracy of the NN classifier is not high. Based on the fact that the Kanji characters in Japanese are very similar to the Hanzi characters in Chinese, we proposed a method to use the information from our well-trained Japanese OCR to improve the performance of the Chinese OCR. Firstly, we conducted visual similarity analysis between Chinese and Japanese font images. A mapping table has been derived. For those Chinese characters which have visual equivalents in Japanese, their corresponding Japanese prototypes can be used to recognize Chinese characters.

The method above has been implemented so that the system can also achieve high recognition accuracy on document in Chinese as same as in Japanese. A multi-lingual document recognition system using the international standard, Unicode, is being also developed.

5 Post Processing

Character segmentation and character classification are error-prone. To improve recognition result, post-processing methods can be applied. During the post-processing stage, various contextual constraints can be used to detect and correct OCR errors. In our JOCR system, we are developing methods for two specific tasks: recognition error correction and segmentation error correction.

In the task of recognition error correction, a top first choice has to been generated for each character which is suspected to be classified incorrectly. The confidence score provided by the classifier can be used to decide whether postprocessing is needed for the character. Two types of constraints can be considered: one is the linguistic constraints; the other is the visual similarity constraints.

One of the simplest types of linguistic constraints is the character bigram. A character bigram can be defined as a tuple, (c_1, c_2, f), where (c_1, c_2) is a character pair, and f is the frequency of c_1 and c_2 being together. Bigram data can be trained from a large text corpus. For a character image with several alternatives generated by a classifier, bigram information can be very useful to find a choice from the alternative list which best fit its local context (e.g., the characters before or after the character).

In past years, we have conducted experiments on using bigram information for English recognition error correction. The methods have been implemented in our JOCR system. For Japanese, a bigram database with 242,682 bigram entries has been collected from a large Japanese news corpus.

The post processing is done only when it is needed. For a text segment in the original image, if there is no character with a high enough confidence can be found in the fonts, then we say that the "truth value" of this text segment is not yet determined and that the post processing needs to be performed on this text segment. Because the top five choices chosen as possible candidates for any text segment are always available, so the post processing can be done as the following. First of all, we take the first choice of the five candidates and compare it with each of the five choices of its left or right neighbor text segment respectively, if there is such a text segment. Then we can get five "distance values". These distance values are defined according to the frequency of the two characters appearing together in documents such as books and newspapers of the language. Once we have these distance values, we can use a strategy to find a most possible case that the candidate characters should appear in the text of our special document.

Visual similarity constraints can also be useful for recognition error correction. The intuition behind is that: visually equivalent character images should be recognized as the same characters. By examining OCR result, we found that this constraint sometimes is violated. Different alternative lists can be provided by a character classifier for several visually almost equivalent character images. Previously, we developed methods to cluster character images based on their visual similarities [12]. The clustering algorithm has also implemented as a module in the JOCR system. After clustering, a procedure of consistency checking will be applied to those character images in the same cluster to correct potential errors.

Another types of errors can be corrected by post-processing is character segmentation errors. For those character images with low recognition confidence and with very large or small width/height aspect ratio, segmentation errors may happen. Re-segmentation can be conducted by splitting or merging, then character classification can be applied to get new recognition results. Progress has been made to design a postprocessing procedure to correct segmentation errors.

6 Evaluation of OCR Results Using HTML

For rapid prototyping of evaluation tools which are more flexible and less platform-dependent, a framework for integrating the JOCR system with the Internet/Intranet has been outlined (see Figure 8). Under the framework, the output of the JOCR system is converted to HTML files, which can be viewed with a Web browser such as Netscape Navigator 2.0/3.0. Simple Perl scripts can be written to convert the OCR results in a detailed format (in the format shown in Figure 7) into HTML files.

Compared with the approach of designing special GUI programs for performance evaluation, the advantages of using HTML and a Web browser are obvious: scripts for converting OCR results into HTML are much easier to write, test and maintain than X-Window-based GUI programs; HTML files for OCR results can be accessed on different platforms; and the Japanese language is automatically supported in Netscape Navigator 2.0/3.0.

For a Japanese document page, the recognition system can generate OCR results for each text block. In current experiments, we focused on the task of converting OCR results for a text block into HTML files. Aspects of system performance to be evaluated are: character segmentation, character recognition and character clustering. Given OCR results of a text block, a Perl script was written to generate six HTML files:

1. `Block_Image`: the HTML file for the input image; The TIFF image is converted to GIF format for use in HTML.

2. `Character_Images`: the HTML file for character segmentation; This file shows character segmentation results. A sequence of character images is displayed. Here, images are organized in their original order.

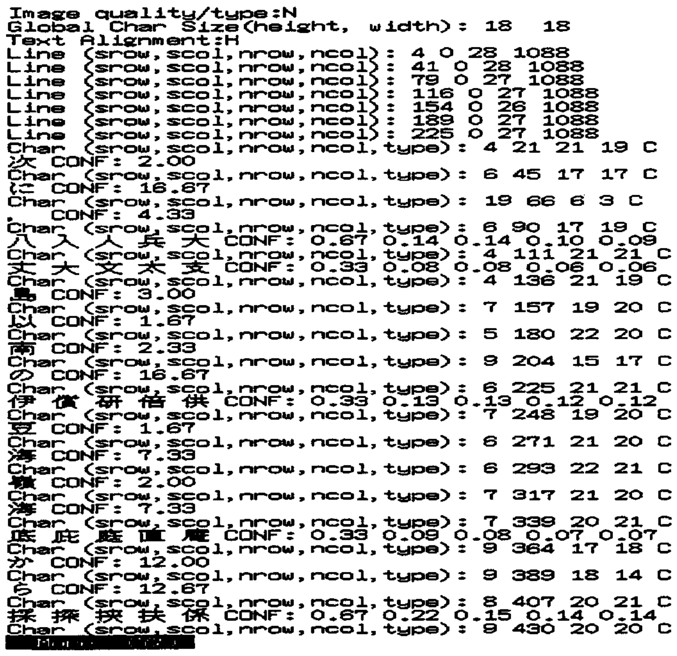
3. `OCR_Result`: the HTML file for recognition results; This file shows character recognition results. Corresponding to each character image, its top choice is displayed in JIS code.

4. `Symbol_or_Image`: the HTML file for the hybrid representation of recognition results; For each character position, either a JIS code or a GIF image is placed. For a character, a JIS code is given if the OCR confidence is high; otherwise, the original character image is placed there.

5. `Image_Clustering`: the HTML file for image-based clustering; Results of character image clustering are saved in the file. In a cluster, visually equivalent character images are grouped.

6. `OCR_Result_Clustering`: the HTML file for OCR-result-based clustering. Result of JIS-code-based clustering is saved in the file. Character images recognized as the same JIS code are grouped into clusters.

In each of those HTML files except the first one, for every character position (either a JIS code or an



```
Image quality/type:N
Global Char Size(height, width): 18  18
Text Alignment:H
Line (srow,scol,nrow,ncol): 4 0 28 1088
Line (srow,scol,nrow,ncol): 41 0 28 1088
Line (srow,scol,nrow,ncol): 79 0 27 1088
Line (srow,scol,nrow,ncol): 116 0 27 1088
Line (srow,scol,nrow,ncol): 154 0 26 1088
Line (srow,scol,nrow,ncol): 189 0 27 1088
Line (srow,scol,nrow,ncol): 225 0 27 1088
Char (srow,scol,nrow,ncol,type): 4 21 21 19 C
次 CONF: 2.00
Char (srow,scol,nrow,ncol,type): 6 45 17 17 C
に CONF: 16.67
Char (srow,scol,nrow,ncol,type): 19 66 6 3 C
.  CONF: 4.33
Char (srow,scol,nrow,ncol,type): 6 90 17 19 C
八 入 人 兵 大 CONF: 0.67 0.14 0.14 0.10 0.09
Char (srow,scol,nrow,ncol,type): 4 111 21 21 C
丈 大 文 太 支 CONF: 0.33 0.08 0.08 0.06 0.06
Char (srow,scol,nrow,ncol,type): 4 136 21 19 C
島 CONF: 3.00
Char (srow,scol,nrow,ncol,type): 7 157 19 20 C
以 CONF: 1.67
Char (srow,scol,nrow,ncol,type): 5 180 22 20 C
南 CONF: 2.33
Char (srow,scol,nrow,ncol,type): 9 204 15 17 C
の CONF: 16.67
Char (srow,scol,nrow,ncol,type): 6 225 21 21 C
伊 償 研 倍 供 CONF: 0.33 0.13 0.13 0.12 0.12
Char (srow,scol,nrow,ncol,type): 7 248 19 20 C
豆 CONF: 1.67
Char (srow,scol,nrow,ncol,type): 6 271 21 20 C
海 CONF: 7.33
Char (srow,scol,nrow,ncol,type): 6 293 22 21 C
鞭 CONF: 2.00
Char (srow,scol,nrow,ncol,type): 7 317 21 20 C
海 CONF: 7.33
Char (srow,scol,nrow,ncol,type): 7 339 20 21 C
底 庇 庭 直 雇 CONF: 0.33 0.09 0.08 0.07 0.07
Char (srow,scol,nrow,ncol,type): 9 364 17 18 C
か CONF: 12.00
Char (srow,scol,nrow,ncol,type): 9 389 18 14 C
ら CONF: 12.67
Char (srow,scol,nrow,ncol,type): 8 407 20 21 C
採 探 挟 扶 採 CONF: 0.67 0.22 0.15 0.14 0.14
Char (srow,scol,nrow,ncol,type): 9 430 20 20 C
```

Figure 7: The details of text recognition results.

image), a hyperlink to its OCR file is provided so that its OCR result can be examined by clicking a mouse on it.

The OCR file of a character is also in HTML. In the file, the character image is extracted from the block image and saved in GIF format; the classification result is also saved in a HTML file.

The Netscape browser provides a mechanism to have several frames simultaneously inside its window. Each frame can display an individual HTML file. Using the browser, a homepage for JOCR results was designed with three frames: the TOC frame, the OCR frame and the MAIN frame (see Figure 9). The top one is the TOC frame which has a table in which each entry is a hyperlink to one of six HTML files described above. The left bottom one is the OCR frame which is used to display the recognition result for a character. The right bottom one is the MAIN frame which is designed for displaying one of the six HTML files defined above. By clicking on a hyperlink in the TOC frame, the linked file will be displayed in the MAIN frame; by clicking on a hyperlink in the MAIN frame, the linked OCR file will be shown in the OCR frame [13]. [1]

Activated by selecting the entry Symbol_or_Image in the TOC frame, a hybrid HTML file for OCR results can be displayed in the MAIN frame (see Figure 9). Here, each character is either a JIS code or an image. For this text block, most characters are in JIS because their OCR confidence scores are high. For those characters with low OCR confidence, their original images are displayed. Obviously, those positions where images are placed usually have segmentation problems or recognition errors. Although there are many segmentation and recognition errors in the OCR results which make the JIS code text difficult to read, the content in the hybrid representation is quite readable and not much information is lost because of recognition errors. If the user wants to correct those errors, he can load the hybrid HTML file into an HTML editor (such as HotMetal from SoftQuad, Inc or the WYSIWYG editor which is embedded in the Netscape's Navigator Gold for Windows 95 and NT) and replace those images with their equivalent JIS code.

Experiments on visual contextual analysis have been conducted in our system [12]. In the simplest case, character image clustering is performed. After image clustering, images which are visually equivalent will be grouped as clusters. Image clustering results can be converted into HTML format, as shown in Figure 10 (a). By browsing this page, accuracy and efficiency of the clustering method can be evaluated. Because each image in a cluster is also a hyperlink to its OCR file, by checking the OCR result of every character inside a cluster, we can examine whether the recognizer runs consistently on visually

[1]The HTML pages presented in the paper can be found at `http://www.cedar.buffalo.edu/JOCR/HTML-demo/w95.html2.frm.html`. More demonstrations for various Japanese and Chinese document pages can be located at `http://www.cedar.buffalo.edu/JOCR/HTML-demo/index.html`.

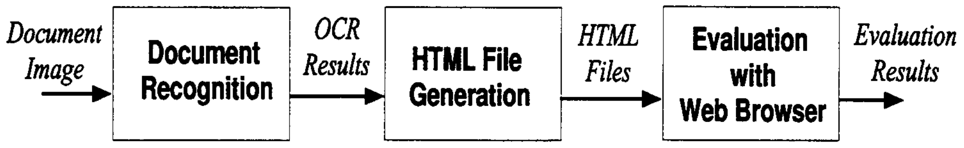


Figure 8: Integrating the JOCR system with the Internet/Intranet.

similar images.

Similarly, clustering based on OCR results is also very useful for fast error correction [1]. The MAIN frame of Figure 10 (b) shows the clustering result for the example block. All images which are recognized as the same character are grouped together.

It is easy to modify the Perl script when the appearance of an HTML page needs to be changed. In our experiments, performance of character segmentation, character recognition and image-based clustering were evaluated. New scripts for converting OCR results to HTML files can be designed rapidly when some other aspect of OCR performance needs to be studied.

OCR transforms a given document from its image representation into its symbolic representation. After this process, we obtain a text document which is electronically searchable, indexable and reusable. However, the transformation is error-prone. Therefore, if we just save the pure text document generated by the OCR process, the text may have many errors. Without referring to the original image representation from which they were derived, some of those errors may not be recoverable. As suggested by our experiment shown in Figure 9, a hybrid document which combines a symbolic representation and an image representation may relieve the problem. Such a hybrid document in HTML is searchable and also error-recoverable. If we represent an OCRed document in such a hybrid format, OCR errors will not have much negative effect on the human reading process. Therefore, the hybrid HTML can be recommended as an ideal format for OCRed documents in an online digital library [24].

Our Japanese OCR system is designed to recognize the first level of Kanji in the JIS code. When characters from the second level of the JIS code are encountered, OCR results for those characters will always be incorrect. It is best to keep those characters as original images so that the text will still be readable with a browser [15, 13].

Document images can be edited after segmentation. Usually this needs a specially designed GUI program [4]. An HTML file browser or editor can work for this purpose, as we have shown in our experiments. For a text block image, after character segmentation, a sequence of character images arranged in their original linear order can be rendered as readable as the original image, and different markup instructions can be applied. This kind of image-based document reconstruction can be very useful. We conducted another experiment to reconstruct documents more efficiently. For a Japanese document, usually there are many visually equivalent character images (as we have shown in our image-based clustering experiment). After image clustering, those visually equivalent character images can be viewed as the same image at different positions. Therefore, when the document image is reconstructed, we do not have to paste every image to its position. Instead, for those equivalent character images, a prototype can be generated to represent all of them at different positions [15, 13].

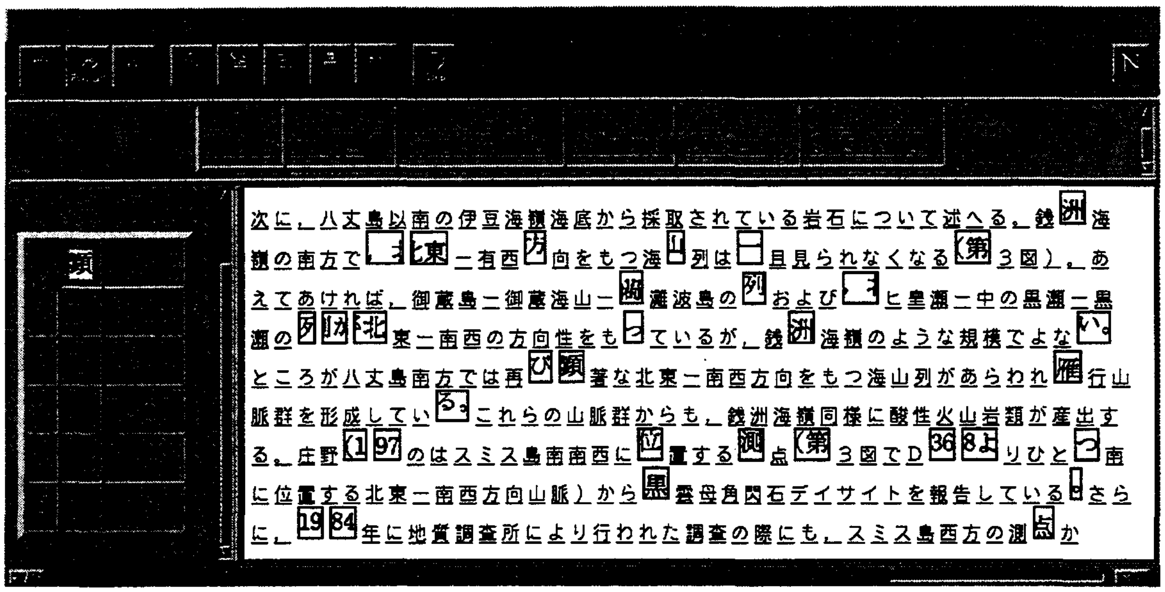
7 Conclusions and Future Directions

The Japanese OCR system, *Cherry Blossom*, has been designed to recognize Japanese documents digitized at low resolution such as 200 ppi. A wide variety of Japanese document images extracted from newspapers, journals, facsimiles and other sources have been used in the development and testing of this system. Results of character classifiers using CEDAR's Japanese character database have been presented. Experiments on Internet-based system evaluation is also reported. we focused on using HTML to evaluate performance at the block level. Experiments can be generalized to use HTML for physical/logical layout analysis. In this framework, we use hyperlinks and image-embedding functions provided by HTML to integrate components of the OCR results with components of the text image to overcome the difficulties of OCR performance evaluation and error correction. In the general paradigm of multivalent documents [31], we will extend the approach to tackle more sophisticated problems in visual document recognition.

More efforts will be directed towards further improving each module of the system and benchmarking of the performance of the system.

Acknowledgment

Present and past members of the *Cherry Blossom* team include: Geetha Srikantan, Yi L. Chiang, Chi Fang, Brian Grom, Zhou Hong, Jonathan J. Hull, Stephen W. Lam, Dar-Shyang Lee, Qunfeng Liao, Minsio Kariya, Seiichiro Kashimori, Chong O. Kim, Daniel R. Mechanic, Xiaomin Ren, Weh-Chih Wang,



次に，八丈島以南の伊豆海嶺海底から採取されている岩石について述べる。銭洲海
嶺の南方で北東ー有西方向をもつ海山列は旦見られなくなる（第3図）。あ
えてあげれば，御蔵島ー御蔵海山ー湖灘波島の列およびヒ皇瀬ー中の黒瀬ー黒
瀬の列が北東ー南西の方向性をもっているが，銭洲海嶺のような規模でよない。
ところが八丈島南方では再び顕著な北東ー南西方向をもつ海山列があらわれ雁行山
脈群を形成している。これらの山脈群からも，銭洲海嶺同様に酸性火山岩類が産出す
る。庄野(1 97のはスミス島南南西に位置する測点（第3図でD368よりひとつ南
に位置する北東ー南西方向山脈）から黒雲母角閃石デイサイトを報告している。さら
に，1984年に地質調査所により行われた調査の際にも，スミス島西方の測点か

Figure 9: Multi-frame design for presenting OCR results.

Shu-Fang Wu, Ying Yao, Xiaoming Yang, Victor C. Zandy and Jason Zhu.

References

[1] Illuminator user's manual, release 0.9. Technical report, RAF Technology, Inc., May 31 1995.

[2] T. Akiyama and N. Hagita. Automated entry system for printed documents. *Pattern Recognition*, 23:1141–1154, 1990.

[3] S. Ariyoshi. A character segmentation method for Japanese printed documents coping with touching character problems. In *Proceedings of the International Conference on Pattern Recognition*, pages 313–316, 1992.

[4] S.C. Bagley and G.E. Kopec. Editing images of text. *CACM*, 37(12):63–72, 1994.

[5] J.T. Favata, G. Srikantan, and S. N. Srihari. Handprinted character/digit recognition using a multiple feature/resolution philosophy. In *International Workshop on the Frontiers of Handwriting Recognition, IWFHR - 4*, 1994.

[6] K. Fukunaga. *Introduction to Statistical Pattern Recognition.* Academic Press, 1990.

[7] G.W. Gates. The reduced nearest neighbor rule. *IEEE Transactions on Information Theory*, IT-18(3):431–433, May 1972.

[8] P.E. Hart. The condensed nearest neighbor rule. *IEEE Transactions on Information Theory*, IT-14(3):515–516, May 1967.

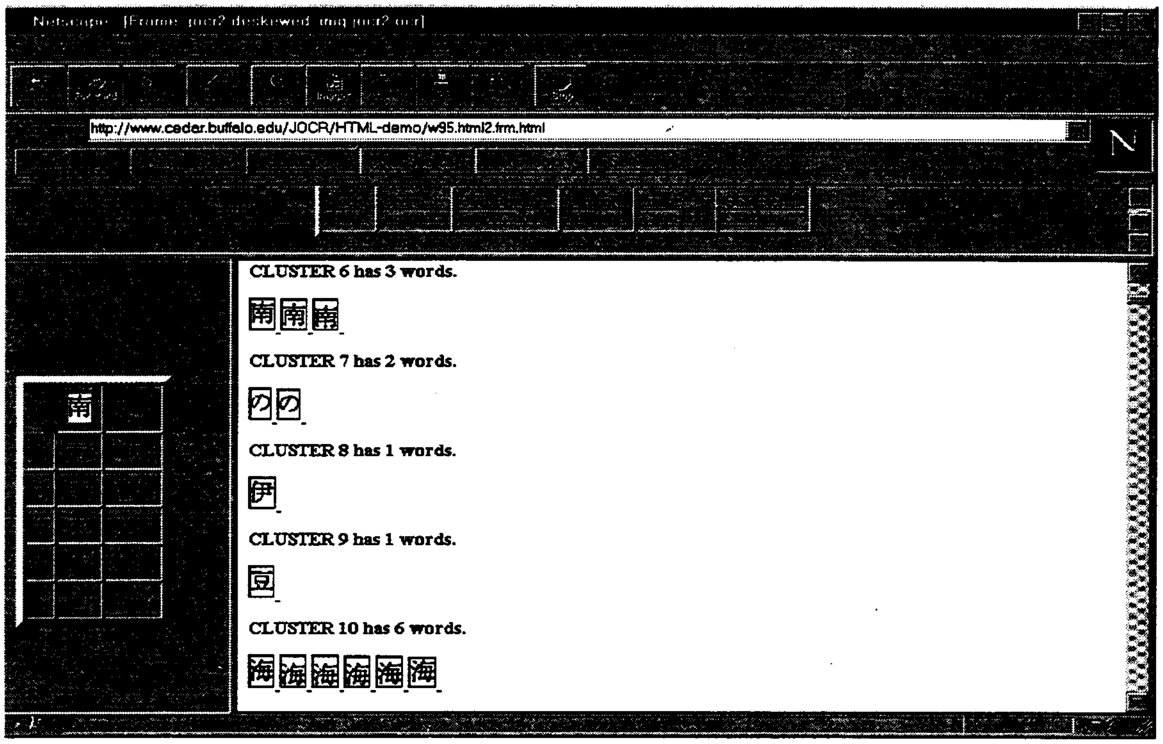
[9] T. Hong. *Degraded Text Recognition Using Visual and Linguistic Context.* PhD thesis, Computer Science Department of SUNY at Buffalo, 1995.

[10] T. Hong and J. J. Hull. Improving OCR performance with word image equivalence. In *Symposium on Document Analysis and Information Retrieval*, pages 177–190, April 24-26 1995.

[11] T. Hong, S.W. Lam, J.J. Hull, and S.N. Srihari. The design of a nearest-neighbor classifier and its use for Japanese character recognition. In *Proceedings of the Third International Conference on Document Analysis and Recognition ICDAR'95*, 1995.

[12] T. Hong, S.W. Lam, J.J. Hull, and S.N. Srihari. Visual similarity analysis of Chinese characters and its uses in Japanese ocr. In *Proceedings of the Conference on Document Recognition, 1995 SPIE Symposium(SPIE95)*, February 1995.

[13] T. Hong and S. N. Srihari. Representing ocred documents in html. In *Proceedings of International Conference on Computational Intelligence and Multimedia Applications*, pages 10–12, 1997.



http://www.cedar.buffalo.edu/JOCR/HTML-demo/w95.html2.frm.html

CLUSTER 6 has 3 words.

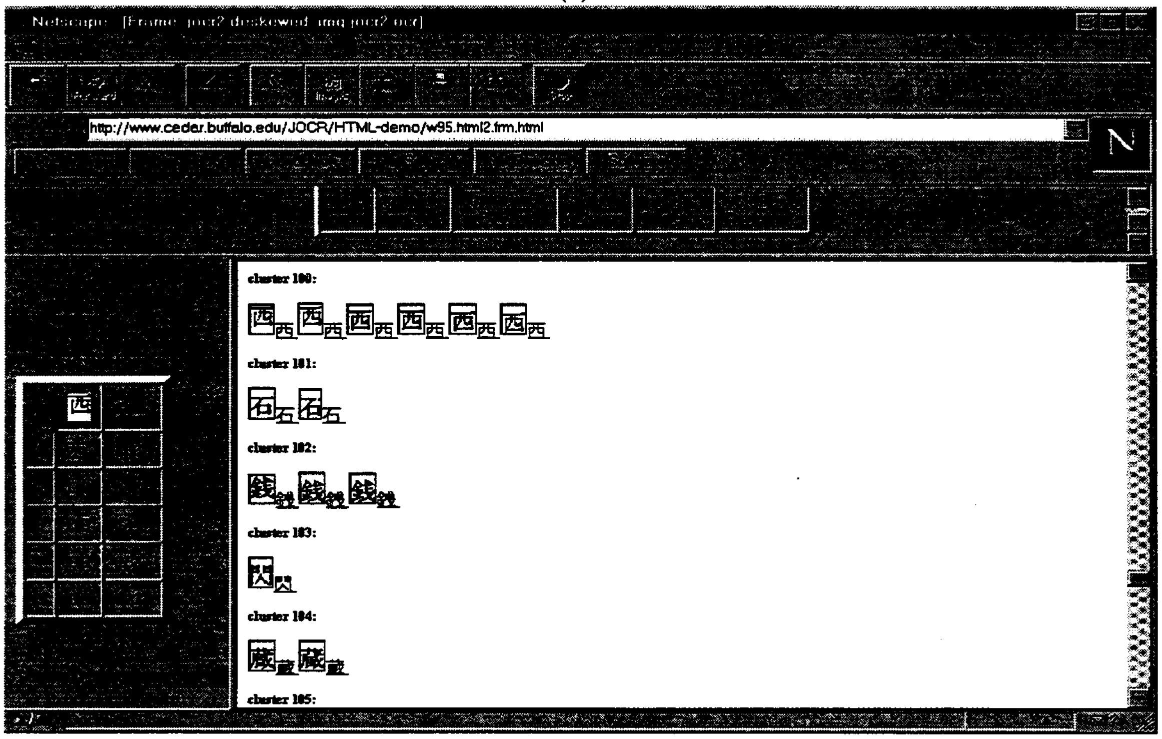
CLUSTER 7 has 2 words.

CLUSTER 8 has 1 words.

CLUSTER 9 has 1 words.

CLUSTER 10 has 6 words.

(a)



http://www.cedar.buffalo.edu/JOCR/HTML-demo/w95.html2.frm.html

cluster 100:

cluster 101:

cluster 102:

cluster 103:

cluster 104:

cluster 105:

(b)

Figure 10: (a) The result of image-based clustering is shown in the MAIN frame; (b) The result of JIS-code-based clustering is shown in the MAIN frame.

[14] T. Hong, G. Srikantan, V.C. Zandy, C. Fang, and S. N. Srihari. Character recognition in a Japanese text recognition system. In *Proceedings of the Conference on Document Recognition, 1996 SPIE Symposium(SPIE96)*, January 1996.

[15] T. Hong, S. Wu, and S.N. Srihari. Evaluating japanese document recognition in the internet/intranet environment. In *Proceedings of Workshop on Document Analysis Systems*, pages 14–16, 1996.

[16] http://www.cedar.buffalo.edu/Database/JOCR/. CEDAR's Japanese character image database.

[17] T. Ijima, H. Genchi, and K. Mori. A theory of character recognition by pattern matching method. In *Proc. First IJCPR*, pages 587–594, 1973.

[18] D.J. Ittner and H.S. Baird. Language-free layout analysis. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 336–340, 1993.

[19] F. Kimura, M. Shridhar, and Y. Miyake. Relationship among quadratic discriminant functions for pattern recognition. In *International Workshop on the Frontiers of Handwriting Recognition, IWFHR - 4*, 1994.

[20] F. Kimura, K. Takashina, S. Tsuruoka, and Y. Miyake. Modified quadratic discriminant functions and the application to Chinese character recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9:149–153, 1987.

[21] S. W. Lam. A local-to-global approach to complex document layout analysis. In *Proceedings of Machine Vision Applications*, 1994.

[22] S. W. Lam, Q. F. Liao, and S. N. Srihari. A divide-and-conquer approach to Japanese text segmentation. In *Proceedings of the Conference on Document Recognition, 1995 SPIE Symposium(SPIE95)*, February 1995.

[23] S. W. Lam and V. C. Zandy. Skew detection using directional profile analysis. In *Proceedings of Machine Vision Applications*, 1994.

[24] M. Lesk. Substituting images for books: The economics for libraries. In *Symposium on Document Analysis and Information Retrieval*, pages 1–16, April 15-17 1996.

[25] K. Lunde. *Understanding Japanese Information Processing.* O'Reilly and Associates, Inc., Sebastopol, CA, USA, 1993.

[26] Q. Luo, T. Watanabe, and N. Sugie. A structure recognition method for Japanese newspapers. In *SDAIR*, pages 217 - 234, 1992.

[27] Y. Maeda, F. Yoda, K. Matsuura, and H. Nambu. Character segmentation in Japanese hand-written document images. In *Proceedings of the International Conference on Pattern Recognition*, pages 769–772, 1986.

[28] S. Mori, K. Yamamoto, and M. Yasuda. Research on machine recognition of handprinted characters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6:386–405, 1984.

[29] E. Oja. *Subspace Methods in Pattern Recognition.* John Wiley & Sons, 1983.

[30] R. Oka. Handwritten Chinese character recognition by using cellular feature. In *Proc. 6th Int. Joint Conf. Pattern Recognition*, pages 783–785, 1982.

[31] T. Phelps and R. Wilensky. Multivalent documents: Inducing structure and behaviors in online digital documents. In *Proceedings of the 29th Hawaii International Conference on System Sciences*, January 3-6 1996.

[32] K. Sakai, S. Hirai, T. Kawada, S. Amano, and K. Mori. An optical Chinese character reader. In *Proc. 3rd Proceedings of the International Conference on Pattern Recognition*, pages 122–126, 1976.

[33] S. N. Srihari and Geetha Srikantan. Japanese optical character recognition. In *Proceedings of 1995 Symposium on Document Image Understanding Technology*, pages 236–246, 1995.

[34] G. Srikantan, S.W. Lam, and S. N. Srihari. Gradient-based contour encoding for character recognition. *Pattern Recognition Journal*, 29(7):1147–1160, July 1996.

[35] S. Tsujimoto and H. Asada. Major components of a complete text reading system. *Proceedings of the IEEE*, 80:1133–1149, 1992.

PENMAN: A System for Reading Unconstrained Handwritten Page Images

Sargur N. Srihari **Gyeonghwan Kim**
Center of Excellence for Document Analysis and Recognition
State University of New York at Buffalo
520 Lee Entrance
Amherst, NY 14228
{srihari,gkim}@cedar.buffalo.edu

Abstract

PENMAN is an end-to-end system for reading handwritten page images. Five functional modules are defined in the system: (i) pre-processing, which concerns introducing an image representation for easy manipulation of large page images and image handling procedures using the image representation, (ii) line separation, concerns extracting images of lines of text from a page image, (iii) word segmentation, concerns locating word gaps within a line of text obtained efficiently and in an intelligent manner, (iv) word recognition, concerns handwritten word recognition algorithms, and (v) linguistic post-processing, concerns incorporating natural language knowledge to filter confusion cases and improve recognition performance. The functional modules have been developed for dealing with the diversity of handwriting in its various aspects with a goal of system reliability and robustness. The system, written in C, has about 30,000 lines of code, and takes less than one minute on a SUN Sparc 20 to process an 8.5 in × 11 in page scanned at 300 dpi binary.

1 Introduction

There have been many efforts in the past in the recognition of handwriting. Most have focused on dealing with isolated units, such as characters and words [1–3]. Recognition of unconstrained handwritten page images has been a challenge because of the diversity of handwriting in many aspects, such as writing styles, inconsistent spacing between words and lines, and uncertainty of the number of lines in a page and the number of words in a line.

This paper is a status report of an end-to-end system, PENMAN, which is being developed at CEDAR for reading handwritten pages [4]. The system includes functional modules that have been developed in CEDAR for handwriting recognition researches, and newly developed algorithms to handle the page images effectively. The functional modules are: (i) image pre-processing, (ii) line separation, (iii) word segmentation, (iv) word recognition, and (v) linguistic post-processing. Each module is designed to deal with the real problem of handling handwritten page images effectively. Preliminary results are promising in both accuracy and speed aspects.

The paper is organized as follows: Section 2 describes overview of the PENMAN. Section 3 explains details of each functional module defined in the Section 2. Section 4 is on the preliminary experimental results and Section 5 provides summary of the paper and future work. A GUI version of the system is briefly introduced in the appendix.

2 System overview

Figure 1 shows the overall flow of the PENMAN system. Inputs to the system are a handwritten page image, a lexicon for the word recognizer, and a corpus for the linguistic post-processor. Outputs of the system are post-processing results of each line and word recognition result of each word segment along with the corresponding images.

After a page image is read, the binary image is converted into chain code representation for efficient manipulation without loss of shape information. *Line separation* module locates lines of text from the page image and splits lines when lines are touching. A clustering algorithm based on local maxima and minima points extracted from the components is applied for the purpose. *Image pre-processing* is performed on each line of text image separated, and includes slant correction, noise removal and smoothing contours. *Word segmentation* module interprets and locates gaps between words. Because range of the number of words in a line is wide, in contrast to other applications (such as street name interpretation in postal addresses), accuracy of the module is the crucial factor which determines the entire system performance in terms of both accuracy and speed. *Word recognition* inputs a word segment

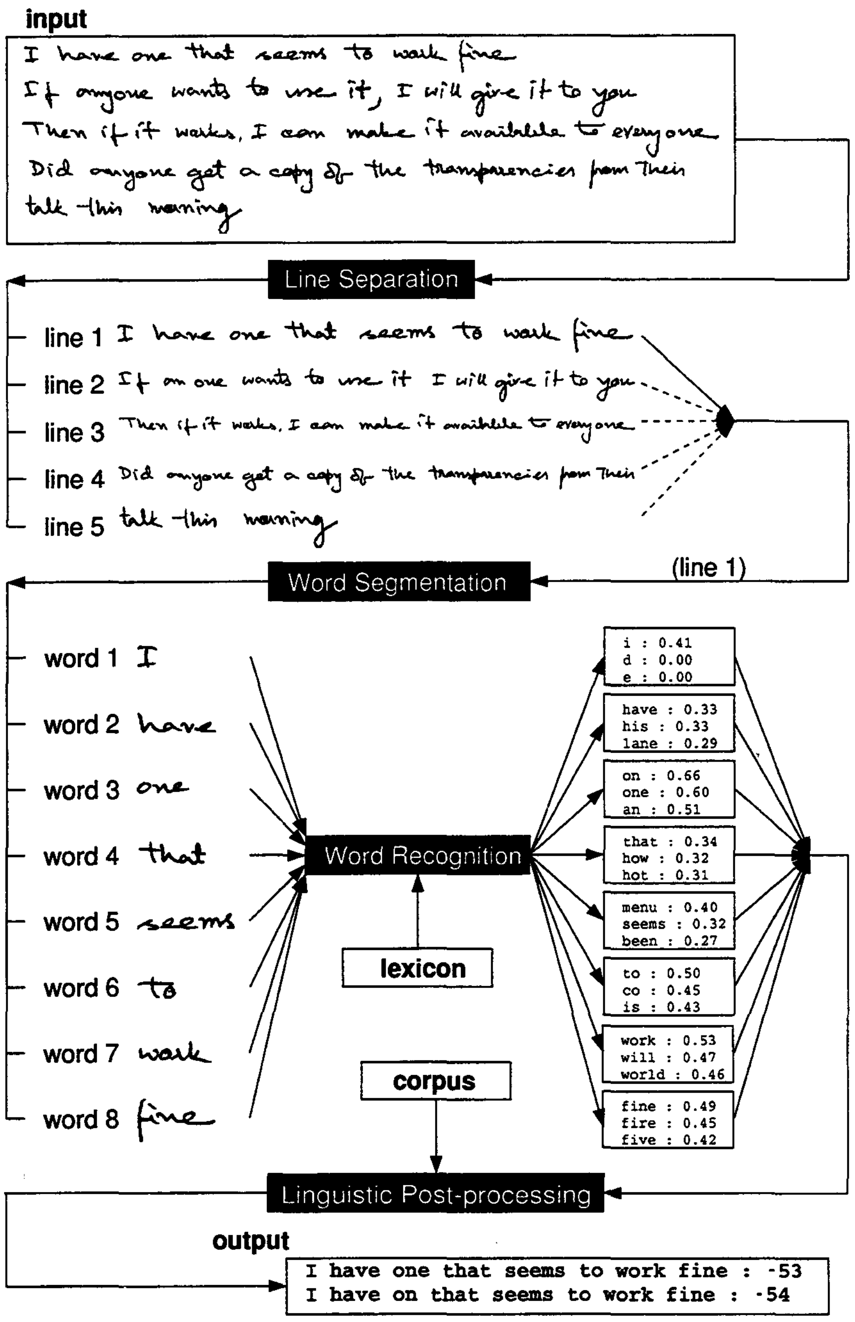


Figure 1: System flow of PENMAN

at a time and matches the image against entries in the lexicon. Top choices of the recognition result are saved along with their confidences. *Linguistic post-processing* module picks up most appropriate words from the top choices of the word recognition result by looking at the recognition confidence and linguistic information so that the recognition performance in a sentence level can be improved.

Following section describes more details of each functional module shown in Figure 1.

3 Functional modules

Techniques developed in CEDAR over several years have been applied to build the PENMAN. Most of the techniques were aimed at postal address interpretation [5]. Therefore, many of the techniques were needed to be modified to suit the requirement of the PENMAN. In addition, new algorithms were developed to integrate the modules effectively and make the system reliable and fast. This section describes the key ideas of each functional module.

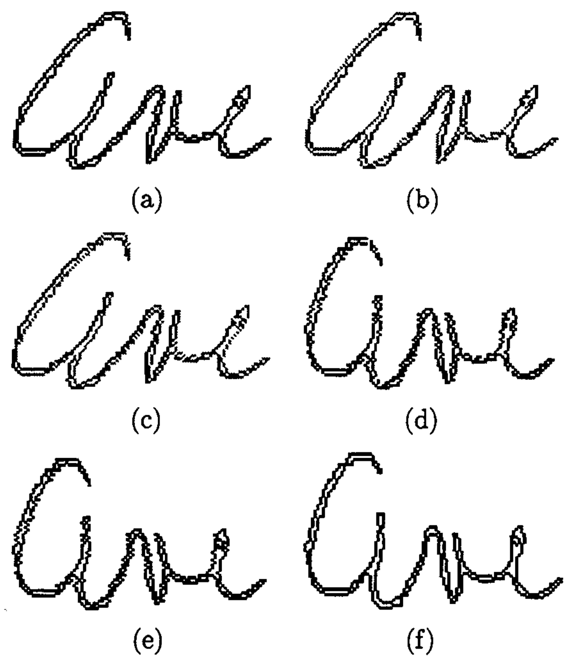


Figure 2: Pre-processing: (a) contour representation of an image, extraction of upward (b) and downward (c) vertical lines, (d) correction of x-coordinates based on the estimated angle, (e) connecting broken points. (f) smoothing contours

3.1 Pre-processing

Dealing with big size input images is a serious concern. For example, image size of a US letter size (8.5in × 11in) with 300 dpi(dots per inch) resolution is 8.4 Mbyte! Therefore, we need to introduce an image representation which capable of reducing the amount of data to be processed without loss of shape information. We have chosen the chain code scheme among the several image representation existing [6].

Once the input image is converted into chain code representation, all subsequent image handling procedures are needed to be developed using the image representation to take full advantage of the conversion. The following three functional modules in the system, including line separation, word segmentation and word recognition, work with this image representation.

Pre-processing steps applied to the system are noise removal, slant correction and smoothing chain code contours. Figure 2 shows the steps applied.

Details of the processing steps using the image representation are in [7].

3.2 Line separation

Lines of text must be first separated before any recognition can proceed. A line separation algorithm that works on address blocks has been modified for the purpose.

The algorithm uses the idea that the baseline of a handwritten line is well-defined. People write on an imaginary line on which the core of each word of the line resides. This imaginary baseline is approximated by the location of the local minima points from each component. A clustering technique is used to group the minima of all the components to identify the different handwritten lines.

1. Identify all meaningful components in the image, ignoring components that are likely to be noise or camera artifacts.

2. Find all extreme points (maxima and minima) contributed by each meaningful component. The average y-value difference between consecutive maxima and minima is taken to be the component height (Figure 3(b)).

3. Sort all minima by x-value.

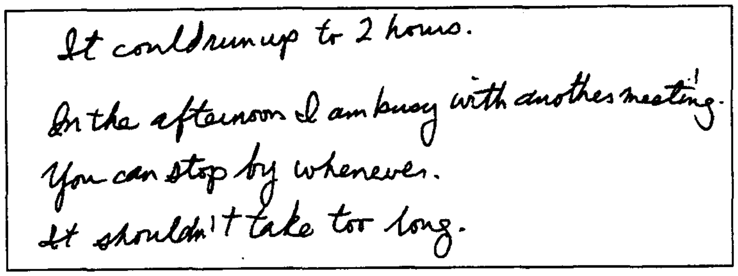
4. Using average component height as a useful measure of probable distance between neighboring clusters, begin clustering of the minima from the left side of the image. A new cluster is created when the next minimum is farther than the average component height from all currently existing clusters.

5. Repeat the clustering from the right side of the image.

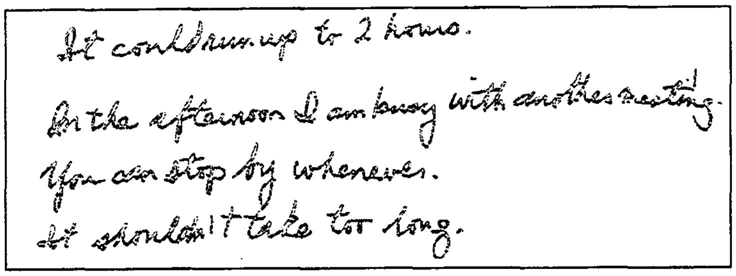
6. Combine information from both the left and right clusterings. If the left and right clusterings produce the same number of clusters, and membership of each cluster is the same in both the left and right clusterings, the clusters are final. If there is disagreement, a combination strategy is employed.

7. Post-processing is necessary to deal with individual components that contribute minima points to more than one cluster. Decisions must be made to assign components to line-groups based on their contributions of minima points. In some cases, a single component must be forcibly split into two or more components, each of which is then assigned to the appropriate line-group. This can happen, for example, when a descender from a character in one line touches a character which belongs to the line below (Figure 3(c)).

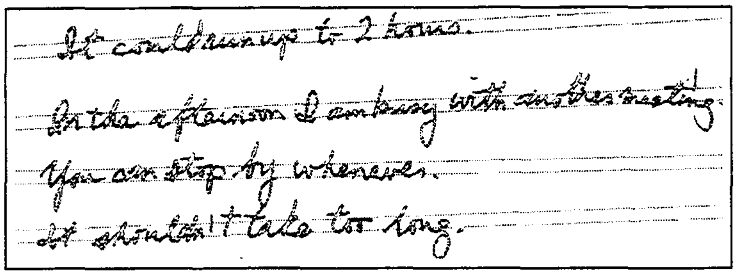
Figure 3 shows the different steps in line separation.


It could run up to 2 hours.
In the afternoon I am busy with another meeting.
You can stop by whenever.
It shouldn't take too long.

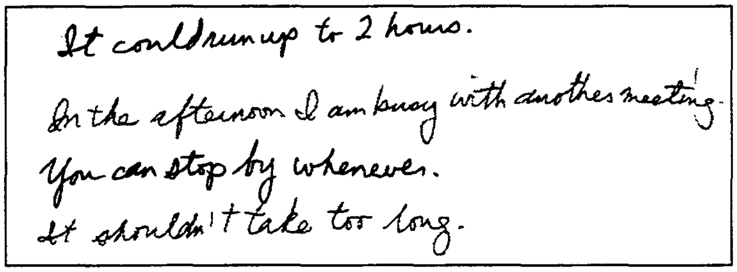
(a)


It could run up to 2 hours.
In the afternoon I am busy with another meeting.
You can stop by whenever.
It shouldn't take too long.

(b)


It could run up to 2 hours.
In the afternoon I am busy with another meeting.
You can stop by whenever.
It shouldn't take too long.

(c)


It could run up to 2 hours.
In the afternoon I am busy with another meeting.
You can stop by whenever.
It shouldn't take too long.

(d)

Figure 3: Line separation procedure - (a) a part of handwritten page image, (b) locating extreme points (dots) on contours, (c) line-groups after clustering, and (d) line segmentation result with different intensity

3.3 Word segmentation

Most recognizers are developed under the assumption that the separation is not an issue and focus on the recognition of isolated units (a character or a word). However, correct segmentation of the units without recognition is crucial in the implementation of the page reading system. Without feedback from recognizer, it is almost impossible to locate correct word gaps in the fields of handwriting interpretation. Therefore, the word segmentation performance is directly related to the performance of the system in terms of both speed and accuracy.

Few papers have dealt with the word segmentation issues and most of them have focused on identifying gaps using only geometric distances between connected components [8, 9]. It is assumed that gaps between words are bigger than the gaps between characters. However, in handwriting, excep-

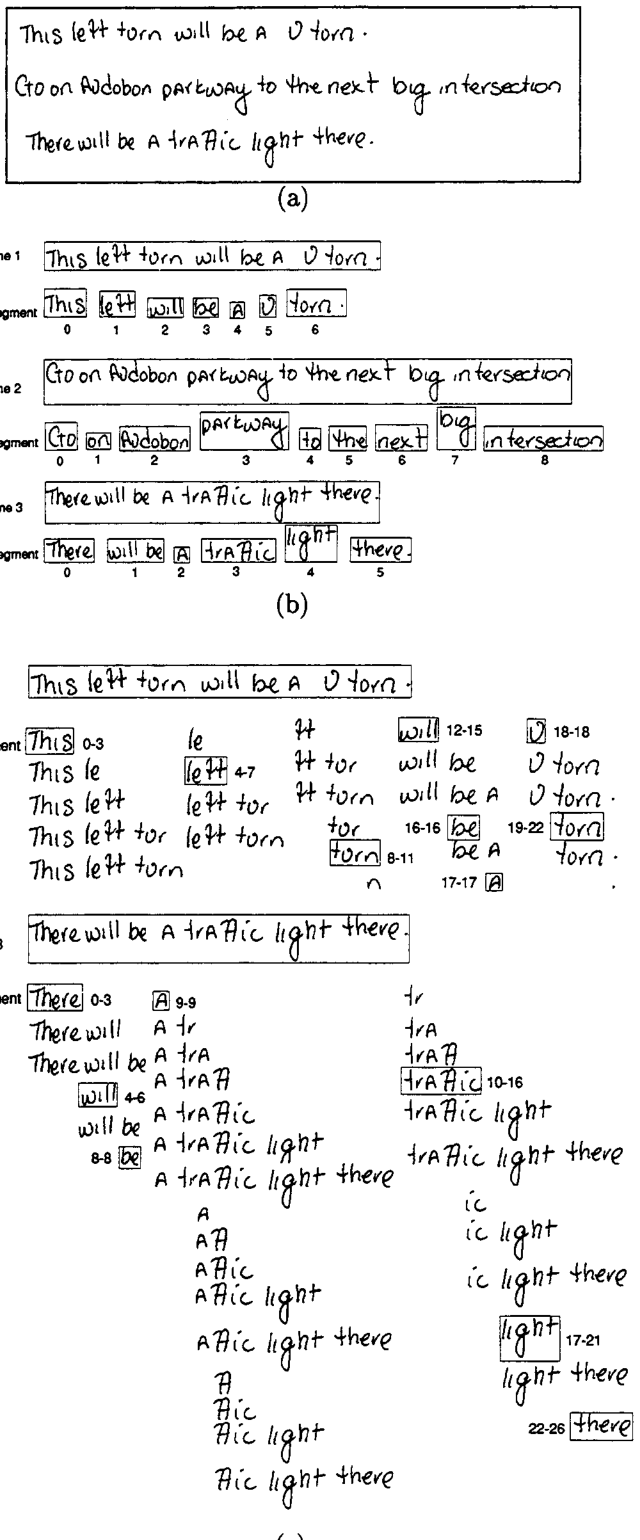

This left turn will be A U turn.
Go on Audobon partway to the next big intersection
There will be A traffic light there.
(a)
Line 1
Segment
Line 2
Line 3
(b)
(c)

Figure 4: (a) A sample text image with 3 sentences. (b) Word segmentation results of the new algorithm - most probable word gaps are located using a neural network without any word hypothesis (segment 1 in line 3 is under-segmented). (c) A conventional method - based on gaps between components and combining them together to avoid missing true word gaps (numbers next to the boxes represents the beginning and the ending connected component numbers).

tions are common place.

We have developed an intelligent word segmentation algorithm which incorporates cues that humans use and does not rely on using only the simple one-dimensional distance information. Gaps between character segments (a character segment can be a character or a part of character) and heights of character segments are used in the algorithm. In addition, to locate a word break, we use two consecutive intervals and more than two consecutive heights. A neural network is trained using the information extracted from training images (Figure 5).

More details for the algorithm can be found in [10].

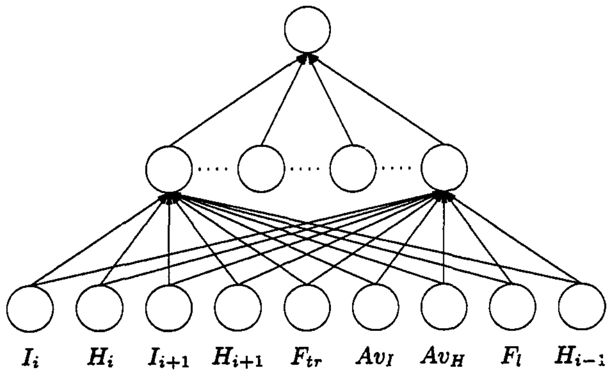


Figure 5: Structure of the neural network used for the word segmentation: inputs to the network are extracted from character segments

Twenty page images were collected and used for training the neural network. Each line image separated was fed to the training module and true word gaps were located manually. 18,211 training patterns were collected from the training images.

A sample image extracted from a page image is shown in Figure 4(a). In general, the word segmentation problem is more difficult when the script has discrete writing style rather than cursive one. Locating correct word gaps from line images in the sample text image is a real challenge (without any help from recognition). As shown in Figure 4(b), our new algorithm locates the most probable word gaps using the trained neural network without any word hypothesis. Figure 4(c) illustrates a conventional approach to deal with the problem. All possible word hypotheses are generated to avoid missing true word breaks and sent to the recognizer. As we can see in the example, when the gaps between words and characters have no significant difference, the problem is non-trivial in the conventional approach. Words in boxes are the words we want to look for.

Significant decrease in processing time has been observed as fewer word segments need to be recognized consequently fewer sentences need to be passed to the linguistic post-processor. (Timing profile shown in Section 4 indicate that the word recognition and the linguistic post-processing modules consume more than 70 percent of the entire processing time.)

3.4 Word recognition

Implementation of a fast handwritten word recognizer has been a challenging task in many real applications. The common design issues in developing a fast recognition system are: (i) implementation of efficient image processing routines and (ii) developing an appropriate matching algorithm.

The word model recognizer (Figure 6) is the one designed to meet the requirement. We used a chain code representation for efficient image handling. All image handling procedures such as normalization, segmentation, and feature extraction are performed on this representation. The reduction in data allows for a significant speed improvement. On the other side, the matching is efficiently designed to handle the large dimensional feature vectors which represent shape description of characters in a word. The recognizer employs a lexicon driven approach. The lexicon is introduced in early stages of the recognition process - an input word image is compared with *only* those words present in the lexicon.

Also, the concept of variable duration, which is obtained from character segmentation statistics and used for determining the size of matching window during the recognition is used in the recognizer. The variable duration maximizes the efficiency of the lexicon driven recognition scheme in terms of both speed as well as recognition accuracy.

Details on the recognizer are described in [11].

3.5 Linguistic post-processing

Post-processing models are unavoidable for applications such as phrase recognition and sentence recognition. Handwriting recognizers output a list of word choices with their scores for each input word position. The use of natural language knowledge is to filter confusion cases from the recognition results and improve overall recognition performance. Figure 7 shows a typical case the post-processing can improve sentence level recognition result.

The Human Language Technology group at CEDAR is involved in developing algorithms to solve the "graph search problem". A variation of the models is incorporated into the PENMAN. More details on the models being developed in CEDAR can be found in papers published by the group [12, 13].

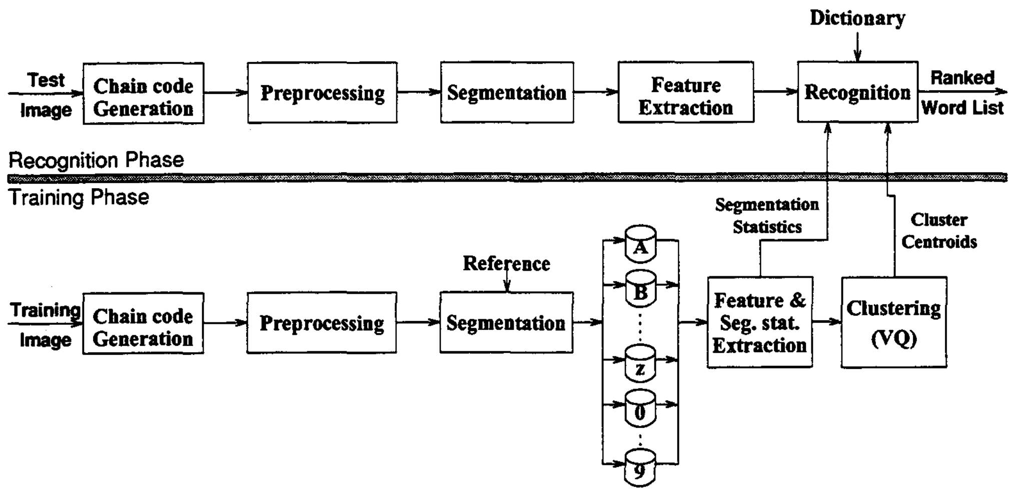
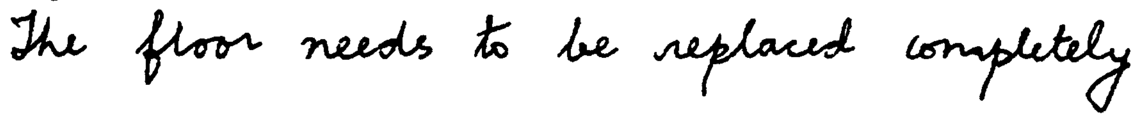


Figure 6: Overall structure of the word model recognizer(WMR)

image



word recognition results

the:0.39	from:0.39	need:0.62	to:0.31	in:0.52	**replaced:0.52**	**completely:0.47**
she:0.39	**floor:0.38**	**needs:0.56**	tv:0.25	is:0.45	ruined:0.40	butterfly:0.34
like:0.36	for:0.38	field:0.52	is:0.23	on:0.39	walked:0.39	military:0.34
we:0.31	fun:0.35	meet:0.48	as:0.18	**be:0.38**	placed:0.39	immediately:0.33
line:0.31	fear:0.35	area:0.48	ps:0.16	led:0.33	married:0.37	directly:0.33

post-processing results

1. **the floor needs to be replaced completely : -54.328**
2. the fun needs to be replaced completely : -54.367
3. the floor area to be replaced completely : -54.379
4. the floor needs to be replaced directly : -54.484
5. the fun needs to be replaced directly : -54.522

Figure 7: An example of post-processing that improves sentence level recognition: bold-faced words in the word recognition results show that truth of each word is not always top choice, but the post-processor chose the correct one as the top choice.

4 Experiments and results

A set of preliminary test result was collected to evaluate the performance of the system and measure the speed. Performance of the post-processing module was not evaluated because the module needs further development.

4.1 Image database

Twenty page images written by different authors were used for the evaluation (the images are different from ones used for training the neural network in Section 3.3). The database consists of scanned handwritten pages obtained in the process of collecting an on-line database [14]. The images are scanned at 300 dpi.

4.2 Timing analysis

Because each page image contains different number of lines and words, we chose a typical page image with discrete writing style as shown in Figure 8. Table 1 shows the timing profile of the system for the page image. It should be note that the time taken for loading data (such as lexicon and corpus) can be subtracted from the total processing time because the data files are common to all images and do not need to be loaded every time.

Module	time (sec)	percent
Reading image	1.65	3.0
Loading data(lex, corpus)	8.38	15.3
Chain code generation	1.98	3.6
Pre-processing	0.36	0.7
Line separation	0.40	0.7
Word segmentation	0.85	1.6
Word recognition	23.69	43.3
Ling. Post-processing	17.42	31.8
Total	54.73	100.0

Table 1: Timing profile of the system for a typical page image shown in Figure 8 with 300 dpi resolution (measured on a Sparc 20 platform).

Size of the lexicon used for the word recognizer is about 1,600 and the lexicon contains all words appear in the image set. Top 5 choices of each word recognition result were passed to the post-processing module. If we increase the number of choices, the processing time of the module also increases exponentially.

The efficiency of the image handling routines we have developed can be observed from the timing profile. Only less than 10 percent of the processing time was consumed by the image handling modules (chain code generation, pre-processing, line separation and word segmentation).

4.3 Performance analysis

Table 2 shows performance of the line separation, the word segmentation, and the word recognition for each image. Performance of the line separation and the word segmentation modules are very promising. It must be noted that the word segmentation scheme we applied for the evaluation has no word segmentation hypothesis for taking care of any possible over/under-segmentations.

5 Summary and future work

A complete end-to-end system for reading handwritten page image has been implemented. Each functional module integrated into the system has been designed to suit the requirement of the PENMAN. The preliminary evaluation results are promising. The entire processing time for a typical page image scanned with 300 dpi is less than a minute on a single Sparc 20 platform.

For further improvement of the system, the following issues are being considered:

- *image handling scheme*: Chain code, the image representation used in the PENMAN system, provides us many advantages in terms of implementing a faster system due to its reduced data size without loss of shape information. However, because of its one dimensional structure, implementation of some of the image handling algorithms is not simple. An alternative image representation, which can minimize the problem but still provides big data reduction, is to be considered.

- *line separation*: The version of line separation algorithm integrated into the system has originally developed for separating lines from address blocks in envelops which are smaller than ordinary page images. Therefore, some problems have been observed when longer and skewed lines are processed. A natural learning algorithm based on Hough transform has been being considered as a solution to separating lines from general text images. Collecting additional information that can be used for approximation of imaginary baseline, in addition to the local minima points from each component, is being considered. The local minima points often mislead the clustering process when long descenders and touched lines are present.

- *word segmentation*: Because the preliminary evaluation result of the neural network based word segmentation is very promising, several

image number	no. of lines	no. of lines separated	no. of words	no. of words segmented	no. of words recognized		writing style
					top 1	top 2-10	
0001	26	26 (100.00)	167	166 (99.40)	125 (75.30)	158 (95.18)	discrete
0007	28	26 (92.86)	170	163 (95.88)	124 (76.07)	152 (93.25)	discrete
0028	20	20 (100.00)	202	172 (85.15)	51 (29.65)	124 (72.09)	cursive
0036	27	27 (100.00)	150	137 (91.33)	70 (51.09)	114 (83.21)	mixed
0051	25	21 (84.00)	145	137 (94.48)	88 (64.23)	115 (83.94)	mixed
0055	22	20 (90.91)	172	172 (100.00)	104 (60.47)	137 (79.65)	mixed
0063	27	27 (100.00)	196	191 (97.45)	113 (59.16)	152 (79.58)	mixed
0072	17	15 (88.24)	93	91 (97.85)	28 (30.77)	59 (64.84)	mixed
0077	28	24 (85.71)	165	150 (90.91)	77 (51.33)	115 (76.67)	cursive
0093	27	23 (85.19)	205	200 (97.56)	107 (53.50)	150 (75.00)	mixed
0104	22	20 (90.91)	97	79 (81.44)	48 (60.76)	73 (92.41)	discrete
0118	22	22 (100.00)	75	69 (92.00)	47 (68.12)	52 (75.36)	discrete
0123	16	16 (100.00)	88	84 (95.45)	36 (42.86)	58 (69.05)	cursive
0126	20	20 (100.00)	168	166 (98.81)	110 (66.27)	147 (88.55)	mixed
0128	17	17 (100.00)	117	106 (90.60)	78 (73.58)	93 (87.74)	discrete
0132	27	27 (100.00)	165	165 (100.00)	67 (40.61)	110 (66.67)	cursive
0136	3	3 (100.00)	15	14 (93.33)	10 (71.43)	13 (92.86)	discrete
0137	27	27 (100.00)	147	146 (99.32)	79 (54.11)	112 (76.71)	mixed
0144	18	18 (100.00)	151	139 (92.05)	51 (36.69)	95 (68.35)	mixed
0166	10	10 (100.00)	72	60 (83.33)	37 (61.67)	52 (86.67)	cursive
Total	429	409 (95.34)	2760	2607 (94.46)	1450 (55.62)	2081 (79.82)	

Table 2: Performance of the system - line separation, word segmentation and word recognition

approaches to enhance the performance and to resolve some problems in the current scheme can be considered. When a line contains a single word and its writing style is discrete, the algorithm tries to segment the word based on the information collected from the line. Incorporation of global information from nearby line images for such short line images should be considered as a solution. Behavior analysis of the neural network can give clues on which segment needs to be further segmented and which segment needs to be combined. In addition to the inputs selected for the neural network, exploration of selecting more effective features is needed.

- *word recognition*: As the word segmentation and the line separation schemes are being developed, the importance of reliable character segmentation scheme is increased. Modification of the current character segmentation algorithm so that it can be shared by the word segmentation and possibly the line separation algorithms is being considered. Benefit of sharing the segmentation scheme will bring improvement in processing speed and provide easy expansion of the system when necessary. Punctuation problem has been ignored in the single word recognition domain, but is unavoidable in sentence recognition. Ways to handle punctuation marks should be researched.

- *linguistic post-processing*: It has been noted that the post-processor that is being developed is failing because the probabilities of score given word are based on a simulator program that is not matched well to the actual behavior of the word recognizer. In order to improve the post-processing performance we need extensive effort in many areas. In order to train the post-processor to integrate scores correctly, recognizer data for many word images is needed.

Acknowledgments

The authors would like to thank Dr. John Favata and Dr. Venu Govindaraju for their contribution to the project in its early stage, and Yao Pu for her participation in enhancing the line separation algorithm and developing the GUI version.

References

[1] R. M. Bozinovic and S. N. Srihari, "Off-line Cursive Script Word Recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 11, pp. 68–83, January 1989.

[2] J. C. Simon, "Off-line Cursive Word Recognition," *Proceedings of the IEEE*, vol. 80, no. 7, pp. 1150–1161, 1992.

[3] J. T. Favata, G. Srikantan, and S. N. Srihari, "Handprinted Character/Digit Recognition using a Multiple Feature/Resolution Philosophy,"

in *The fourth International Workshop on Frontiers in Handwriting Recognition*, Taipei, Taiwan, pp. 57–66, 1994.

[4] S. N. Srihari, V. Govindaraju, and J. Favata, "Unconstrained Handwritten Text Recognition," in *Symposium on Document Image Understanding Technology*, Oct. 24-25, Maryland, pp. 226–235, 1995.

[5] V. Govindaraju, A. Shekhawat, and S. N. Srihari, "Interpretation of Handwritten Addresses in US Mail Stream," in *The third International Workshop on Frontiers in Handwriting Recognition*, Buffalo, New York, pp. 197–206, 1993.

[6] H. Freeman, "Computer Processing of Line-Drawing Images," *Computing Surveys*, vol. 6, no. 1, pp. 57–97, 1974.

[7] G. Kim and V. Govindaraju, "Efficient Chain Code Based Image Manipulation for Handwritten Word Recognition," in *Proc. of the SPIE symposium on electronic imaging science and technology (Document Recognition III)*, San Jose, CA, vol. 2660, pp. 262–272, February 1996.

[8] G. Seni and E. Cohen, "External Word Segmentation of Off-Line Handwritten Text Lines," *Pattern Recognition*, vol. 27, no. 1, pp. 41–52, 1994.

[9] U. Mahadevan and R. C. Nagabushnam, "Gap Metrics for Word Separation in Handwritten Lines," in *Third International Conference on Document Analysis and Recognition (ICDAR-95)*, Montreal, Canada, pp. 124–127, 1995.

[10] G. Kim and V. Govindaraju, "Recognition of Handwritten Phrases as Applied to Street Name Images," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR96)*, San Francisco, CA, pp. 459–464, June 1996.

[11] G. Kim and V. Govindaraju, "Handwritten Word Recognition for Real-Time Applications," in *Third International Conference on Document Analysis and Recognition (ICDAR-95)*, Montreal, Canada, pp. 24–27, August 1995.

[12] D. Bouchaffra, E. Koontz, V. Kripasundar, and R. Srihari, "Incorporating Diverse Information Sources in Handwriting Recognition Postprocessing," *International Journal of Imaging Systems and Technology (IJIST)*, vol. 7, 1996.

[13] D. Bouchaffra, E. Koontz, V. Kripasundar, R. Srihari, and S. Srihari, "Integrated Signal and Language Context to Improve Handwritten Phrase Recognition: Alternative Approaches," in *Proceedings of the Sixth International Workshop on Artificial Intelligence and Statistics*, Ft. Lauderdale, Florida, January 1997.

[14] R. K. Srihari, "Database of On-line Handwriting," in *http://www.cedar.buffalo.edu/ Linguistics/database.html.*

Appendix: Graphic User Interface

For evaluation and debugging purpose, we have developed a GUI version of the system. We used Tcl/Tk because of its easy interface to the system. Each functional module has been modified to produce intermediate files. As the program proceeds by clicking buttons on the screen, the files are displayed by the tool.

A message line is provided to guide the operator at the bottom of each window.

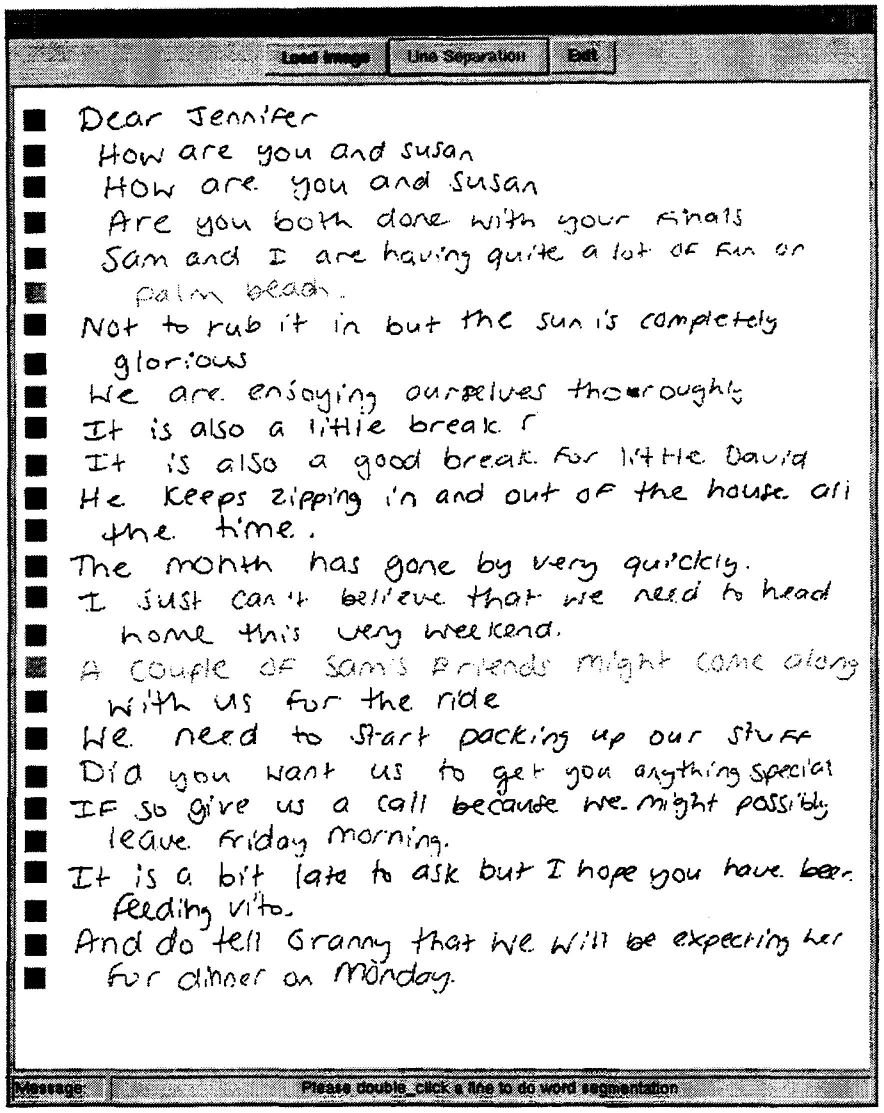
Figure 8 shows a line separation result of a typical page image. Different colors are assigned to the separated lines.

A line can be selected by clicking the line or the box in the beginning of the line with the same color. Word segmentation result of the selected line is displayed with (red) bounding boxes (figure 9).

After the word segmentation, word recognition is performed for all word segments identified. By clicking a box, word recognition result of the segment is shown as Figure 10.

By clicking the post-processing button (after the word recognition), most probable sentences determined by the module is displayed on the screen with the corresponding scores (Figure 11).

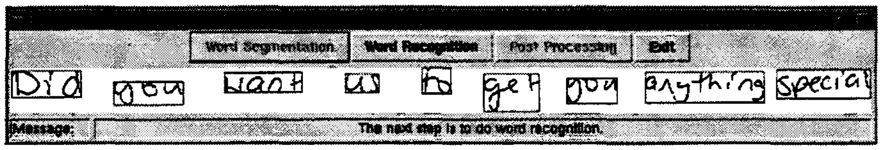
Any other line can be chosen at any point of the above procedures.



Dear Jennifer
How are you and susan
How are you and Susan
Are you both done with your finals
Sam and I are having quite a lot of fun on
palm beach.
Not to rub it in but the sun is completely
glorious
We are enjoying ourselves thoroughly
It is also a little break. r
It is also a good break for little David
He keeps zipping in and out of the house all
the time.
The month has gone by very quickly.
I just can't believe that we need to head
home this very weekend.
A couple of Sam's friends might come along
with us for the ride
We need to start packing up our stuff
Did you want us to get you anything special
If so give us a call because we might possibly
leave friday morning.
It is a bit late to ask but I hope you have been
feeding vito.
And do tell Granny that we will be expecting her
for dinner on Monday.

Message: Please double_click a line to do word segmentation

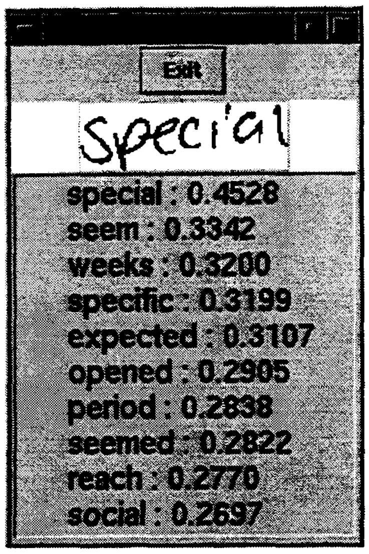
Figure 8: A typical page image after the line separation: small square in the beginning of each line represents the line and is used for selecting the line for the subsequent operation.



Did you want us to get you anything special

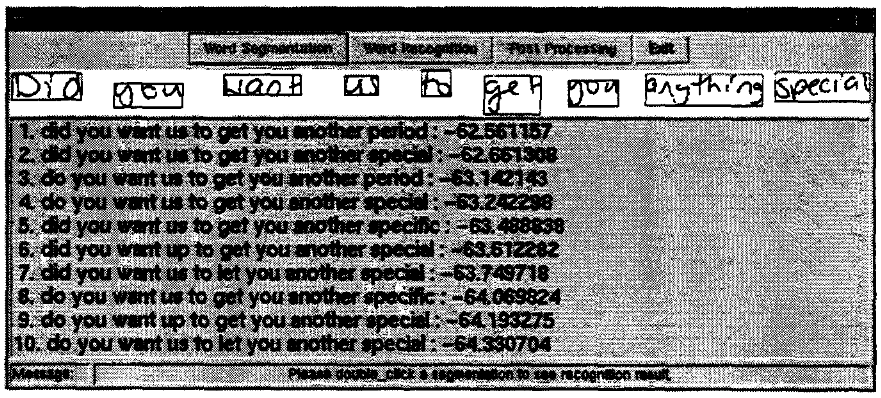
Message: The next step is to do word recognition

Figure 9: A line was selected from the image shown in Figure 8 and the word segmentation algorithm was applied. Boxes represent the word segmentation result.



```
special : 0.4528
seem : 0.3342
weeks : 0.3200
specific : 0.3199
expected : 0.3107
opened : 0.2905
period : 0.2838
seemed : 0.2822
reach : 0.2770
social : 0.2697
```

Figure 10: After word recognition, recognition result of a word(segment) is shown by clicking a box in Figure 9.



```
1. did you want us to get you another period : -62.561157
2. did you want us to get you another special : -62.661308
3. do you want us to get you another period : -63.142143
4. do you want us to get you another special : -63.242298
5. did you want us to get you another specific : -63.488838
6. did you want up to get you another special : -63.612282
7. did you want us to let you another special : -63.749718
8. do you want us to get you another specific : -64.069824
9. do you want up to get you another special : -64.193275
10. do you want us to let you another special : -64.330704
Message: Please double_click a segmentation to see recognition result.
```

Figure 11: Post-processing result of the line shown in Figure 9

An Extension of Illuminator for Chinese and Korean OCR Applications

Robert Thibadeau
School of Computer Science
Carnegie Mellon University
rht@ri.cmu.edu

Abstract

We will report on our work in Chinese and Korean OCR by showing how we have extended the RAF Illuminator software to include the Chinese and Korean pattern recognition systems. We will discuss hard problems of Korean OCR.

Developing an OCR system requires interactive tools as well as an environment in which the various system components can be tested and refined as components of a whole system. We have experimented with utilizing Illuminator as the base for such a system configuration. The goal is to achieve a versatile platform which facilitates the development of ground truth and training data sets, facilitates assessment of multiple component configurations such as the use of staged classifiers, and, finally, permits use of the components in an end-to-end, scan-to-finished text , workstation environment that supports newspaper scanning as well as conventionally sized paper this. This report will provide a description of our implementation, http://www.ul.cs.cmu.edu/software/illuminator, and appraisal.

Finally we will show and discuss a web based system under development in a separate project in coordination with scanning and conversion efforts underway with the National Academy Press that we term the CMU book object (http: //www.ul.cs.cmu.edu/search).

An Off-Line Arabic Recognition System for Machine-Printed Documents

J. Trenkle **A. Gillies**
S. Schlosser
Environmental Research
Institute of Michigan
P.O. Box 134001
Ann Arbor, MI 48113-4001

Abstract

This paper will discuss all aspects of a high-performance recognition system for machine-printed Arabic text. The system accepts any page image and first verifies that the page contains only machine-printed Arabic text — any other character sets or handwritten material is rejected. Other qualities of the image such as aspect ratio/resolution, quality, and orientation/skew are determined and corrections are made as necessary. All structures such as figures, pictures, charts, and lines are removed from the resampled, re-oriented and despeckled image. The remaining text is then decomposed into regions which are further broken into lines of text. The lines are normalized to a fixed height, segmented into words and these words are recognized using lexicon-based approaches. The individual algorithms developed for processing Arabic pages and recognizing Arabic text in this scheme will be discussed.

1 Background

The task of recognizing Arabic text imagery poses difficulties that are not encountered when attempting to recognize text written using the Latin alphabet. Machine-printed Arabic differs from European languages in that it is exclusively a script language; as with cursive handwriting, there is typically no separation between the characters in a word. The absence of character separation in Arabic text makes it very difficult to use recognition approaches based on character-level segmentation.

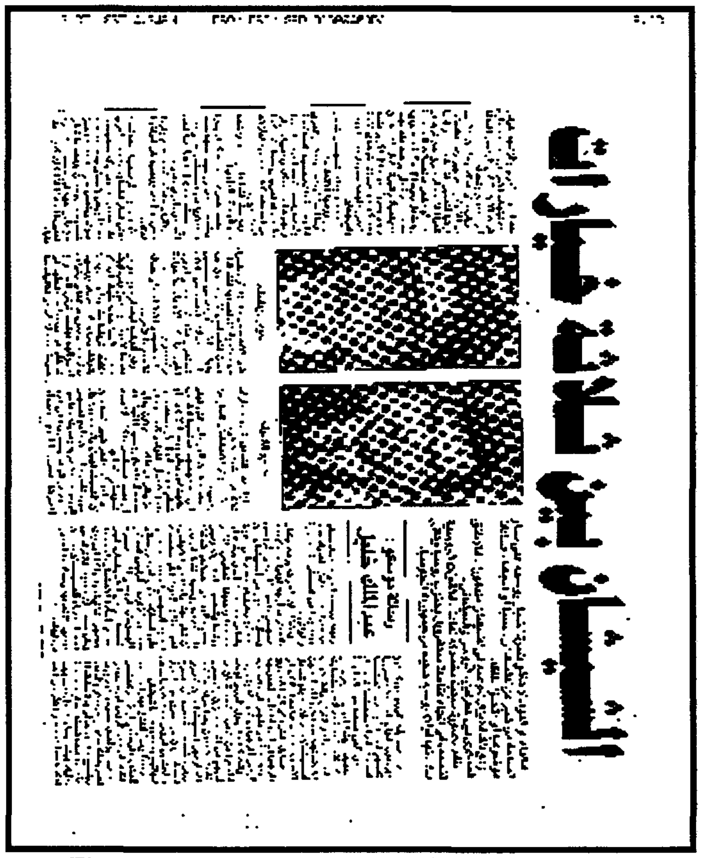


Figure 1. A Page of Faxed Arabic Text

Another problematic feature of Arabic script is the frequent use of ligatures, in which two (or more) adjacent characters are written as a third, entirely different, character. Clearly, character-level segmentation cannot be performed on two characters written as a ligature. Other problems which hamper efforts to recognize Arabic text include the potential presence of diacritical marks in the written text; the large number of characters that have associated dots whose size is in the range of speckle and noise; the presence of Kashidas, the stretch character, which is used for text justification and has no inherent meaning; the fact that between-word spacing is inconsistent and is often in the same range as intra-word spaces; and finally, grammatical practices in the Arabic language in which prepositions and determiners may actually become part of the word. Figure 1 shows a complex page of Arabic that has been faxed.

Another aspect of the problem, beyond the idiosyncrasies of Arabic text, lies in the fact that this system has been designed to tackle the entire gamut of Arabic text with respect to quality, resolution, font variability, page layout and media — any incoming page must be handled appropriately. What makes the situation even more challenging is that this must be done in a fully automated fashion with absolutely none of the user interaction common in most traditional OCR systems.

In the next section we discuss the algorithms employed in processing and recognizing pages of Arabic text using segmentation-free techniques.

2 System Concepts and Architecture

Our system for Arabic text recognition utilizes many concepts that may be generalized to the recognition of other machine-printed or handwritten text, notably the whole-word, segmentation-free approach to recognition. However, it is abundantly evident that to attain a high accuracy recognition system for any language it is necessary to focus on, adapt to, and exploit those properties of the language that will benefit various aspects of the processing including page decomposition, text segmentation and recognition. No system that claims to be able to handle many disparate languages will ever achieve the same performance as a tightly-coupled system that has been designed specifically to deal with one language at all levels. This notion has been the driving force behind the design of our Arabic text recognition system.

The Arabic recognition system is designed to accept any input image in TIFF format and process it appropriately — e.g., if the image contains any machine-printed Arabic text this will be extracted and recognized. If no Arabic text is detected the image will be rejected. The system has the capability of processing grayscale or binary images as necessary. The input data may be faxed, photo-copied, scanned or produced electronically. Any resolution, typesize and quality of input is handled to the degree possible. No assumptions are made about page orientation or resolution; information about the page is automatically detected and applied in the recognition process.

Typically, a page is sent to the Arabic recognition system and processing is initiated. First it is determined whether the page is binary or grayscale. If it is grayscale, it is binarized. Next an assessment of *page type* and quality is performed. Page type can be newspaper, magazine, business document or plain text. The page may have also been copied or faxed. If a fax banner is detected, the banner is removed, and any Latin text is recognized and saved. At this point it is appropriate to perform the removal of low levels of speckle and noise. If the page type is *Fax* and it was sent in the low resolution mode (which is 198 dpi per row, but only 96 dpi in the y-direction) this is detected and rows are replicated in order to make the aspect ratio 1.

Next, we determine the major *orientation* of the page, i.e., whether it is rightside up, upside down, sideways to the left, or sideways to the right. The assessment of orientation is with respect to any text present in the image. Once the page has been re-oriented, a finer determination of the *skew* of the image ($\pm 45°$) is determined using the Hough transform and the image is resampled.

At this stage we have an image which is properly aligned. It is then determined whether this page contains Arabic text and whether that text is machine-printed or handwritten. If the page does not contain any Arabic it is rejected, otherwise structured noise such as lines and pictures are removed from the image. The cleaned page is then decomposed into zones of text which are then broken into lines and words.

Once word images have been extracted from the input image, they are passed into one or more of the recognition subsystems. If more than one subsystem is applied, then recognition results are combined using a decision strategy to produce a single recognition response. Figure 2 outlines the architecture of the Arabic recognition system.

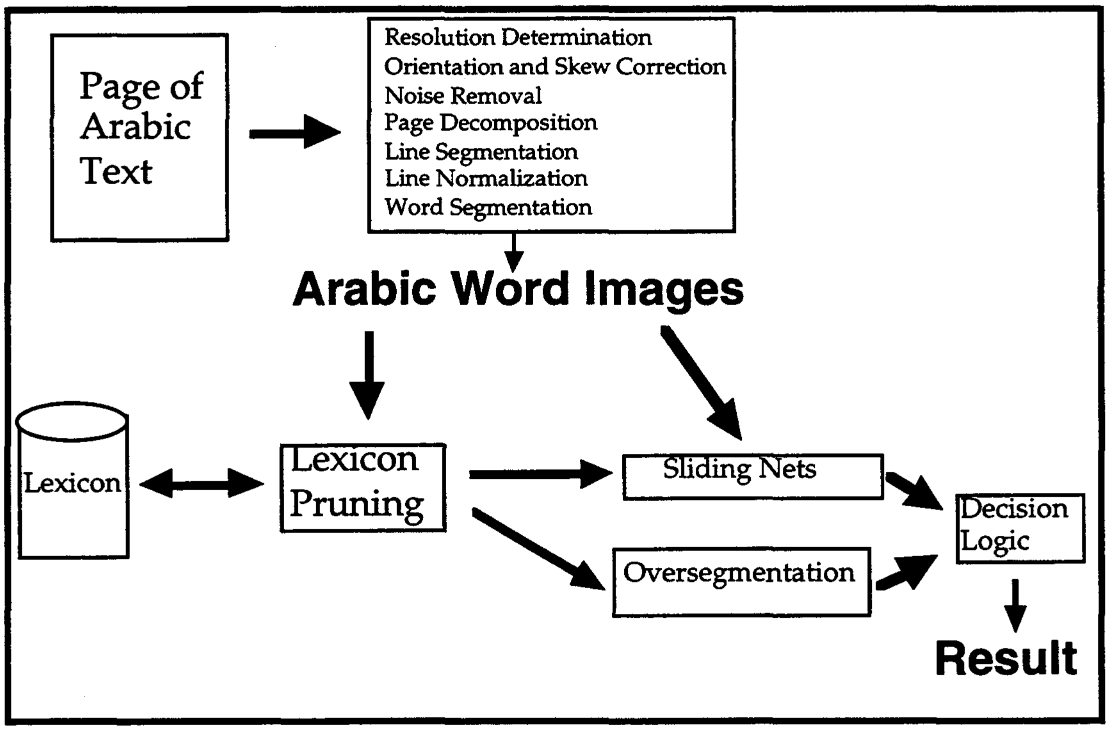


Figure 2: Summary of Arabic Recognition System Architecture

2.1 Preprocessing Subsystems

The preprocessing module consists of all modules which extract information about the incoming image, normalize various aspects of the image, clean noise and culminate in either the rejection of the image or the presentation of the processed image for segmentation. The detection, correction and removal stages involved in this subsystem are as follows:

1) Bits-Per-Pixel

2) Foreground / Background

2) Page Type

3) Resolution/Aspect Ratio

4) Orientation/Skew

5) Image-Domain Alphabet

6) Machine-Print vs. Handwritten

7) Structured Noise

An incoming image is first subjected to various simple tests to deal with the image appropriately. First, a histogram of the image is examined to determine whether the image is binary or 8-bit grayscale. If it is grayscale, the image is reduced to binary using a locally adaptive thresholding technique. If the initial image is binary, we examine the ratio of background to foreground pixels in order to guarantee that the pixels representing background are zero and those representing text are nonzero.

Next we determine whether or not this page has been faxed. This is done by searching for the fax banner that usually occurs at the top of a page and has other distinct characteristics such as fairly consistent height and fixed-pitch Latin characters. If it is determined that the page is *Fax,* then a check is made to see whether this page was sent in low resolution mode, which alters the aspect ratio of text. This is done by examining the connected components from the image that are approximately the size of characters or small subwords. These segments are fed into a neural network that has been trained to discriminate between segments (at any orientation which is a multiple of 90 degrees) that have a normal aspect ratio and those that have an aspect ratio of 1:2. A vote is taken over all collected segments and the majority defines the aspect ratio of the page. If the page is determined to be low resolution, the page is upsampled in the y-direction, not simply by row replication, but using an algorithm designed to make the text more amenable to recognition processes.

Once the aspect ratio of the page is known to be 1:1, a page-level feature vector is extracted that is sent to a neural network which is trained to distinguish those pages with simple layouts from those with more complex layouts. A simple page consists of a single column of lines of text which are easily segmented; complex pages have multi-columns, figures, lines, charts, etc., that require more processing to handle.

Next, another neural network which has been trained to discriminate between segments presented at the four orientations of 0, 90, 180, and 270 degrees, is used on the connected components and a vote is taken to determine overall orientation. If the orientation is nonzero, then the page is rotated. Then a determination of fine skew is initiated using the Hough Transform to determine if the image still slightly askew. Any detected skew is corrected and the image is resampled and cropped. At this point, the image is rightside up, text is aligned horizontally,

and the resolution is correct.

Because a recognition system is potentially just one part of an overall multi-lingual recognition capability, it is important to know that each actually contains the type of text that it was designed to recognize. Before going any further, the system attempts to discern that the script or alphabet used in the page is, in fact, Arabic. This is again done by using the powerful method of analyzing connected components using a neural network that has been trained to recognize various major alphabet classes including Arabic, Latin, Ideographic (Kanji/Chinese/Korean) and Cyrillic. if the majority of the components on a page are not determined to be Arabic the page is rejected. Finally, we verify that the Arabic text present on the page is machine-printed, not handwritten, by again using a connected component approach with neural nets that discriminate machine-printed vs. handwritten Arabic.

The final stage of preprocessing is to remove all the noise and structured noise in the image and leave only text. *Noise* may be speckle, junk induced in the corners through photo-copying, etc. *Structured noise* consists of non-text items that may interfere with segmentation and recognition such as long vertical and horizontal lines, figures, pictures, charts, etc. We use run-length encodings of the image and mathematical morphology techniques to robustly detect and remove noise from the image. The result of the preprocessing stage is an image suitable for segmentation into lines and words.

2.2 Segmentation Subsystems

We are now prepared to break the image into text zones, individual lines, and words.

2.2.1 Page Decomposition

Only pages whose page type has been determined to be *complex* need to be processed by the page decomposition module. This module determines the major typesizes of text occurring in an image and separates the data into *planes*, each containing a different size of text.

These *planes* are then broken into individual zones of paragraph-size using horizontal and vertical projections.

2.2.2 Line Segmentation and Normalization

This module takes an entire *simple* page or the constituent zones from a *complex* page and breaks these into lines. Our language-specific line normalization algorithm uses preliminary recognition of un-normalized characters to estimate the best choice of scaling factor, which is then applied to the line as a whole. The first step is to look for common isolated characters and subwords in the unnormalized line. The features used for this initial recognition are different than those used after normalization. Statistics have been gathered for 24 characters, generating distributions of expected placement of character tops and bottoms with respect to the baseline. Recognized characters are then compared to these distributions to determine the best scale factor for the line as a whole.

2.2.3 Word Segmentation

Normalized lines are then broken into individual words for recognition. Word segmentation is done by using image-domain properties of Arabic text. Of most value is the fact that Arabic text may be broken into *subwords* which are defined as components separated by horizontal space. The concept of subwords arises because of the six or seven Arabic characters that are called non-connecting in that they should never connect to a stroke following on the left Subwords always occur in a word with embedded non-connecting characters.

Segmenting a line image of Arabic text into subwords is much easier than word segmentation. Given the fact that we can accurately obtain the subwords from a line, we utilize recognition capabilities to enhance our ability to classify gaps occurring in the word image as *between* words or *within* words. The process entails considering each subword from a line image and the adjacent gaps. Recognized elements from the line allow us to more accurately categorize gaps and thus segment words.

2.3 Recognition Subsystems

Each of the two Arabic recognition subsystems uses a different approach to recognizing word images without character-level segmentation. The two subsystems are referred to as the oversegmentation system and the sliding neural network system.

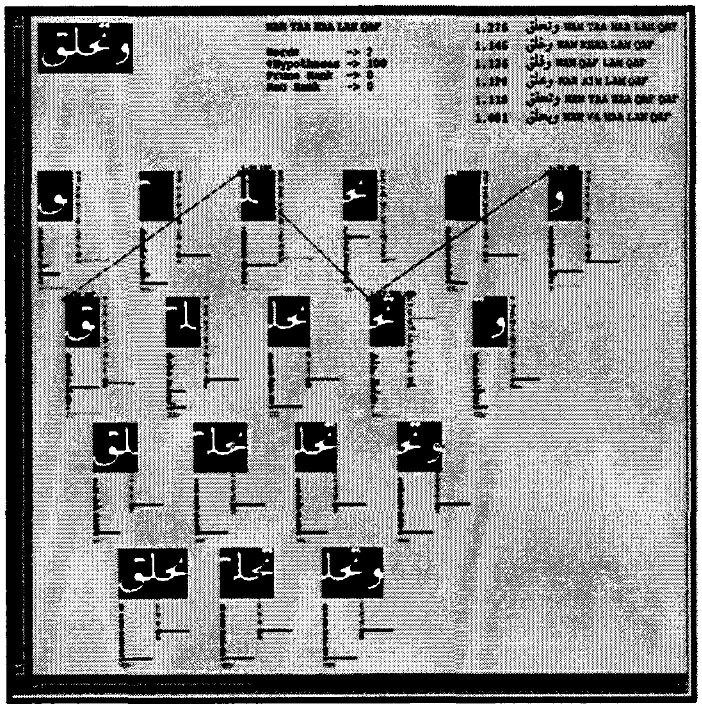


Figure 3: Segment Unions from Oversegmentation Subsystem

2.4 Lexicon Use

Each recognition subsystem makes use of a lexicon of known Arabic words to constrain its recognition. The current system uses a lexicon of roughly 50,000 words compiled from our truthing efforts. Furthermore, this lexicon is pruned for each word-image so that a minimum number of possibilities need be considered by the recognition systems. Lexicon pruning has the double benefit of improving the recognition results and increasing the speed of the system.

Pruning techniques have been implemented and incorporated in the Arabic system. Each of these methods is essentially an abbreviated version of a full-scale recognition system. The purpose of the abbreviated version is to quickly eliminate words in the lexicon which are unlikely to be the correct choice, and let a full recognition system discriminate between the remainder of the choices.

2.3.1 Oversegmentation

The oversegmentation system is based on the process of segmenting a word-image at local extrema in the image contours. Segmenting in this fashion breaks the image up into pieces which are *smaller* than individual characters. These segments are combined into all possible unions of one, two three or four segments. Each of these unions is scored by a neural net trained on whole-character images, and these scores are saved in an array. A dynamic programming algorithm is then applied to these scores to determine the best-cost way to assemble those unions into a spelling of a word from the word lexicon. The word lexicon is pruned for each word so that only a fraction of the possible words need be tested in this way. The word with the highest best-cost dynamic programming score is presented as the oversegmentation system's top choice. Figure 3 shows an array of segment unions created by the oversegmentation system. The word image from which the segments were extracted is in the upper left corner of the figure.

2.3.2 Sliding Neural Network

The sliding neural network system utilizes neural networks in a two-stage approach. The first stage is to detect probable locations of character-centers and the second is to recognize characters at those probable locations. During both stages, the neural network views a window which slides across the word image and produces a recognition signal at each location along the image. This technique obviates the need explicit character segmentation. The character hypotheses from the second stage are then combined with the aid of a dynamic programming algorithm, which uses a pruned list of words from the lexicon to constrain its search for the most probable word hypothesis. An example of the two-level application of sliding neural nets with recognition-signals is shown in Figure 4.

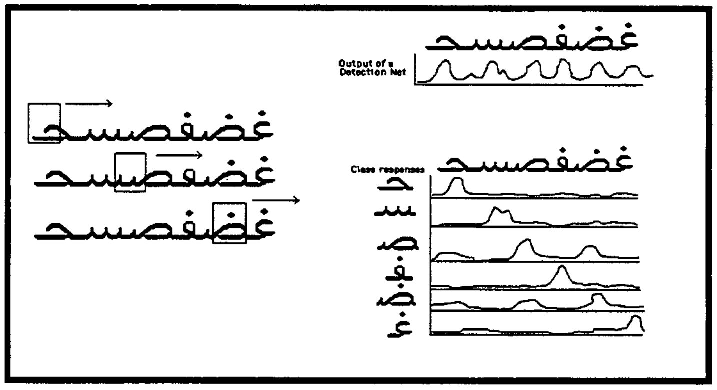


Figure 4: Illustration of Two-Level Sliding-Net Recognition

3. Training Data

Each of the recognition subsystems was trained on a set of data collected during two data collection efforts. The first data collection entailed roughly 700 pages of Arabic text collected from 45 documents from the University of Michigan's Arabic library. The text was chosen to include a range of fonts and sizes, including typewriter as well as type-set characters. Digitization produced varying degrees of noise quality, encompassing both high quality and noisy text. A second collection yielded another 400 images or variable quality binary data. The truth text for the data was entered in Unicode.

4 Recognition Results

The Arabic system performance was gauged on a random subsample of recent testing data. The system was run on 12455 words with full lexicon coverage and a result of 81% word recognition rate was obtained. This can be extrapolated to an approximate rate of 96.5% character recognition. This test assumes perfect word segmentation, but another test was run which better gauges the effect of preprocessing and segmentation. This test measured the recognition accuracy across 15 full pages of Arabic text with incomplete lexicon coverage and an average rate of 75% was achieved (95% character recognition rate).

5 Status

Current work on the Arabic recognition system is focusing on those areas that seem to be most beneficial for continued improvement. One of the most difficult tasks in the processing of Arabic text is in accurately breaking images of lines of text into images of words. Traditionally in text processing of Latin-based languages, a statistical analysis of the horizontal gaps occurring in a line image has been adequate to separate lines into words. In Arabic text, the fundamental consistency present in Latin script is lacking — gaps between words and subwords may be of the same order of magnitude whereas in Latin text the difference between gaps within words and between words is quite evident. This inconsistency has lead to the need for other approaches to word segmentation that more successfully utilize the characteristics of the Arabic language. Another possibility is the use of line-level techniques for overcoming the errors inherent in Arabic word segmentation.

References

[1] Trenkle, J.M., Gillies, A.M, Schlosser, S.G., Erlandson, E.J., Arabic Character Recognition, *Proceedings of the Symposium on Document Image Understanding Technology*, pp. 191-195, Bowie, Maryland, 1995.

[2] Trenkle, J.M. and Vogt, R.C. Word recognitin for Information Retrieval in the Image Domain. *Symposium on Document Analysis and Information Retrieval,* Las Vegas, April 1992

[3] Almuallim, H. and Yamagushi, S. A Method of Recognition of Arabic Cursive Handwriting. IEEE Transactions on Pattern Analysis and Machine Intelligence. Vol. PAMI-9, No. 5, September, 1987

[4] Amin, A.,Masini, G., and Haton, J.P. Recognition of Arabic Words and Sentences. *Proceedings of 7th International Conference on Pattern Recognition,* Montreal, Quebec, 1984

[5] Amin, A., and Masini, G. Machine Recognition of Multifont Printed Arabic Texts. *Proceedings of the 8th International Conference on Pattern Recognition,* Paris, France, 1986

An automated system for numerically rating document image quality

Michael Cannon, Judith Hochberg, Patrick Kelly, and James White
Los Alamos National Laboratory

Abstract

As part of the Department of Energy document declassification program, we have developed a numerical rating system to assess the quality of a document image. The rating algorithm produces scores for different document image attributes such as speckle and touching characters. We have found that our quality measures are sufficiently meaningful that the OCR error rate for a document can be predicted from a weighted sum of the measures. The predicted OCR error rate will be used to screen documents that would not be handled properly with existing document processing products. The individual quality measures indicate how a document image might best be restored.

1 Introduction

The Department of Energy has undertaken the classification review of 337 million pages of legacy documents dating back to the Manhattan Project. Present plans call for the scanning of each document into digital image form. Some documents are of good quality, but the quality of others is degraded by photocopying, FAXing, carbon-copying, and aging fibrous paper. Automatically computed quality measures will be used to determine which documents can be successfully OCR'd and which must be first digitally enhanced and how.

We find that we can gauge the accuracy of our quality measures by using them to predict the OCR error rate. We have analyzed both a sample corpus of degraded documents that we produced ourselves by repeatedly photocopying selected pages from a book, and a small corpus of actual declassified DOE documents.

2 Data

In order to create a quality assessment algorithm under controlled conditions, we first created a corpus of documents spanning a range of gradually decreasing document image quality. We did this by repeatedly photocopying a page from a book, so that we had nine versions of it, the original and eight following generations. Each successive generation was increasingly plagued with common attributes of lower quality document images: speckle, fattened brush strokes, and touching characters. We also created a second set of degraded documents by repeatedly photocopying a second page from the book in the same manner, producing a total corpus of 18 documents.

We computed the OCR error rate (the percentage of incorrect words) for each photocopy generation of our two sets of documents. This was done using the Caere OmniPage Pro OCR package and comparing its output with the original page. We averaged the error rates of the two document sets together, generation by generation. As shown in Figure 1, the error rate increased with each generation.

Our current DOE corpus consists of 144 declassified DOE documents, their images, and OCR error rates*. The error rate distribution is summarized in the histogram in Figure 2. This corpus represents problematic documents, hence the unusually high OCR error rates.

3 Quality Measures

We developed quality measures for the document images through an analysis of our sample dataset of photocopied images. We began by computing histograms of the sizes of white and black connected components for each of the nine photocopy generations; sample plots of the zeroth and fifth generation histograms are shown in Figures 3 and 4.

* The documents were scanned by Prof. Nathan Brener, LSU, and OCR'd by Prof. Tom Nartker, UNLV. Prof. Nartker also computed the OCR error rates from keyed text provided by Prof. Brener.

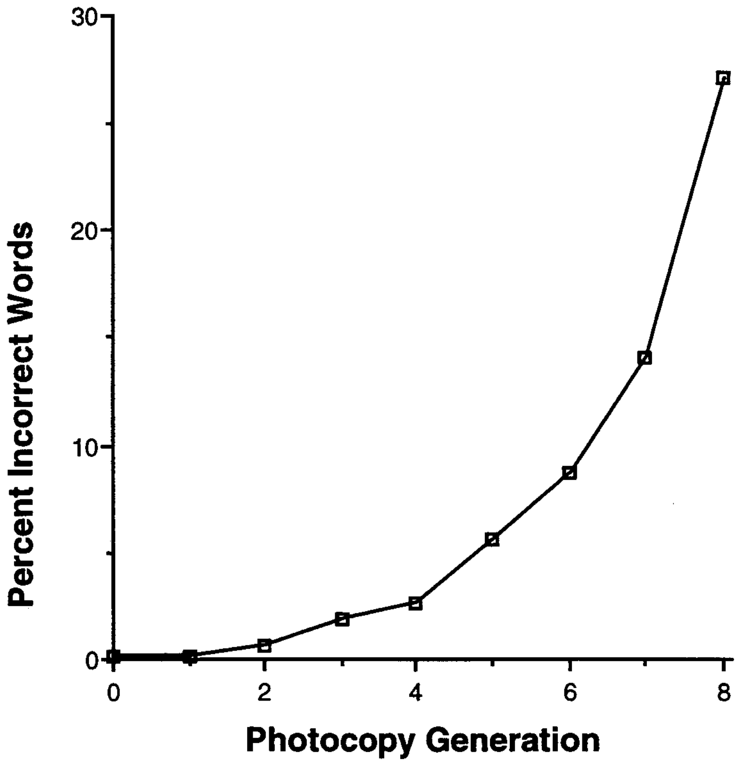


Figure 1. OCR error rates for the nine generations of documents in our test corpus.

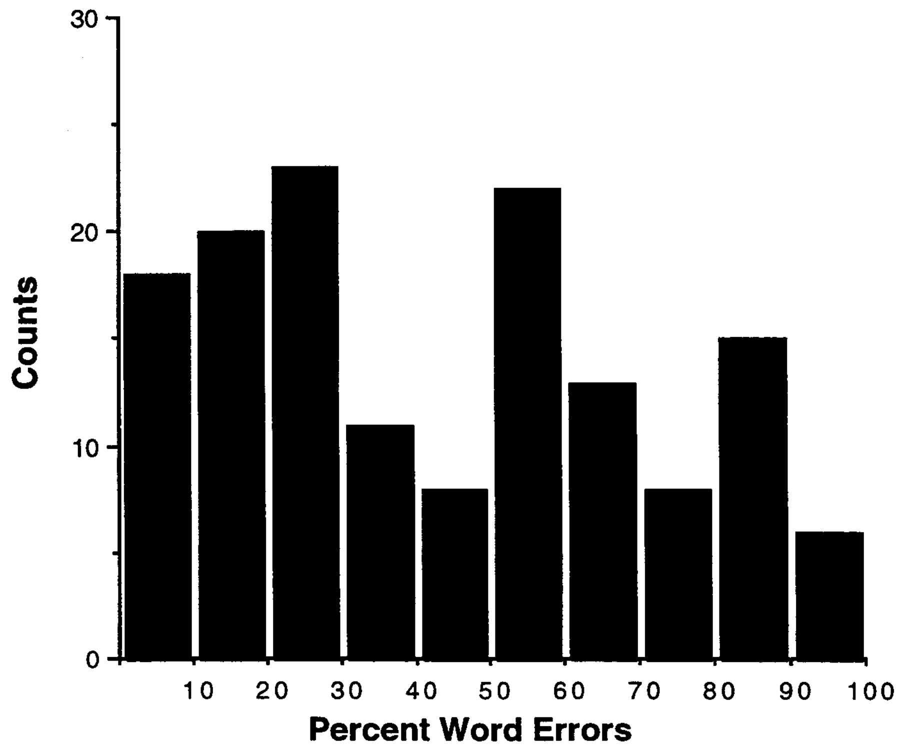


Fig. 2. Histogram of the OCR error rates for the DOE corpus

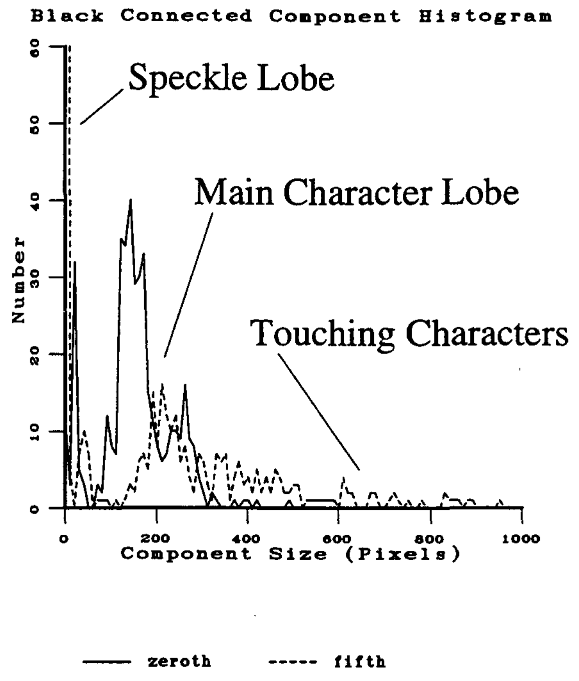


Figure 3. Histograms of black connected component sizes for the zeroth and fifth generation photocopies of one of our sample documents. Annotation indicates portions of the histograms that yield clues to the quality of the documents.

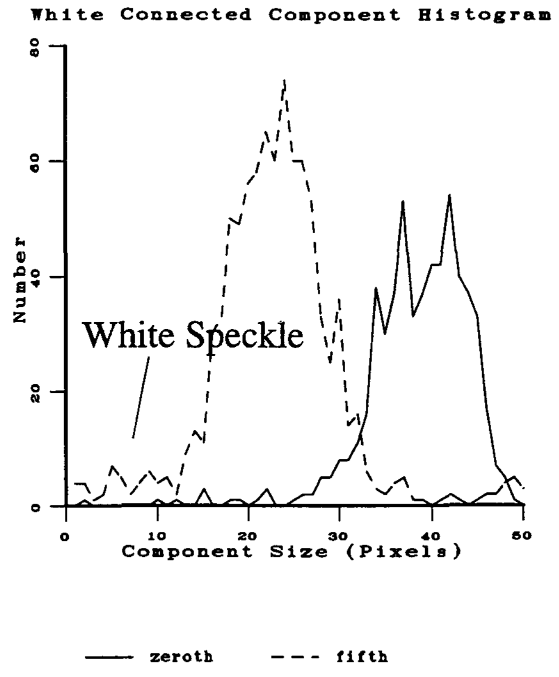
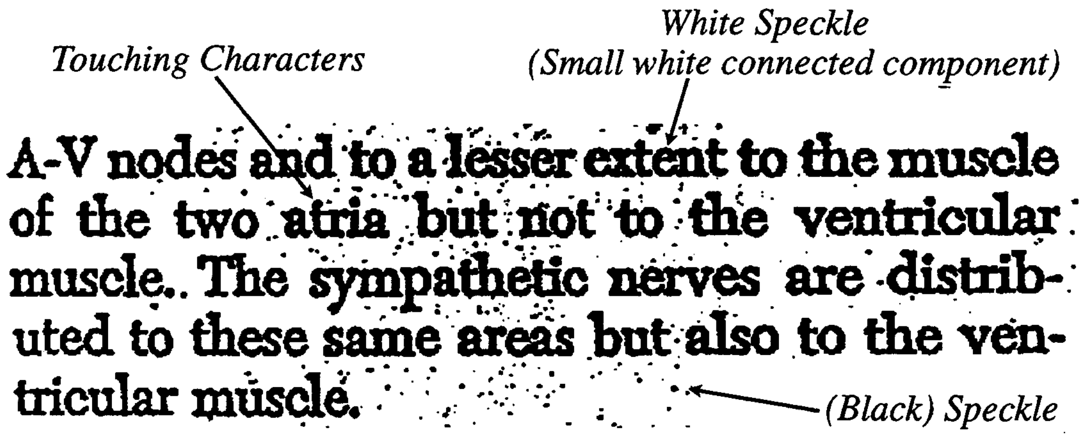


Figure 4. Histograms of white connected component sizes for the zeroth and fifth generation photocopies of the same document as that in Figure 1. Very small white connected components are labeled as White Speckle.

We observed several interesting features of these histograms:

- The number of very small black connected components increased with photocopy generation due to the increasing incidence of background speckle.
- The number of very small white connected components increased as white connected components shrank in size due to the fattening of black brush strokes, and as neighboring characters began to touch and create still more small white connected components.
- Black connected components greater than 600 pixels in size occured as characters began to touch.
- The main lobe of black connected components, initially centered at approximately 150 pixels in size, broadened and shifted to the right as brush strokes fattened.

Some of the corresponding degradations in the images from which these histograms were computed are shown in Figure 5.


Touching Characters
White Speckle
(Small white connected component)
(Black) Speckle

A-V nodes and to a lesser extent to the muscle of the two atria but not to the ventricular muscle.. The sympathetic nerves are distrib-uted to these same areas but also to the ven-tricular muscle.

Figure 5 A sample fragment of degraded text, showing black speckle, white speckle due to shrinking white connected components, and touching characters.

Based on these observations, we derived three easily computed quality measures that quantitate the decreasing quality of each photocopy generation. A fourth one is derived from observing fragmented characters in the corpus of DOE documents.

1. *Black Speckle Factor* (BSF). We count the number of black connected components that are less than or equal to ten pixels in size as a measure of the amount of black background speckle in the document image. This count includes periods and dots over the letters *i* and *j*, a minimal perturbation. We normalize the Speckle count by the total number of black connected components in the image that are greater than 100 pixels in size.
2. *White Speckle Factor* (WSF). We measure the incidence of white connected components (actually the increase in the number of very small white connected components) using a method put forth by Blando, et. al.[1]. We count the number of white connected components that are less than or equal to 3x3 pixels in size, normalized by the total number of white connected components in the image. Blando refers to this feature as the White Speckle Factor (WSF), a term we have retained in this paper. We limited our count of white connected components to those less than 300 pixels in size.
3. *Touching Character Factor* (TCF). We count the total number of black connected components greater than 600 pixels in size. We normalize the count by the total number of black connected components in the image that are greater than 100 pixels in size.
4. *Broken Character Factor* (BCF). We count the number of black connected components between 10 and 100 pixels in size *along a line of text* and normalize it by the total number of black connected components along the line.

The first three measures are plotted in Figure 6 as a function of photocopy generation. The plotted numbers are derived from our two sets of degraded documents, averaged together generation by generation. The fourth feature, Broken Character Factor, did not apply to our sample corpus.

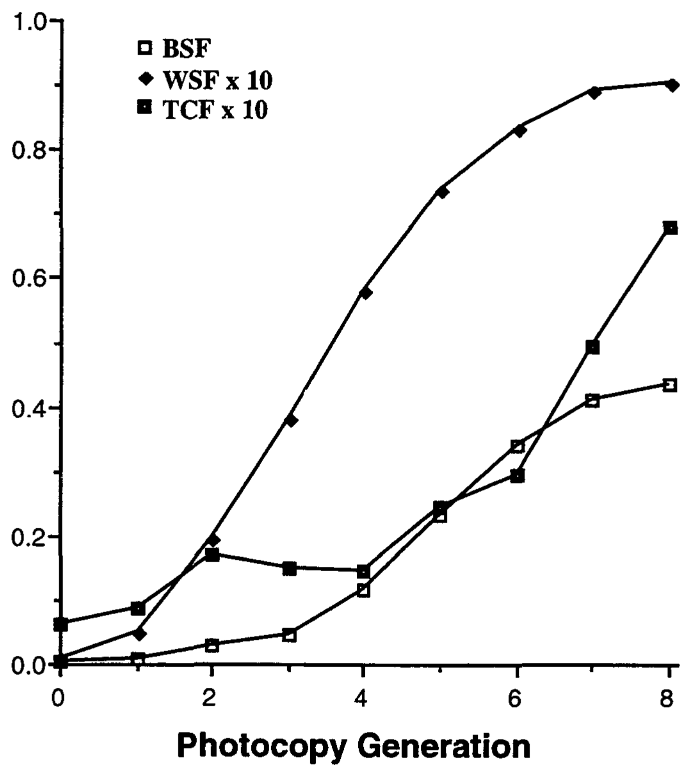


Figure 6. The values of our three document image quality features, Black Speckle Factor (BSF), White Speckle Factor (WSF), and Touching Character Factor (TCF), plotted over the nine generations of our document corpus.

For the DOE corpus, we calculated all four measures, including the Broken Character Factor, because broken characters were observable in these documents. We computed their correlations with each other and with OCR error rate. These are summarized in Figure 7. All four measures correlated significantly with OCR error rate, White Speckle Factor and Touching Character Factor most strongly. Within the measures, the strongest correlations were the positive correlation between Touching Character Factor and White Speckle Factor, and the negative correlation between Touching Character Factor and Broken Character Factor. Both of these are intuitively reasonable, as touching characters create white speckles, and are the opposite of broken characters. In addition, Black Speckle Factor was correlated positively, though weakly, with White Speckle Factor and Broken Character Factor.

	Error	BSF	WSF	TCF	BCF
Error	1.00	0.22*	0.54***	0.61***	0.19*
BSF		1.00	0.17*	0.15	0.24**
WSF			1.00	0.34***	0.13
TCF				1.00	-0.41***
BCF					1.00

Figure 7. DOE Corpus: Correlations among quality measures and OCR error rate. Significance of $p<0.5$, 0.05, and 0.005 indicated by *, **, and ***.

4 OCR Error Rate Prediction

The significant correlations between quality measures and OCR error rate suggested that the measures could be used to predict the OCR error rate of a document image. A good prediction would demonstrate the utility of the quality measures.

We hypothesized that a linear combination of the three measures would be an effective error rate predictor. That is, for the photocopied dataset (where BCF does not apply),

$$\text{OCR error rate} = \alpha\cdot\text{BSF} + \beta\cdot\text{WSF} + \delta\cdot\text{TCF} + \gamma, \quad (1)$$

where α, β, δ, and γ are determined from a training set of data, and the BSF, WSF, and TCF measures are computed from the document image in question.

To determine α, β, δ, and γ from our corpus of nine generations of increasingly degraded photocopies, we form a linear system of equations

$$z_i = \alpha w_i + \beta\cdot x_i + \delta\cdot y_i + \gamma, \quad (2)$$

where z, w, x, and y represent error rate, BSF, WSF, and TCF, while i represents the photocopy generation, 0 through 8. We solve the system to obtain the α, β, δ, and γ that minimize the mean square error in predicting z.

After we obtained α, β, δ, and γ , we used them to "predict" the OCR error rate for the nine generations of photocopied documents in the sample document corpus based on the three quality features derived from each document. In actuality, because of the paucity of data, we used a jack knife approach. We omitted one document from the estimation of the coefficients and then used those coefficients to predict the error rate of the omitted document. We repeated the experiment by omitting another document, and so on. These results are plotted in Figure 8, overlaid on the actual OCR error rates. The correlation between actual and predicted error rates was 0.97 (p<0.005). This high correlation is a measure of the ability of the three measures to capture the quality of a document image.

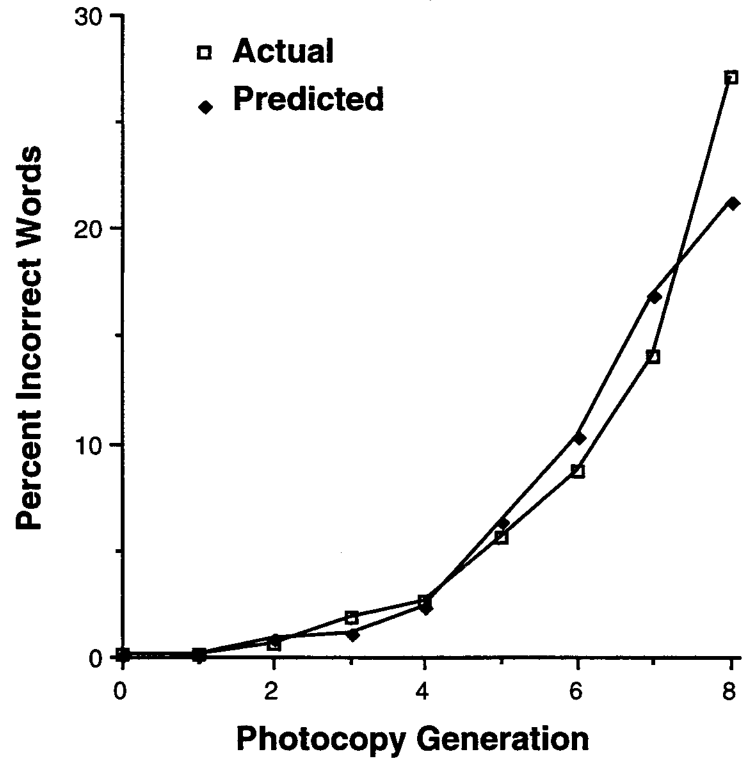


Figure 8: The actual OCR error rate from Fig. 1, overlaid with the predicted error rate based on our three quality measures.

We performed a similar analysis for the DOE corpus, including the fourth quality measure -- the Broken Character Factor -- since these documents, unlike the starter corpus, contained broken characters. We used the method of Equation (1) to predict the OCR error rate using the jack knife approach. We included quadratic terms this time because we found it gave a better prediction. The predicted OCR error rates are shown in Figure 9, plotted against the actual error rates. Again, there was a strong correlation between actual and predicted error rates: r = 0.84, p<0.005.

Figure 10 shows the difference between actual and predicted errors, sorted by the size of the difference. Some insight as to how our quality features can be improved can be gained by looking at the documents at the far right of the curve, corresponding to the largest differences in OCR error rate prediction. Portions of these documents are shown in Figure 11, along with the document having the lowest difference.

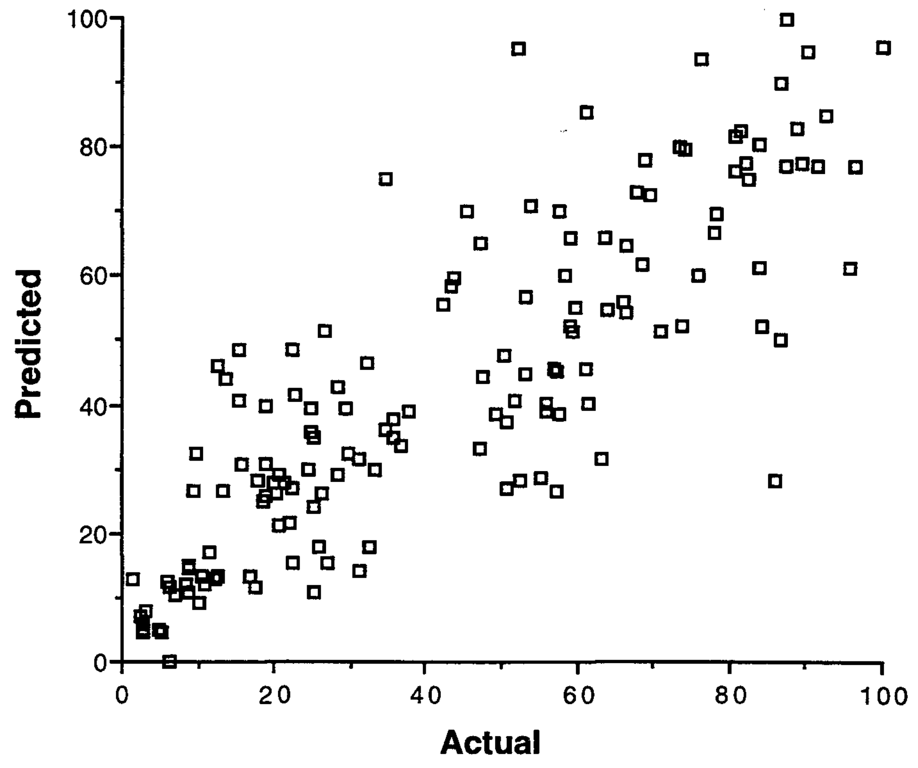


Figure 9: The actual OCR error rates of the 144 DOE documents are plotted against the predicted error rates. The closer the predicted error rates lie along a 45^{o} line, the better the integrity of the four quality measures upon which the prediction is based.

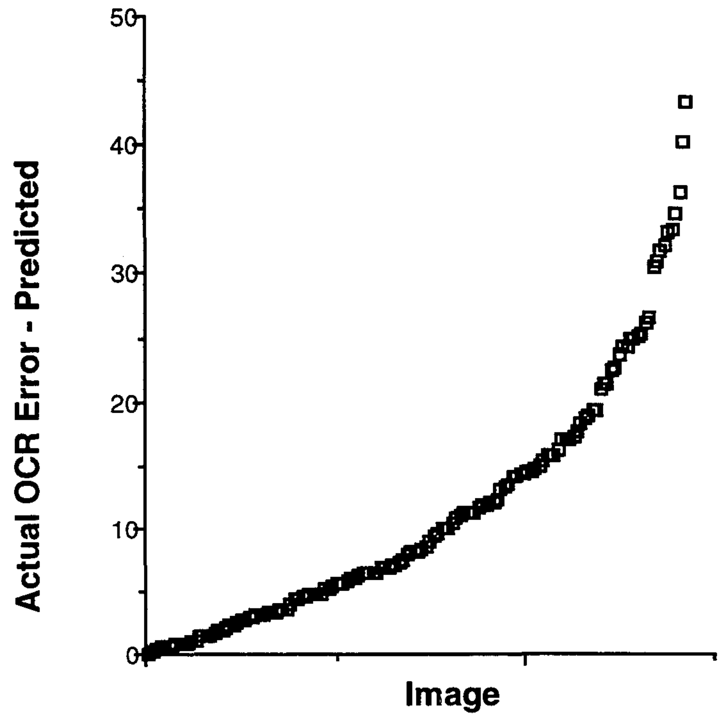
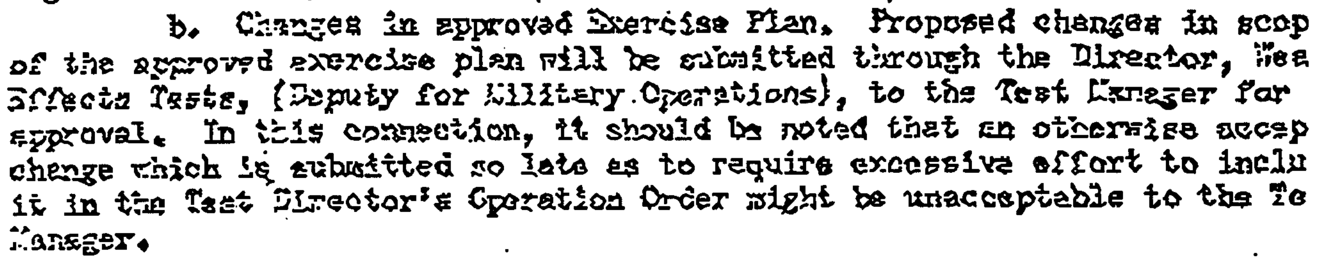
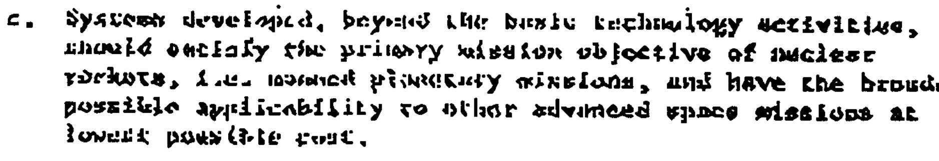


Figure 10: The differences between actual and predicted error are plotted here, sorted by magnitude.

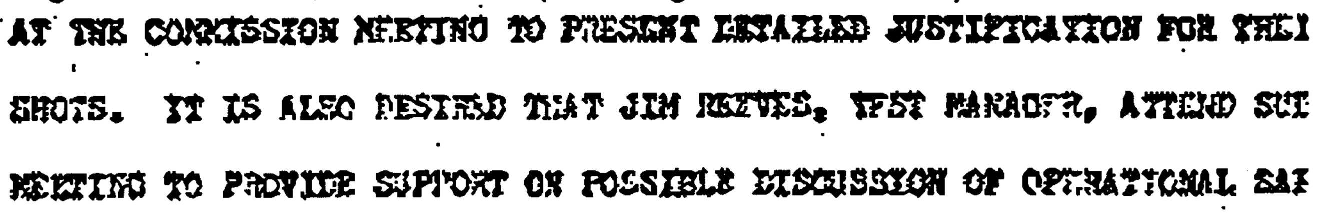
Page 112: Actual 26%, Predicted 26% (Smallest Prediction Error)


b. Changes in approved Exercise Plan. Proposed changes in scop of the approved exercise plan will be submitted through the Director, Wea Effects Tests, (Deputy for Military Operations), to the Test Manager for approval. In this connection, it should be noted that an otherwise accep change which is submitted so late as to require excessive effort to inclu it in the Test Director's Operation Order might be unacceptable to the Te Manager.

Page 287: Actual 35%, Predicted 75% (Largest Prediction Error)


c. Systems developed, beyond the basic technology activities, should satisfy the primary mission objective of nuclear rockets, i.e. manned planetary missions, and have the broadest possible applicability to other advanced space missions at lowest possible cost.

Page 184: Actual 87%, Predicted 50% (Next Largest Prediction Error)


AT THE COMMISSION MEETING TO PRESENT DETAILED JUSTIFICATION FOR THEI

SHOTS. IT IS ALSO DESIRED THAT JIM REEVES, TEST MANAGER, ATTEND SUC

MEETING TO PROVIDE SUPPORT ON POSSIBLE DISCUSSION OF OPERATIONAL SAF

Page 395: Actual 96%, Predicted 61% (Third Largest Prediction Error)

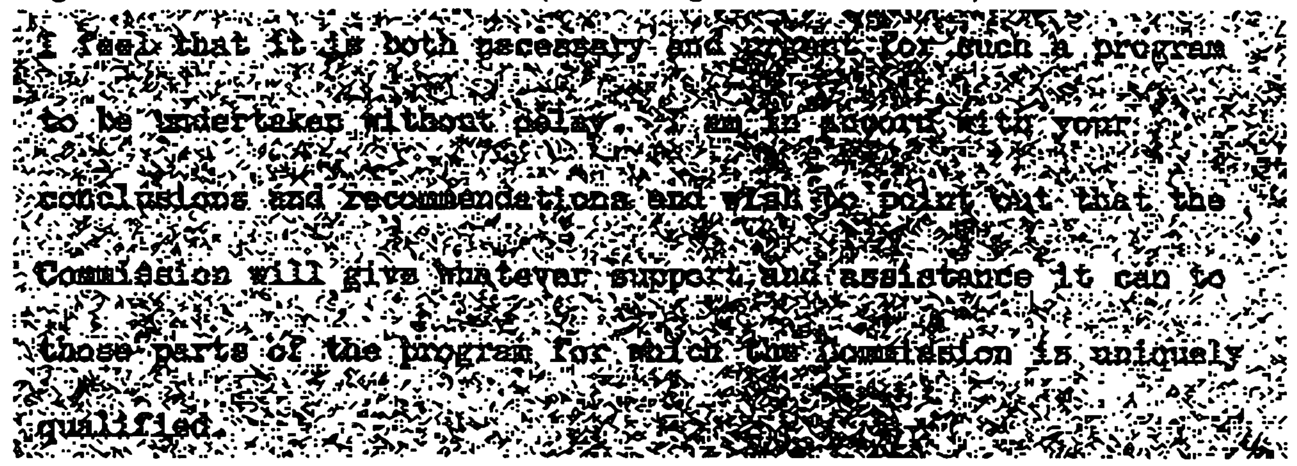

I feel that it is both necessary and urgent for such a program to be undertaken without delay. I am in accord with your conclusions and recommendations and wish to point out that the Commission will give whatever support and assistance it can to those parts of the program for which the Commission is uniquely qualified.

Figure 11: Portions of the documents having the smallest and largest errors in OCR error rate prediction..

It can be seen that the largest errors were made in predicting the OCR error rate if the font was too big or too small relative to a normal typewritten font, or if there was a large amount of background speckle. These documents and OCR prediction errors are representative of the corpus of 144 documents, and they indicate how our quality measures can be improved.

5 Conclusions

We have developed four quality measures that appear to accurately assess the quality of a document image. We have evaluated the integrity of the measures by using them to predict the OCR error rate that we might expect on a particular document. This evaluation has given us insight into how well the quality measures perform and how they can be improved.

The predictive capability of our four document quality features is encouraging. At the same time, it seems clear that their utility will increase if they are made font-size-dependent, and if the Black Speckle Factor is improved. True improvements in these features will be signaled by an improvement in OCR error rate prediction. We hope to describe the results of these improvements in the oral presentation of the paper.

Acknowledgments

Nathan Brener and Tom Nartker, in addition to supplying us with the DOE document corpus, have contributed many meaningful conversations and insight to our work. Tom Burr, Los Alamos, has given us assistance in using proper statistical methods for our data. Tom Curtis, US Department of Energy, has kindly funded our project.

Reference

1. Luis R. Blando, Junichi Kanai, Thomas A. Nartker, *Prediction of OCR Accuracy Using Simple Image Features*, Proceedings ICDAR '95, Montreal, Canada, p319.

Multimedia

Document Image Understanding Research at Penn State

R. Kasturi H. Luo U. Gargi
Dept. of Computer Science & Engg.
The Pennsylvania State University
University Park, PA 16802
kasturi@cse.psu.edu

S.H. Strayer
HRB Systems Inc.
P.O. Box 60, Science Park Road
State College, PA 16804
shs@hrb.com

Abstract

Document image understanding has been a field of research at Penn State for many years. We present the current activities of the group which lie in three areas: the development of a graphics recognition and interpretation software toolkit (in collaboration with HRB Systems Inc.); interpretation of engineering drawings; and a recent effort on the extraction of text from digital video. All three projects are still in development and hence we discuss directions and goals.

1 GRIT: The Graphics Recognition and Interpretation Toolkit

1.1 Introduction

Graphics recognition is a research area in Document Image Analysis concerned with the automated extraction of information from graphical elements in documents. These graphical elements may accompany text in a document page such as a logo in a business letter or a diagram in a journal article. They also appear in special document image types where the majority of the information content is expressed graphically such as an engineering schematic, a map, a fingerprint sample, or a musical score. Graphics recognition techniques are used in a variety of applications. These applications include such diverse activities as conversion of engineering drawings to CAD, development of graphics image databases utilizing query by sketch search and retrieval techniques, fingerprint classification and automated map reading. The Document Image Analysis Group at the Centre for Intelligent Information Processing (a joint research center founded by the Pennsylvania State University and HRB Systems, Inc.) began development of the Graphics Recognition and Interpretation Toolkit (GRIT) in mid 1995. The toolkit is being designed to provide a framework for research and development for graphics recognition algorithms. Specifically our goals are:

- to create a rich set of basic processing modules to support rapid development and prototyping of graphics recognition applications for toolkit users
- to provide a flexible, easily extensible framework (including data objects and library functions) for algorithm research which supports rapid coding of new functions for toolkit programmers
- to facilitate the exchange of research results and the evaluation of algorithm performance through the use of this common framework.

1.2 Aims of the Toolkit

Our long–term objective is to develop a complete, Document Image Analysis tookit tailored toward graphics recognition which will include modules for operations at a wide range of levels. These operations will include:

- Pixel–level Processing
    - Thresholding
    - Noise Reduction
- Page Layout Analysis
    - Skew Detection
    - Geometric Page Segmentation

- Feature Level Processing for Graphics Recognition
  - Text String Extraction
  - Solid Object Detection
  - Boundary Tracking for Recognition of Non–solid Bounded Objects
  - Line Tracking for Thin Objects
  - Critical Point Detection
  - Curve Detection/Fitting
  - Dashed Line Recognition
- Graphics Interpretation
  - Loop Tracking and Recognition
  - Shape Recognition
  - Hatching/Fill Pattern Detection
  - Interconnection Detection and Interpretation

1.3 Current Functionality

The toolkit is currently being implemented in Khoros, a standard design and analysis environment in the computer vision and image processing community. Khoros provides an intuitive graphical user interface and an extensive set of libraries and commands for signal and image processing and visualization applications. Other standard design tools were considered, most notably the Image Understanding Environment (IUE) [1] currently under development as a five–year, ARPA–funded effort. Since IUE was not scheduled to be available for several months after the beginning of our development cycle, Khoros was chosen as the initial development tool for the Graphics Recognition and Interpretation Toolkit. We will continue to track the on–going development of the IUE and will evaluate the feasibility of porting the GRIT to the IUE when the environment becomes available.

The current version of GRIT includes commands, data objects and functions to implement routines for a number of graphics recognition and document image analysis processes. Our focus over the first development phase has been on feature–level recognition. Our algorithms are implemented to work with the image data in a compressed format. We store the data for internal processing as an array of run–length encoded information. We have developed kroutines for the following processes:

- separation of text strings from graphical components using an algorithm developed by Fletcher and Kasturi [2] and subsequently enhanced by Kasturi, et. al. [3]
- recognition of solid objects using an erosion–dilation approach
- boundary–point detection
- a variety of line–tracking and detection routines
- critical point detection
- thinning
- curve–fitting
- dashed–line detection.

The list also includes utility routines for noise–filtering, scaling, smoothing, several logical operations (transpose, complement, intersect, union), and Hough–transform computation, Our kroutines currently operate on bitonal, TIFF images only. We are expanding our capabilities to support other image data types including color.

For programmers wishing to develop their own kroutines, we include a wide variety of library functions to facilitate the process. Our library includes functions for file I/O, data structure manipulation including creation, destruction, modification, copying, searching, and sorting. We also provide several statistical and mathematical computation functions such as histograms and least squares approximation. At the present time, the GRIT library contains about eighty functions.

1.4 Future Work

We have described an on–going research effort to create a tool for rapid development and analysis of algorithms and systems for graphics recognition research. We have three basic objectives for developing the toolkit. We seek to provide a large set of basic processing modules to support the rapid development and prototyping of graphics recognition applications. We support new algorithm development by ensuring that our toolkit is flexible and easily extensible. Finally we wish to facilitate the exchange of research results and the evaluation of algorithm performance by providing a common development environment. The development of a toolkit of this type was one of the goals established during the International Workshop in graphics recognition held at the Pennsylvania State University in August of 1995.

Our next focus will be on expanding the tool kit to support detection and recognition of features in color maps. Maps pose an especially challenging problem in document image analysis. They are designed to provide excellent representations of spatial data for interpretation by humans. Cartographers are trained to represent information in a succinct form using a variety of symbols, colors and notations. In addition, cartographers are allowed significant license in presenting the pertinent information without undue clutter. For example, lines are broken to accomodate text, insets are used to show greater details, and so forth. Maps also present a significant processing challange due to the size of the corresponding data set. At present we are extending our text/graphics segmentation algorithm to handle color composite map images.

2 Interpretation of Telephone Company Engineering Drawings

2.1 Introduction

This work on interpretation of several types of line drawings is sponsored by NYNEX telephone company. These drawings are typically 34" x 44" in size and are composed primarily of horizontal and vertical lines. After scanning at 300 DPI, a typical image would have 10,200 columns by 13,200 rows resulting in a raw image of over 130 MB. Due to the complexity of the drawings, a general purpose line drawing interpretation system is not useful. Our solution is to establish a system for interpreting each class of drawings efficiently, incoporating all the information available about the domain of the drawings. Basically, our algorithms first take advantage of run length encoding to reduce the amount of memory and the processing time. Then the primitives such as intersections for each class of drawings are extracted and combined to generate the higher level structures such as lines, boxes and tables.

2.2 Run Length Encoding Representation

Run Length Encoding(RLE) representation reduces the space needed to represent the raw image to around 2 MB, which permits loading the complete image in RAM. The reduction in storage space depends on the characteristics of the drawing. Most of our algorithms operate directly on this compressed representation. Although we need to locally restore the raster image from RLE representation to recognize special structures, the RLE representation significantly speeds up the subsequent processing.

2.3 Interpretation of Equipment Rack Drawings

Equipment drawings contain information about the equipment placed in racks in the central telephone exchanges. A small section of a typical drawing is shown in Figure 1. The following steps are applied to such drawings to extract the graphical information.

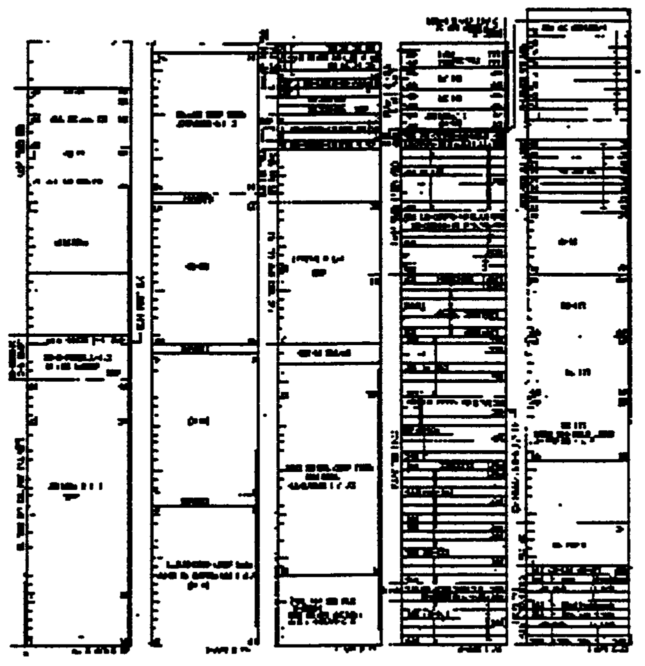


Figure 1: A typical equipment drawing.

1. **Intersection extraction:** The graphical elements of the equipment drawings are composed of horizontal and vertical lines. These lines interrelate to form boxes and table–like structures. Horizontal and vertical lines intersect to form any of structures shown in Figure 2. These different intersection patterns are first extracted from the images.

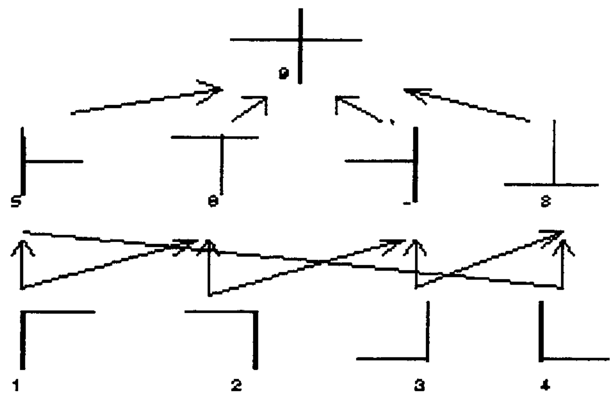


Figure 2: Hierarchy of intersections.

2. **Higher level structure extraction:** The following steps extract higher level structures such as boxes and tables by using the relationships between the relevant intersections.
   - If a corner of Type 1 is found, search regions are established to look for corners of Type 2, Type 3, and Type 4.
   - Lines between corresponding pairs of corners are searched; if they are present, a box is recognized.
   - Lines between each pair of left T (Type 5) and right T (Type 7) junctions are searched; if there is a line between them, an internal partitioning line is detected.
   - "+" (Type 9) may indicate a vertical line. Each time a "+" is found, the vertical line is tracked in both directions from the position of the "+".

   After the structures are extracted, text blocks are segmented, characters are recognized, and text is associated with corresponding graphical blocks.

2.4 Interpretation of Distributing Frame Drawings

Distributing frames provide a way to cross–connect switching and transmission equipment with subscriber and trunk lines. In addition, they provide electric protection and test access. Physically, a distributing frame is a two–sided structure. It is made up of horizontal bars (also called shelves or horizontal) on one side and vertical bars (or verticals) on the other. Drawings representing the distributing frames are processed as follows to extract information.

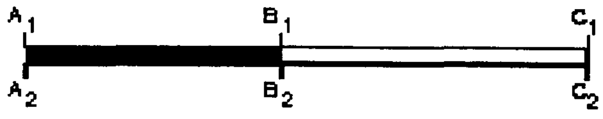


Figure 3: Three pairs of intersections of the line.

1. **Hierarchical representation of lines:** A distributing frame drawing contains horizontal and vertical lines. Filled areas between the two parallel lines indicate that this region of the vertical or horizontal is in use. Non–filled regions indicate available space. Consider the horizontal line segment shown in Figure 3. It contains a region which is solid and another region which is hollow. The entire line can be decomposed into smaller segments where each segment is delineated by short vertical line segments. Thus each line segment is identified by two pairs of intersections (marked as $A_1 - A_2$ and $B_1 - B_2$ for the solid segment, $B_1 - B_2$ and $C_1 - C_2$ for the hollow segment in Figure 3. Considering this observation, the line hierarchy tree shown in Figure 4 is proposed.

2. **Finding intersections:** The intersections defined in Figure 2 are suitable for detection of junctions formed by two lines having approximately the same small thickness. In the distributing frame drawings, the line

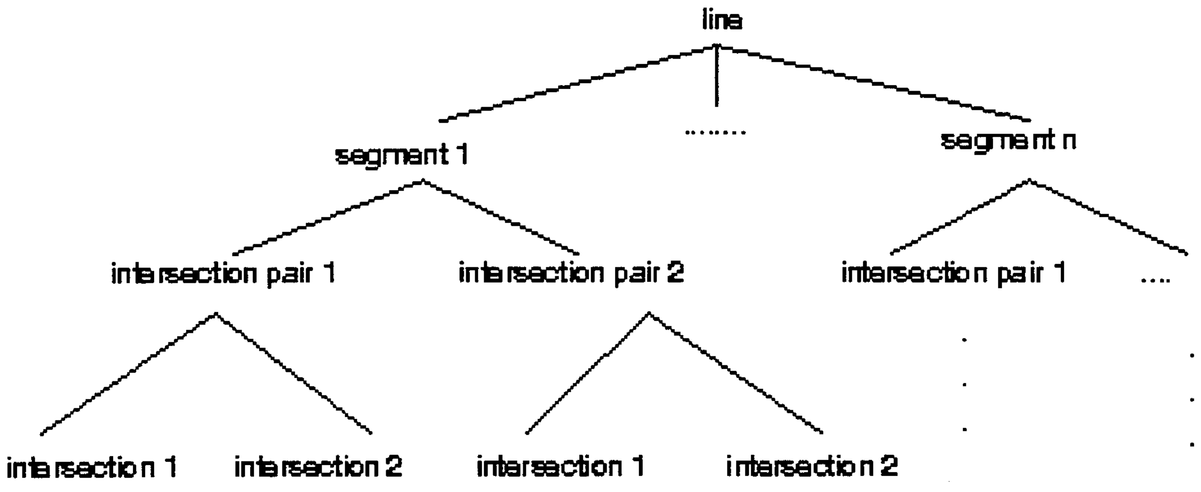


Figure 4: The line hierarchy tree for distributing frame drawings.

thickness are not uniform. Parts of horizontal lines representing segments in use are fairly thick (about 15 pixels compared to thin vertical lines which are about 3 pixels). To account for intersections of thick lines with thin vertical tick marks we modified the algorithm to detect two junctions (one at top and one at bottom). The corresponding intersection hierarchy is shown in Figure 5. The shaded regions in this figure represent line thickness. The intersection at the higher level has priority over those at the lower level. If the intersection at the higher level does not exist, then the intersection at the next lower level is accepted.

3. **Finding intersection pairs:** For each detected intersection, the nearest intersection (within the approximate line width) in the direction perpendicular to the line is found. These two constitute a pair of intersections.

4. **Finding segments:** After detecting intersection pairs, we look for two connected sets of pixels between corresponding points of the two pairs along the line direction. If such sets are detected then a valid segment of a line is noted. The detected segment could be either solid or hollow. The density of the area inside the segment is used to determine the type of the segment. Adjacent segments are then assembled together to form a linked line.

5. **Post–processing—Refinement of line segments:** Some of the detected line segments are false alarms caused by touching text and noise. Occasionally real line segments are missed due to breaks in line pattern resulting in a single line detected as two or more broken segments. The Post–Processing step deletes false segments and connects broken lines.

2.5 Experimental Results

Our algorithms for drawing interpretation have been tested on a number of real images on a SUN SPARCstation 10 machine. Good results have been obtained. Complete processing of a typical equipment drawing of size 10,200 x 13,200 pixels represented as an RLE image takes about 25 seconds. More details on this work may be found in [4–6].

3 Text Extraction from Digital Video

General purpose broadcast video often has textual annotations (captions) in it that supply auxiliary or often essential information. Any digital library system that offers access to digital video will need to extract such text and recognize it. Such annotation can serve as a very good index of the semantic content of the video.

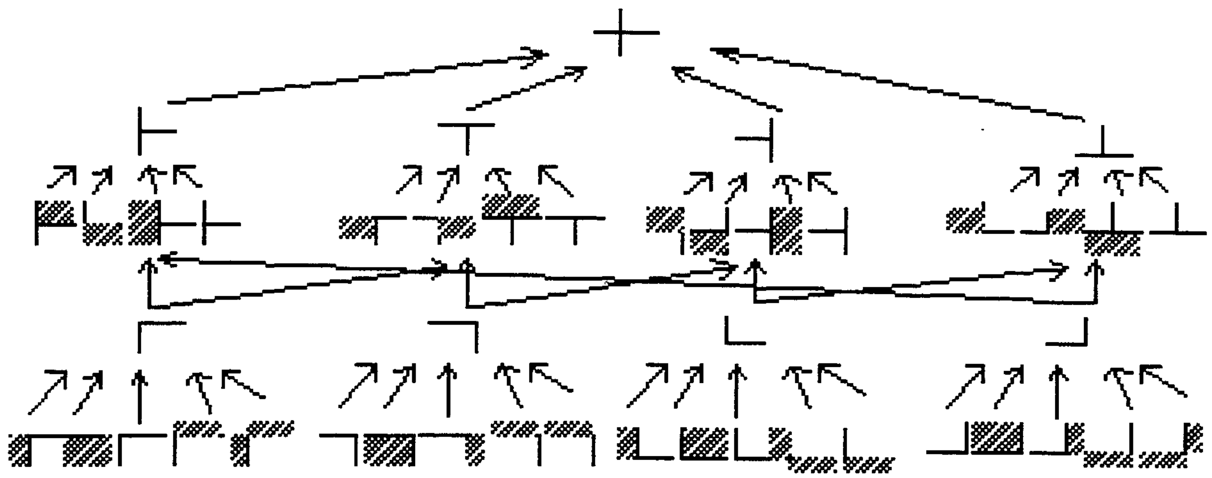


Figure 5: Modified intersection structures.

Extraction of text from general purpose video is therefore an important goal in the drive to build a working digital library. The first task to accomplish this is to detect those regions of the video frame where text is present, and to do this robustly and quickly (ideally, in real time). These regions may then be further processed for text recognition.

Compared to a document analysis system where the scanning process can often be adjusted for best results, the data in video frames is much noisier. There is much variation in the intensity, hue, contrast, opacity, font, and size of the text. Most of the text is horizontally oriented (small skew angles)—we seldom find caption text in any other orientation. There are rarely regular columns and the number of lines of text is usually small. The text can also appear anywhere in the frame. Thus, top–down methods or methods based on segmenting lines (white streams, projection profiles) are unlikely to be of help. Complicating the task is the presence of scrolling captions (as in movie credits or ticker tape news headlines).

We are in the midst of work aimed at robustly detecting text in general–purpose video. The proposed algorithm is being designed to work on color video extracted from TV broadcast. Currently we are investigating the use of morphology in extracting text from video frames. An example video frame and the initial text caption edges extracted from it are shown in Figure 6. Further processing by heuristics will seek to retain only the text regions.

4 Summary

This paper has presented an overview of current document image understanding work at Penn State. The three research efforts discussed were: the development of a graphics recognition and interpretation toolkit, the interpretation of engineering drawings, and the extraction of text from digital video.

References

[1] C. Kohl, J. J. Hunter, and C. L. Loiselle. Towards a Unified IU Environment: Coordination of Existing IU Tools with the IUE. In *5th International Conference on Computer Vision*, pages 2–7, 1995.

[2] L.A. Fletcher and R. Kasturi. A Robust Algorithm for Text String Separation from Mixed Text/graphics Images. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 10(6):910–918, November 1988.

[3] R. Kasturi, S.T. Bow, W. El-Masri, J. Shah, J.R. Gattiker, and U.B. Mokate. A System for Interpretation of Line Drawings. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 12(10):978–992, November 1990.

[4] J. Arias, A. Prasad, R. Kasturi, and A. Chhabra. Interpretation of Telephone

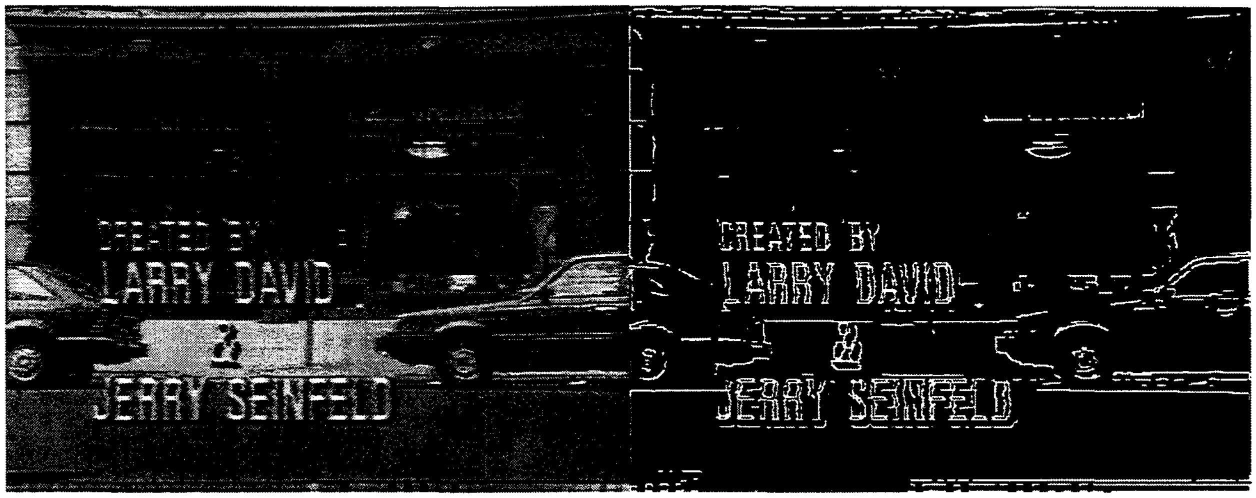


Figure 6: Original and morphological edge detected video frames.

Company Central Office Equipment Drawings. In *12th ICPR*, pages 310–314, 1994.

[5] J. Arias, R. Kasturi, and A. Chhabra. Efficient Techniques for Telephone Company Line Drawing Interpretation. In *3rd ICDAR*, pages 795–798, 1995.

[6] H. Luo, R. Kasturi, J. Arias, and A. Chhabra. Interpretation of Lines in Distribution Frame Drawings. In *accepted for 4th ICDAR*, 1997.

The Knowledge Rendition Server

Edwin R. Addison
KnowledgeLink Corporation
8395 Scarlet Glen Court
Millersville, MD 21108
eaddison@ix.netcom.com

Abstract

KnowledgeLink Corporation is developing a server and browser plug-in technology that provides users with personalized and seamless access to the Internet, Intranets and Online Services. The server technology, also known as the Knowledge Rendition Server, will be an NT based server that selects information sources, generates queries, amalgamates results, distills information into atomic objects, and loads the objects into an object oriented database. The browser plug-in technology, also known as the Knowledge Studio, enables users to establish a personalized profile of a complete electronic publication to be customized in real time. It also renders a personalized view of information taking into account the user's preferences of media type, object or article length, visual vs. Text, raw data vs. Graphs and charts, etc. The system will be completed by strong relationships with publishers of content and online services. The resulting system will be a sophisticated electronic publishing tool that can run on any desktop and give users simple, seamless access to total information. The resulting document will be in http format and will enable users to hyperlink to the originating information sources or online services when desired. The Knowledge Rendition Server combined with the Knowledge Studio will provide a powerful addition to today's methods of browsing and searching the web or private online sources, as well as providing a formidable source of business intelligence information.

A Demonstration will be offered.

The DocBrowse System for Information Retrieval from Document Image Data

Andrew Bruce **Vikram Chalana** **M. Y. Jaisimha** **Thien Nguyen**
MathSoft, Inc.
1700 Westlake Ave. N
Suite 500
Seattle, WA, 98109

Abstract

DocBrowse is a system for information retrieval from document image data. In addition to traditional keyword searching, DocBrowse supports queries based on graphical images. The DocBrowse system consists of a user interface, an object-relational document database, and a variety of document image analysis engines, including OCR, information retrieval, and page segmentation. This paper discusses recent research and development performed on DocBrowse. The system architecture has been revised to improve interoperability, ease of integration, and efficiency. The user interface has been improved to support better query construction and navigation. Some of the new usability features include tools for organizing the query results, automated batch query processing, one-line summary view of query results in addition to thumbnail sketches, filtering to reduce the scope of queries, and highlighting of search terms on retrieved documents. The algorithms for image content querying have been extended to support more types of image queries, including searches which match logos, signatures, words, and pages. Using a multiresolution approach, we have improved the robustness of the searching algorithms towards segmentation errors.

1 Introduction

DocBrowse is a software system for browsing, querying, and analyzing large numbers of document images using both textual and graphical content in the presence of degradations. DocBrowse is oriented towards mixed mode documents consisting of both machine readable text and graphics such as half-tones, logos or handwriting. It incorporates the concept of "query by image example" (QBIE) to support document retrieval based on selected target images. DocBrowse offers a browser and graphical user interface for visually refining and expanding queries.

Several commercial systems have been developed expressly for the problem of management and analysis of document images. These include Capture from Adobe, Visual Recall from Xerox and PageKeeper from Caere among others. These systems feature sophisticated techniques for document analysis problems such as page segmentation and optical character recognition (OCR). They offer state-of-the-art tools and techniques for document analysis and management, and can achieve excellent performance in dealing with clean (low-noise) digital image data. Indexing the text extracted by an OCR and retrieving documents based on textual contents is a standard feature in many of these systems. The performance of these systems, however, is potentially impaired for highly degraded or very noisy images.

Querying document images based on graphical content is available only in a handful of specialized systems. In addition to public domain research systems [6, 10, 9], two commercial systems have recently provided some support for graphical content based querying. Excalibur EFS from Excalibur and the Ultimedia manager from IBM [11] support content-based queries from images based color, texture, position, or shape primitives.

While DocBrowse is designed to handle very general types of documents, our primary research focus is on business letters. DocBrowse supports four types of image queries: logos, handwritten signatures, entire pages, and words not identified by the OCR engine. Bitmap image queries are computationally challenging since they potentially involve matching 100,000's of pixel values. Moreover, in contrast to keyword searching, the retrieval problem is complicated by the fact the matches are seldom precise. The documents may have been subject to degradations introduced by photocopying or FAX transmission. Features such as logos include variations such as half-toning and color. When the documents are scanned using a binary image scanner, both color and half-toning result in significant degradation. Handwritten signatures display additional variability since people rarely sign their name actly the same way each time. Our approach to image retrieval is designed to accommodate all these types of degradations and variations.

In this paper, we give an overview of DocBrowse, focusing on the latest research and development efforts. These efforts span three broad areas of the system: overall software architecture design, usability for query construction and navigation, and algorithms for image based querying. In the software design of DocBrowse, we have defined a new application called "DocLoad" which handles all of the document processing related tasks (e.g., OCR and page segmentation), thereby improving the modularity of the system. Some of the new usability features include tools for organizing the query results, automated batch query processing, one-line summary view of query results in addition to thumbnail sketches, filtering to reduce the scope of queries, and highlighting of search terms on retrieved documents. Finally, we have extended the algorithms for image based querying to apply to pages and words. We have also improved the robustness of the algorithms towards page segmentation errors.

Section 2 presents an overview of the DocBrowse system and its software architecture. The user interface for query construction and navigation is discussed in Section 3. In Section 4, we describe the new algorithms used for image content querying. The paper concludes with a discussion of future research directions in section 5.

2 System Overview

DocBrowse consists of three main components: 1) A browser and graphical user interface (GUI) for visual querying and sifting through a large digital document image database, 2) An object-relational database management system (ORDBMS) for storing, accessing, and processing the data, and 3) "DocLoad," an application which processes the raw document images through specialized document analysis software (OCR, page segmentation, and information retrieval) and inserts this information into the database. The overall system architecture for DocBrowse is displayed in Figure 1.

2.1 User Interface

The heart of DocBrowse is the visual browser and graphical user interface. In addition to standard queries based on textual and document tag information, support is provided for "query by image example" (QBIE). QBIE enables users to retrieve documents based on regions of the image that would not ordinarily be readable by an OCR, such as logos, handwriting (such as signatures), halftoned images, line drawings, and highly degraded words. Users submit queries from the GUI without having to directly manipulate SQL code. Tools are provided for visually browsing the results of a query. A history mechanism helps users to navigate through a succession of queries, with support for iterative query refinement and expansion. The DocBrowse interface is discussed in detail in section 3.

2.2 Database and Schema

The document data is stored in an object-relational database management system (ORDBMS). The ORDBMS provides for efficient handling of large binary image objects, built-in processing for text retrieval and SQL, and modular extensibility according to the object-oriented model. The schema is based on several existing standards for document description and layout including the ISO Office Document Architecture standard [1], SGML, DAFS from RAF Technology [8], and RTF [7].

A *document* is defined as a collection of *document pages.* Document pages are in turn decomposed into *zones.* The decomposition of pages into zones operates at multiple granularities. At the coarsest level, the page can be decomposed into *header*, *footer* and *live matter* zones. At the finest level of granularity each character on the page can be considered a zone. At an intermediate level of granularity each paragraph or body of text which is distinctly separated from adjoining bodies of text or figures can be referred to as a zone. It is this level of granularity that we adopt for DocBrowse. Zones can be of two types - text and non-text or graphics zones. A graphics zone contains information such as figures, line drawings, half-tones or bitmaps (such as logos). Each document, document page and zone can have associated with it one or more tags in the form of attribute-value pairs. Specifically, these tags could include in the case of a document page, the scanned bitmap of a page, the type of document, scan resolution, and OCR text. In the case of a zone, the types include a processed bitmap of the zone or features extracted from the zone which could be used for zone classification or classifier construction.

To improve the computational efficiency of searching, the schema defines search matrices which are pre-computed rankings of the documents for possible search terms. Currently, this is only done for the image based queries since the keyword queries are performed using off-the-shelf database tools.

2.3 DocLoad

DocLoad converts the raw image data into query terms and search matrices, inserts the query terms and other information into the database, and performs user-defined batch queries. In the first stage, the raw document images are processed through specialized document analysis software, including OCR

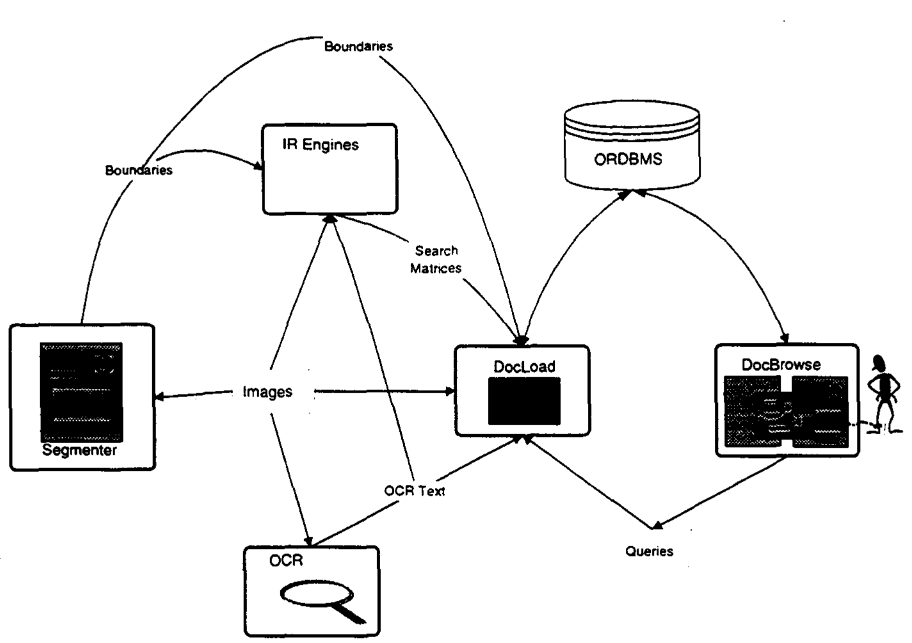


Figure 1: Software architecture for the DocBrowse system. DocBrowse consists of a browser and graphical user interface (GUI), an object-relational database management system (ORDBMS) for storing, accessing, and processing the data, and "DocLoad," an application which processes the raw document images through specialized document analysis software (OCR, page segmentation, and information retrieval) and inserts this information into the database.

and page segmentation. The output of the OCR and page segmentation are then passed to a variety of information retrieval engines. The information retrieval engines construct feature vectors for searching on images and build search matrices designed for fast on-line retrieval of text and images. In the second stage of processing, DocLoad inserts the output of the OCR, page segmentation, and information retrieval engines into the database. It does this by constructing tables for text, tags, images, search matrices, etc.. In the final stage of processing, DocLoad performs user-defined batch queries and saves the results of these queries in the user's folders (see section 3.4).

3 Query Construction and Navigation

DocBrowse offers a graphical user interface (GUI) and visual browser for extracting information from a large collection of possibly degraded digital document image data. Figure 2 shows the primary components of the DocBrowse GUI display. The GUI supports a visual programming interface and textual query language interface to compose queries, a visual browser to scan the results of queries as thumbnail sketches or one-line summaries, a document viewer which highlights search terms and supports query refinement through context sensitive mouse selections, and tools for organizing the results of queries. These are discussed below.

3.1 Types of Queries

DocBrowse supports three basic types of query terms: text/keywords, tags, and bitmap images. In a *text query*, the user queries on text keywords that are present in the OCR output text of the document pages in the database. In a *tag query*, the user queries on attribute-value pairs. For example, a tag query can take the form "Retrieve all documents from the Trash folder." In a *query by image example* (QBIE), the user selects a graphical zone on a page and searches for all other images with similar graphical regions. In a QBIE query, the searching is done using a feature as described in section 4.

An inheritance hierarchy for the specific search terms defined in DocBrowse is shown in figures 3. Text can be either one or more text keywords. In the case of groups of keywords, either proximity or adjacency constraints can be enforced. Tags can either static, having a persistent value in the database, or dynamic, evaluated at query time. Folders are a special type of tag which is used for organizing the documents by category (see section 3.4). Four types of image terms (feature vectors) are defined in DocBrowse: logos, handwritten signatures, words not recognized by the OCR, and entire pages.

Query terms can be combined in a boolean fashion using the operators `and`, `or`, or `not`. For instance, a text query on `Statistical Sciences`

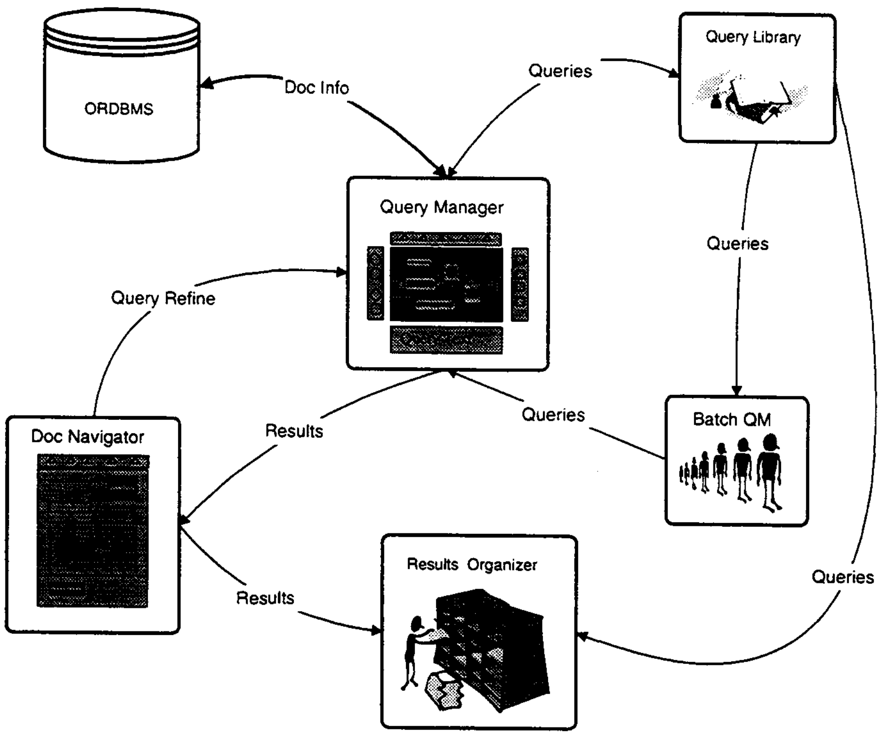


Figure 2: Components of the graphical user interface (GUI) for the DocBrowse system. The GUI supports a visual programming interface and textual query language interface to compose queries; a persistent library of user and system defined queries; a batch query manager for automated query processing; a document navigator for browsing the results of queries as thumbnail sketches or one-line summaries and viewing a document with the search terms highlighted; and tools for organizing the results of queries.

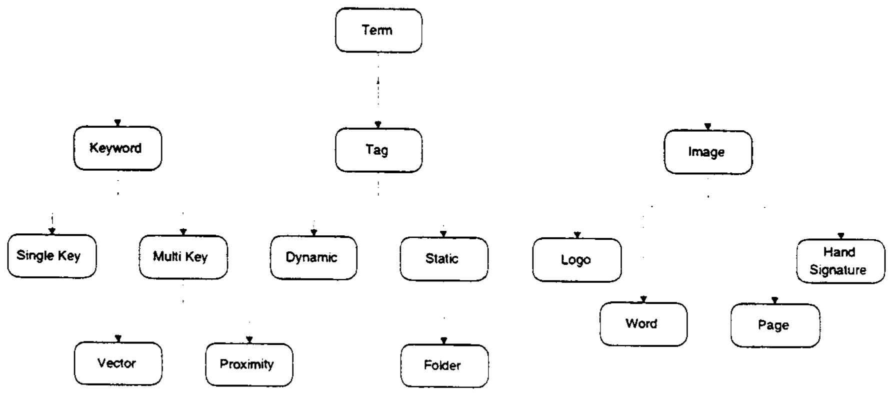


Figure 3: DocBrowse supports three basic types of query terms: text/keywords, tags, and bitmap images. The above diagram shows the inheritance hierarchy for these search terms.

or `University Washington` would return all documents which contain either `Statistical Sciences` or `University Washington` in the OCR text. In this case the and operator is implicit and assumed, so the actual query would be something like (Statistical and Sciences) or (University and Washington).

DocBrowse also supports the combination of imprecise queries. Imprecise queries are an important component of DocBrowse since image queries seldom result in an exact match: the bitmap representations typically involve 10,000 or more pixel values. In DocBrowse, image queries are based a similarity score between zero and one measuring the "closeness" of the image to a target image (see section 4). Imprecise queries are also useful for keyword searching: while keyword searches often result in exact matches, most OCR engines support imprecise matching In a query with a single term, the results can be simply ranked by the score of the match. To support querying on multiple terms with imprecise matching, we extend the definition of and, or, or not operators. Let x_1, x_2, ..., x_p be numeric scores between zero and one giving the strength of a match to the two query terms where zero is a non-match and one is a perfect match. In the current version, the operators are defined by

$$\begin{aligned}
\texttt{and}(x_1, x_2, \ldots, x_p) &= \texttt{min}(x_1, x_2, \ldots, x_p) \\
\texttt{or}(x_1, x_2, \ldots, x_p) &= \texttt{max}(x_1, x_2, \ldots, x_p) \\
\texttt{not}\ x_i &= 1 - x_i
\end{aligned}$$

3.2 Query Manager

Users can compose queries in DocBrowse using the query manager. The query manager supports both a visual iconic programming interface and a command line interface. The iconic programming and command line languages are equivalent, and translate into identical queries. We show a screen snapshot of the interface in Figure 4. The top portion of the interface is a query composition canvas. Below that is the command line pane. On the edges of the query manager window are toolbars. The toolbar buttons allow the user to add query terms — keywords, tags, and images — and operators to the glyph palette. Image example glyphs can also be added by dragging segmented graphical entities such as logos from a document page displayed in the DocBrowse document navigator (see section 3.3).

Queries are composed by connecting term and operator glyphs with the mouse. When the user has finished generating a query using the visual programming interface, the corresponding text query can be examined in the lower half of the interface window. This text query can be edited and the changes are automatically propagated to the visual program graph in the upper window.

The toolbar buttons also support a number of specialized operations, including

- Stepping through the query history for the current session.
- Filtering out `Trash` or other selected folders from the query.
- Eliminating duplicate or near-duplicate documents from the retrieved documents.

Another feature supported in DocBrowse, not shown in figure 4, is the query library manager. The library manager allows users to store and re-use queries and to access system query libraries (e.g., a catalog of logos). The libraries are stored in the database and are persistent objects.

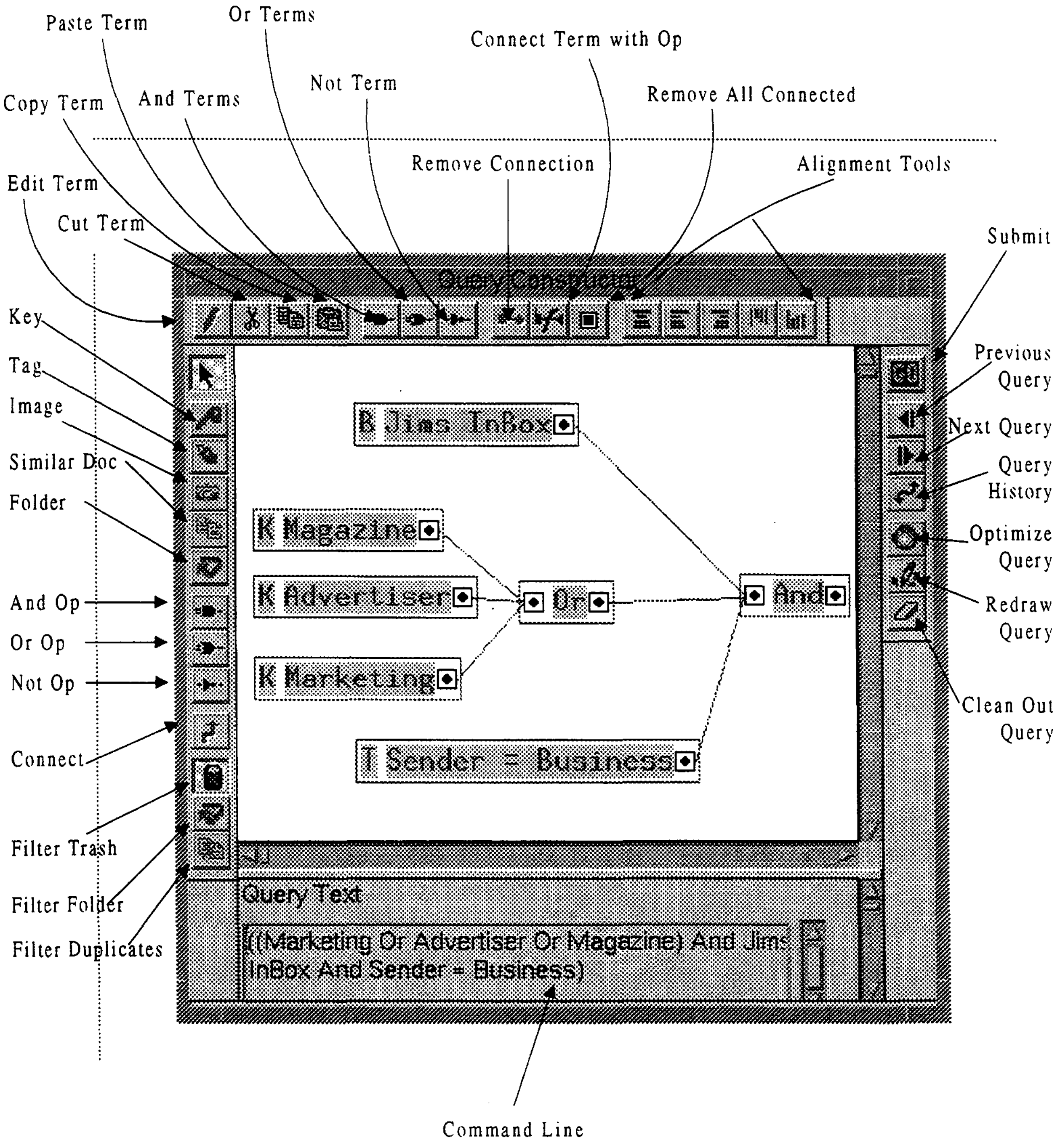
3.3 Document Navigator and Query Refinement

The results of queries are displayed by the DocBrowse document navigator. The document navigator can display either thumbnail sketches of the retrieved document images or as one-line text summaries. If displayed as thumbnail sketches, DocBrowse will display only $N \approx 16$ thumbnails at a time to ensure the images are legible (N is a user specified default). The user can scroll through pages of thumbnails or one-line summaries using a toolbar.

Users can select documents to be viewed individually. Figure 5 displays a typical view of a retrieved document in the document navigator. In addition to displaying the bitmap image, DocBrowse displays a variety of other information:

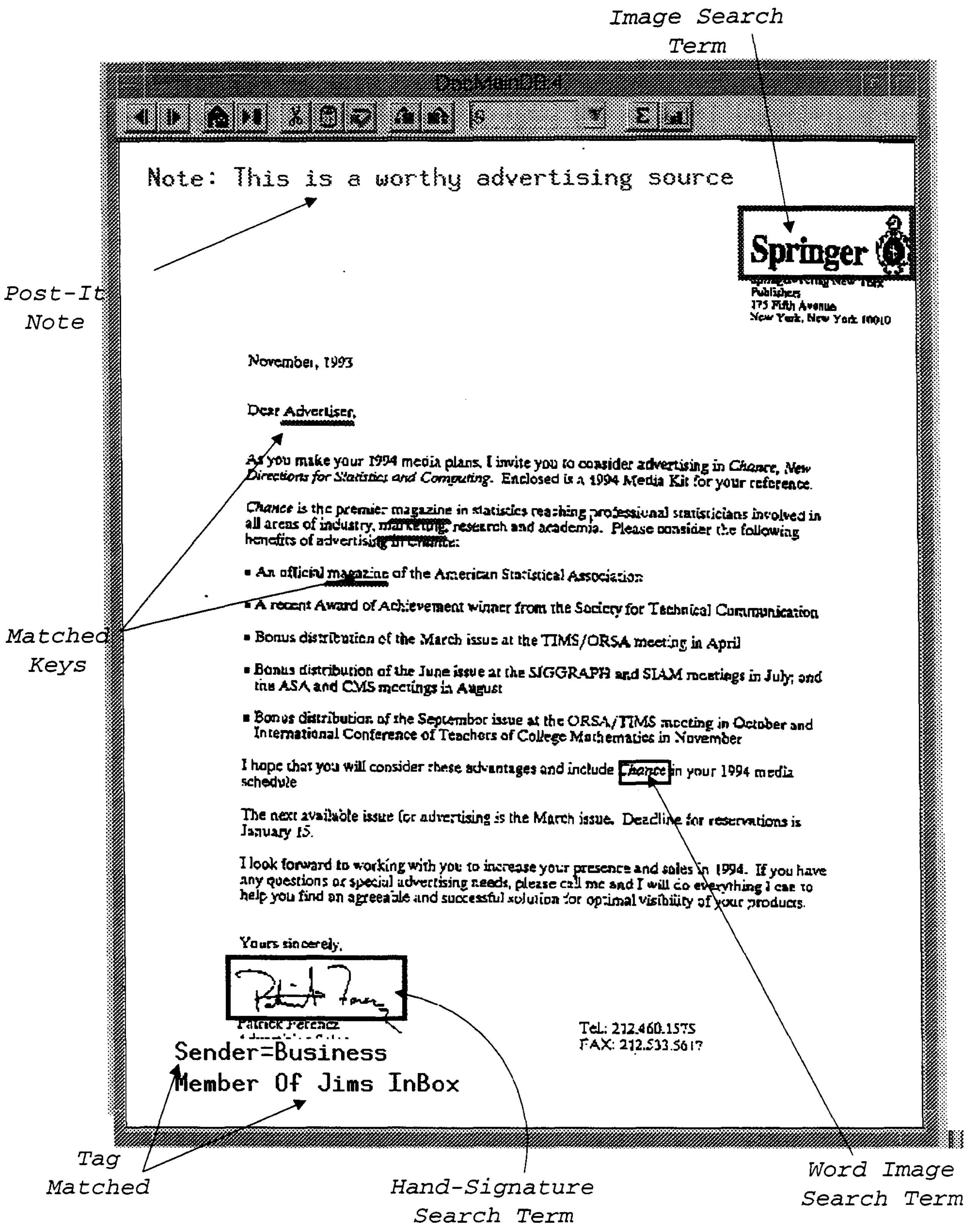
- The search terms which were matched in this query are highlighted.
- Tags which were matched are displayed in the lower left-hand corner.
- Post-it notes, input by the user, are displayed at a user-specified location.
- Searchable graphics zones, which include logos, handwritten signatures, and words not recognized by the OCR, are identified by bounding boxes.

An important feature supported by DocBrowse is the idea of applying relevance feedback from a set of documents retrieved in response to a query. The user can select either keywords from the text or any of the identified searchable graphic zones as being relevant and add these to the existing query. In certain cases, additional relevant documents will be retrieved that were not returned by the initial query.



((Marketing Or Advertiser Or Magazine) And Jims InBox And Sender = Business)

Figure 4: Annotated screen snapshot of the query manager. The query manager supports both a visual iconic programming interface and a command line interface. On the edges of the query manager window are toolbars. The toolbar buttons allow the user to add query terms — keywords, tags, and images — and operators to the glyph palette. The toolbar buttons also support a number of specialized operations, such as history and duplicate document identification.



Note: This is a worthy advertising source

Springer

Springer-Verlag New York
Publishers
175 Fifth Avenue
New York, New York 10010

November, 1993

Dear Advertiser,

As you make your 1994 media plans, I invite you to consider advertising in *Chance, New Directions for Statistics and Computing*. Enclosed is a 1994 Media Kit for your reference.

Chance is the premier magazine in statistics reaching professional statisticians involved in all areas of industry, marketing, research and academia. Please consider the following benefits of advertising in *Chance*:

- An official magazine of the American Statistical Association
- A recent Award of Achievement winner from the Society for Technical Communication
- Bonus distribution of the March issue at the TIMS/ORSA meeting in April
- Bonus distribution of the June issue at the SIGGRAPH and SIAM meetings in July; and the ASA and CMS meetings in August
- Bonus distribution of the September issue at the ORSA/TIMS meeting in October and International Conference of Teachers of College Mathematics in November

I hope that you will consider these advantages and include *Chance* in your 1994 media schedule

The next available issue for advertising is the March issue. Deadline for reservations is January 15.

I look forward to working with you to increase your presence and sales in 1994. If you have any questions or special advertising needs, please call me and I will do everything I can to help you find an agreeable and successful solution for optimal visibility of your products.

Yours sincerely,

[illegible]

Patrick Ferencz
Advertising Sales

Tel: 212.460.1575
FAX: 212.533.5617

Sender=Business
Member Of Jims InBox

Figure 5: A retrieved document displayed by the DocBrowse navigator. In addition to displaying the bitmap image, DocBrowse displays a variety of other information, including highlighting of the search terms, display of the tags, post-it notes, and searchable graphics zones which include logos, handwritten signatures, and words not recognized by the OCR.

The document navigator offers numerous other usability features. Users can prune the results, either from the summaries (text or thumbnails) or from graphical plots of the similarity scores. The user can view the ASCII text corresponding to this image output from the OCR, although typically this text is less readable than the bitmap image. DocBrowse also provides cut and paste facilities that allow users to incorporate the query results in a report or other document.

3.4 Query Result Organizer

The DocBrowse query result organizer allows users to manage the results of queries using an interface similar to electronic mail handling systems. Results from a query, either selected documents or all retrieved documents, can be saved into user-defined folders. Like mail systems, the query result organizer allows users to update and add new folders. In addition, the system defines the Trash folder which can be accessed through toolbar buttons on the query manager (see section 3.2). The user can browse folders using the DocBrowse document navigator. At a system level, folders are just a special type of tag. Hence, folders can be included as a part of query like any other term.

In addition to organizing the results from *ad hoc* queries, folders can be used as a form of a semi-automated database clustering tool. Users can associate a specific query with a folder. Using this query, DocLoad will automatically populate the folder at regularly specified times (e.g., every night). By selectively including and excluding folders, users can effectively partition databases. By default, the system defines an Inbox folder which contains all documents in a database which have not been assigned to another folder.

4 Algorithms for Image Content Querying

As described previously, the DocBrowse system supports a variety of methods for searching/browsing through a document image database based on image content. The four key content-based queries supported by DocBrowse are: logo queries, handwritten signatures, similar document queries, and word image queries.

The basic algorithm (see figure 6) involved in each of these content-based queries is (1) extract and store distortion invariant features for each query term supported (e.g., for logos, for whole images, or for word images, in our case), (2) compute the similarity (or distance) between the target feature vector and the feature vectors for the documents in the database, and (3) select the documents in the database which best match the query term (the target).

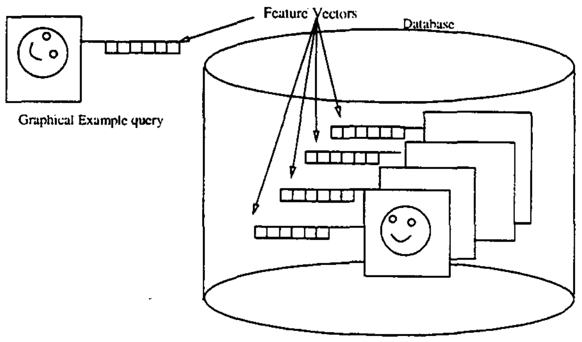


Figure 6: An illustration of the approach taken towards query by image example (QBIE) by DocBrowse. One or more compact feature vectors are precomputed and stored associated with each document in DocBrowse. When the user submits a query by graphical example, the feature vector associated with the graphical example is compared to the feature vectors of the graphic regions in the database. Documents that are close to the query are returned by the system.

We have previously described an algorithm for logo matching where the features used for matching are the wavelet transforms of the x- and y-projections of the logo images [4, 5]. We showed that the choice of this feature vector was quite robust to noise in the document images. This choice of the feature vector also led to very high computational efficiency for matching. In this paper, we explore two extensions to the QBIE algorithm: application to the problem of similar document identification and improved robustness towards segmentation errors.

4.1 Similar Document Searching

The problem of searching for similar documents in a database of document images can be divided into two sub-problems, i.e., identification of duplicate documents and the identification of near-duplicates or similar documents. The identification of duplicate documents is important to remove redundant information if the database. The identification of similar documents is useful for searching and pre-clustering of a database. Both duplicate document identification and similar document identification are supported by the query manager with toolbar buttons: see section 3.2.

Even though the above two problems can be solved using the same feature extraction and matching algorithm, the two problems differ in the magnitudes of the allowable errors. Given a target (query) document x_0 and a document x, two types of errors may be defined:

- Type I Error: $p(x \neq x_0 | x = x_0)$
- Type II Error: $p(x = x_0 | x \neq x_0)$.

The Type I error (false reject) is the probability of classifying the document x as different from x_0, when in reality the two documents are duplicate. The Type II error (false accept) is the probability of classifying the document as duplicate, when in reality they are different. For the automatic identification of duplicate documents, the Type II error needs to very low because document should not be erroneously marked as duplicate and eventually removed from the database. For similar document querying, the Type I error needs to be very low because the user query should not miss documents which are duplicate in reality.

We tested two different algorithms for similar document searching. The first algorithm was based on features extracted from the OCR text for a document, and the second algorithm was based on features extracted from the document image itself.

The OCR-based method for similar document searching first finds words in the OCR text. These words are pruned to construct a list of stemwords representing the document. Stemwords are formed by first removing all stopwords, e.g., prepositions, articles, conjunctions, from the list of words. Next, for the remaining words in the word list, word roots are identified by discarding suffixes for plurals, verbs, etc. The feature vector used for matching is the frequency of occurrence of stemwords. The similarity between two documents is defined as the normalized dot product of the two feature vectors.

The image-based method algorithm used for similar document searching was similar to our logo matching algorithm. We first compute the x- and y-projections of the entire document. Next, we take the wavelet transform of the x- and y- projections using the Daubechies d8 wavelet at nine levels [3]. The low-pass wavelet transform coefficients at scale 2^9 were used as the feature vector for matching, which corresponds to 15 coefficients. Using more coefficients in the feature vector by adding other wavelet transform scales did not significantly effect the performance of the matching. Since the entire document is approximately 3300 × 2400, this feature vector is very compact and is quite insensitive to rotation, translation, or noise in the document. Normalized cross-correlation was used to measure the similarity between two feature vectors.

We tested the performance of this algorithm on a database of 171 documents. These 171 document database had been composed of three degradations of 57 different documents. The three degradations corresponded to photocopying and faxing, photocopying twice and faxing, and photocopying and faxing twice. Thus, we knew that in our database, each document had two duplicate documents. Using each document, in turn, as the target document, we computed the similarity coefficients against all other documents in the database.

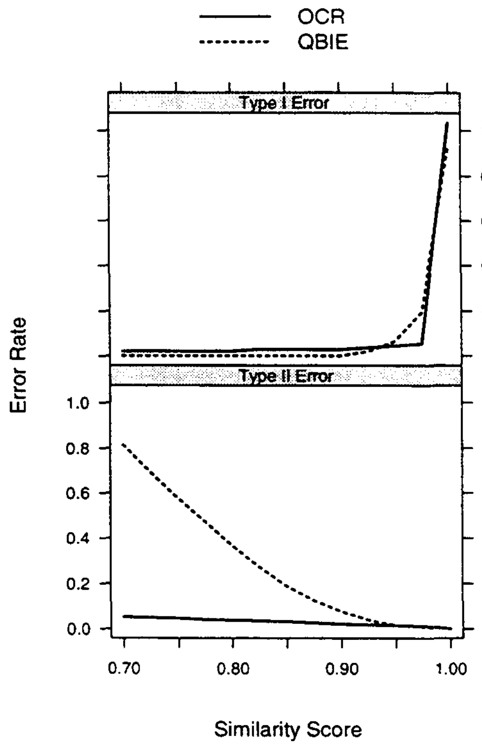


Figure 7: The Type I and Type II error rates for finding similar/near-duplicate documents based on matching the OCR text (labeled OCR) and the bitmap image of the entire page (label QBIE). The QBIE algorithm is good for retrieving similar documents from a database (similar document query mode) because of the very low Type I error. The OCR-based algorithm is good for duplicate document identification because of the low Type II error.

Figure 7 shows the average Type I and Type II errors for the OCR-based method and the image-based method against the different cut-off points for the similarity score. As we can see in the figure, the image-based method (QBIE) has a lower Type I error (error = 0 for score < 0.9) that the OCR-based method; however, the OCR-based method had a substantially lower Type II error for score < 0.95. The OCR-method failed to find the duplicate documents in cases where the document had much handwritten text or very little printed text. Both methods had difficulty in discriminating between documents which were very similar, but not actually duplicate (both methods identified these documents as duplicate).

In conclusion, the QBIE algorithm is good for retrieving similar documents from a database (similar document query mode) because of the very low Type I error. The OCR-based algorithm is good for duplicate document identification because of the low Type II error. For acceptable levels of Type I error, the OCR-based algorithm still has Type II error of about .02 in this experiment. For some applications, even this small of an error may be considered to be unacceptably high for duplicate document identification where the duplicate documents would be tagged for removal from the database.

4.2 Robustness Towards Segmentation Errors

The QBIE algorithm for logo matching in DocBrowse, described in [4, 5], was shown to perform very well under a wide-variety of degradation conditions. However, the the algorithm assumed the logos to be segmented accurately. Preliminary experiments indicate that the performance of the algorithm degrades significantly when the logos are subject to significant segmentation errors. This is because one of the key components of the algorithm was that before computing the feature vector for a logo, the logo image was normalized for rotation and for scale. In on-going research, we apply a multiresolution registration procedure to improve the robustness of the QBIE.

When the logos are segmented automatically using a page segmentation algorithm, as in [2], segmentation errors can occur. Two kinds of segmentation errors occur - (1) only a part of the original logo is segmented, or (2) the segmenter identifies another part of the document as a logo. The second type of error can be corrected only with a better segmenter; however, the first type of error can be corrected by using a better matching algorithm. Therefore, we redesigned our logo matching algorithm to take into the account the fact that the logo may not be completely segmented and partial matches may need to be considered.

The new robust algorithm also uses the the wavelet transform of the x- and y-projections of the logo image as features to match. Unlike our previous algorithm, however, this algorithm does not normalize the logo with the scale or rotation prior to computing the feature vector. The normalization with respect to rotation is not needed because the segmenter that we are using corrects the skew in the document prior to segmentation. The normalization with respect to scale is not helpful to this matching algorithm because we are not assuming the logos to be correctly segmented.

The matching procedure in our robust algorithm is also different from our old logo matching procedure. In the old algorithm, the similarity measure is the cross-correlation between the wavelet coefficients at the coarsest level. In the new algorithm, we first register the feature vectors before we compute the correlation coefficient. This registration is with respect to translation only. Registration with respect to scale will be deferred to future work. We have designed a multiresolution registration algorithm to register the x- and y-projections of logos.

This registration procedure uses the multiresolution approximation (MRA) of signals. Using the wavelet transform coefficients, an MRA of the signal can be constructed where the number of coefficients representing a signal increases with increasing resolution. The left column of figure 8 shows the multiresolution approximation at 5 levels of the y-projection for a sample logo. At the coarsest resolution, the projection (of length 1024) can be approximated by only 32 coefficients. At the next finer resolution, 64 coefficients are needed and so on. The multiresolution registration procedure hinges on the fact that the search for the appropriate registration parameters can be greatly sped-up by starting the search at the coarsest resolution. The coarse level registration parameters can be found quickly because of the small number of coefficients to deal with. These parameters are then carried over to the finer level of detail where the search-space is now substantially reduced. Figure 8 shows this registration procedure for an example where the a complete "Marriott" logo is matched to the one that has been only partially segmented.

We compared our new robust matching procedure to the our original logo matching procedure. We used a database of 16 logos where 3 logos were "Marriott" logos which had been partially segmented by the automatic segmenter. A fully segmented Marriott logo was considered as the query (or target) logo and similarity measures were computed between this logo and each logo in the database. Figure 9 plots the similarity measures for the two matching procedures. The scores for the target Marriott logo and the incorrectly segmented Marriott logos in the database are marked by an "X". The scores for the other logos are marked by a circle. In contrast to the original algorithm, the robust algorithm fully distinguishes between the Marriot logos and the other logos.

5 Discussion

In the immediate future, development on DocBrowse will continue to concentrate on refinement of the interface for query navigation and improvement of the

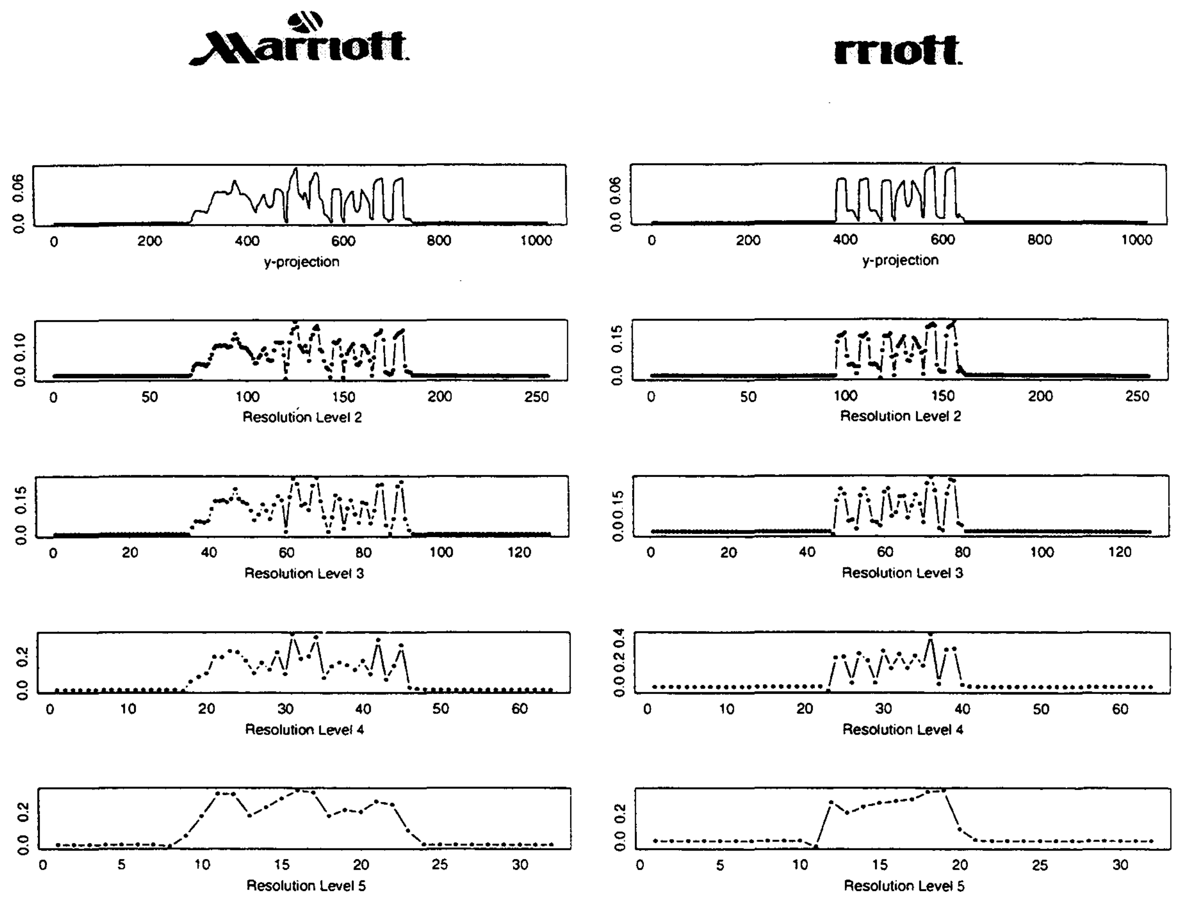


Figure 8: The multiresolution registration procedure for projections of two logos. The logo on the left has been correctly segmented whereas the one on the right has been only partially segmented. The registration procedure starts at the coarsest level (shown on the bottom row) where the search for the best registration parameters needs to be carried out only for five pixels. The best match at this level is found by right-shifting the partially segmented logo by 3 pixels. At the next finer level, a shift of 6 pixels is used as initial registration and two pixels around this are searched for an even better registration. By this multiresolution search the best match at the finest resolution is found by right-shifting the partially segmented logo by 99 pixels. A brute force search requires $O(N)$ operations while the multiresolution search requires $O(logN)$ operations.

system architecture for interoperability and ease of integration. In addition, we are investigating the following research issues:

- We will continue to improve the performance of the query by image example algorithms. Following the investigations described in section 4, we plan to extend the robust algorithm to handle scale invariant searches. We also plan to investigate signature matching using deformable templates and explore specialized methods for word image spotting.

- The result from a query can, and often will, retrieve many documents. It is crucial to rank the retrieved documents according to relevance. When a query is composed of multiple targets (e.g., multiple keywords, tags, and logos), or when multiple IR engines are being used to perform the query, we need a way to combine the results for a composite relevance score. This score will depend on a number of factors, such as precision of a keyword or image match or predicted accuracy of the OCR or IR engine. We currently use very simple rules to combine multiple imprecise matches to query terms (see section 3.1). We will investigate different ways to quantify, combine, and visualize relevance scores from multiple sources.

- Our research has mostly focused on "micro" views of a database focused on a specific query. An important problem is to obtain "macro" views of a database to help determine what is in the database from a global point of view. DocBrowse provides limited tools for semi-automated clustering of a database (see section 3.4). Following other related work, we will

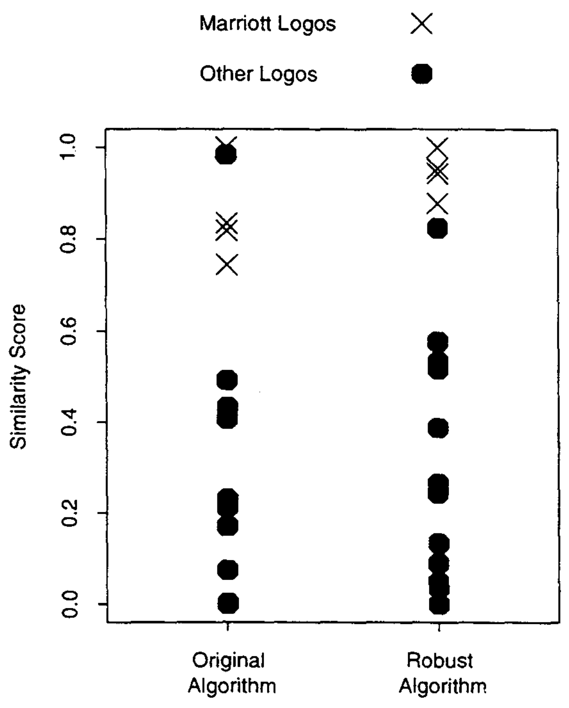


Figure 9: Similarity measures for the original matching algorithm and the robust matching algorithm for a database of 16 logos. The scores for the target Marriott logo and the incorrectly segmented Marriott logos in the database are marked by an "X". The scores for the other logos are marked by a circle. In contrast to the original algorithm, the robust algorithm fully distinguishes between the Marriot logos and the other logos.

explore new tools for more automated database clustering and for visualizing the content of a database.

Acknowledgments

Emmanuel Arbogast (MathSoft, Inc.) and Jeremy Worley (Khoral Research Inc.) contributed substantially to the early design and development of DocBrowse. We also thank Giovanni Marchissio (MathSoft, Inc.) for many useful discussions regarding information retrieval.

References

[1] W. Appelt. *Document Architecture in Open Systems: the ODA standard.* Springer-Verlag, 1991.

[2] R.J. Bennett. Heuristic-based object extraction and recognition. In *Proc. 1995 Symposium on Document Image Understanding Technology*, pages 257–262, 1995.

[3] I. Daubechies. *Ten lectures on wavelets.* Society for industrial and applied mathematics. Philadelphia, PA, 1992.

[4] M. Y. Jaisimha. *Compound Document Retrieval in Noisy Environments.* PhD thesis, University of Washington, Seattle, U.S.A., 1996.

[5] M. Y. Jaisimha. Wavelet features for similarity based logo retrieval. In *Proceedings of SPIE Document Recognition III, San Jose, CA*, pages 350–61, 1996.

[6] O. Lorenz and Gladys Monagan. A retrieval system for graphical documents. In *Proceedings of UNLV Symposium on Document Analysis and Information Retrieval*, pages 291–300, 1995.

[7] Microsoft Corp. *Rich Text Format Specification*, version 1.2 edition.

[8] RAF Technology, Inc., Redmond, WA. *Document Attribute Format Specification*, 1995.

[9] H. Samet and A. Soffer. Marco: Map retrieval by content. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18:783–798, 1996.

[10] R. K. Srihari. Combining text and image information in content-based retrieval. In *Proceedings of Symposium on Document Image Understanding Technology, Bowie MD*, page 14, Oct. 1995.

[11] H. Treat, E. Ort, J. Ho, M. Vo., J.S. Jang., L. Hall, F. Tung, and D. Petkovic. Searching images using ultimedia manager. *Information Services & Use*, 16:15–24, 1996.

Locating and Recognizing Text in WWW Images

Daniel Lopresti dpl@research.panasonic.com

Jiangying Zhou jz@research.panasonic.com

Matsushita Information Technology Laboratory
Panasonic Technologies, Inc.
Two Research Way
Princeton, NJ 08540

Abstract

The explosive growth of the World Wide Web has resulted in a distributed database consisting of millions of documents. While existing search engines index a page based on the text easily extracted from its HTML encoding, an increasing amount of the information on the Web is embedded in in-line images. This situation presents a new and exciting challenge for the field of document analysis, as WWW image text is typically rendered in color and at very low spatial resolutions.

In this paper, we describe the current status of our work in this area. For the problem of locating text regions, we have developed a procedure based on clustering in color space followed by a connected-components analysis. For recognition, we have begun exploring techniques using a "fuzzy" n-tuple classifier that seem promising. We conclude with some preliminary experimental results and a discussion of topics for further research.

Keywords: document analysis, information retrieval, optical character recognition, World Wide Web images, GIF, JPEG.

1 Introduction

With the explosive growth of the World Wide Web (WWW), a great number of documents are being made accessible electronically. The issue of developing efficient methods for searching, browsing, and retrieving these documents is becoming increasing important. Much progress has already been made in these areas. For example, there are various Web search engines currently available for accessing text information from the Web pages. Various issues associated with automated document conversion from paper to electronic have been addressed by researchers from both the information retrieval and document analysis communities [10, 11].

Existing search engines currently only index the raw ASCII text that is easily extracted by parsing the HTML. On the other hand, an increasing amount of the information on the Web is embedded in in-line images. Current WWW tools generally do not provide the ability to assess such information. In a recent investigation, we examined the percentage of text that appears in image format in Web pages. We found that the majority of Web pages contain at least some fraction of their text in image format. In some cases, this can be as high as 44%. Moreover, a significant fraction of the image text often does not appear elsewhere (to a maximum of 100% in some examples) [9]. As an example, we note that the text string "Panasonic Online", featured prominently in the snapshot on the left side of Figure 1, is completely absent from the extracted text on the right side of the figure.

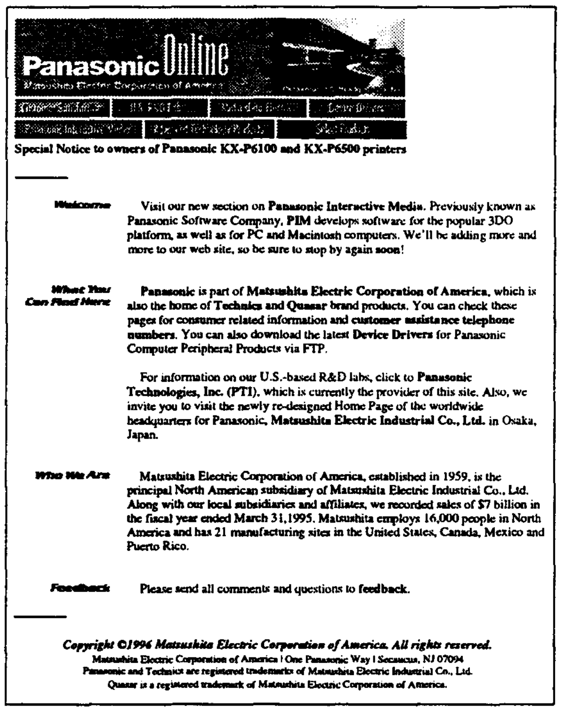
On the other hand, how information is actually encoded may not be obvious or even important from a user's perspective. To a user, the phrase "Panasonic Online" is simply a string of text on a Web page. A user may well want to search it through traditional text query. Consequently, there is a need to develop techniques for recovering the text in Web image data.

There are several significant differences between the Web image processing problem and its traditional counterpart. Electronic bitmaps on the WWW are typically created at a low resolution (72 dpi) intended for screen display. They often make up for this in liberal use of color. This presents a new and exciting challenge for the field of document analysis.

In this paper, we describe the current status of our work in this area. The remainder of the paper is organized as follows. In Section 2, we discuss various formats used for Web images and their impact on the Web image processing problem. Section 3 discusses a procedure for locating text regions from Web images. Section 4 describes a text recognition system based on a fuzzy n-tuple technique. Finally, we give a short conclusion in Section 5.

2 WWW Image Formats

In principle, any format could be used to distribute images over the Web. All that is required is writing an appropriate browser "plug-in" to support it. However, two formats in particular have emerged as de facto standards: GIF and JPEG. Each of these is applicable in its own specific domain. While it is



Special Notice to owners of Panasonic KX-P6100 and KX-P6500 printers

Welcome Visit our new section on **Panasonic Interactive Media**. Previously known as Panasonic Software Company, **PIM** develops software for the popular 3DO platform, as well as for PC and Macintosh computers. We'll be adding more and more to our web site, so be sure to stop by again soon!

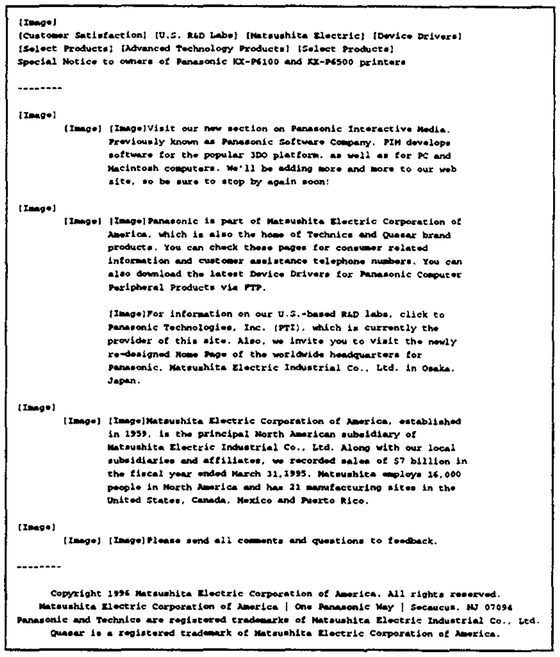
What You Can Find Here **Panasonic** is part of **Matsushita Electric Corporation of America**, which is also the home of **Technics** and **Quasar** brand products. You can check these pages for consumer related information and **customer assistance telephone numbers**. You can also download the latest **Device Drivers** for Panasonic Computer Peripheral Products via FTP.

For information on our U.S.-based R&D labs, click to **Panasonic Technologies, Inc. (PTI)**, which is currently the provider of this site. Also, we invite you to visit the newly re-designed Home Page of the worldwide headquarters for Panasonic, **Matsushita Electric Industrial Co., Ltd.** in Osaka, Japan.

Who We Are Matsushita Electric Corporation of America, established in 1959, is the principal North American subsidiary of Matsushita Electric Industrial Co., Ltd. Along with our local subsidiaries and affiliates, we recorded sales of $7 billion in the fiscal year ended March 31,1995. Matsushita employs 16,000 people in North America and has 21 manufacturing sites in the United States, Canada, Mexico and Puerto Rico.

Feedback Please send all comments and questions to **feedback**.

Copyright ©1996 Matsushita Electric Corporation of America. All rights reserved.
Matsushita Electric Corporation of America | One Panasonic Way | Secaucus, NJ 07094
Panasonic and Technics are registered trademarks of Matsushita Electric Industrial Co., Ltd.
Quasar is a registered trademark of Matsushita Electric Corporation of America.



```
[Image]
[Customer Satisfaction] [U.S. R&D Labs] [Matsushita Electric] [Device Drivers]
[Select Products] [Advanced Technology Products] [Select Products]
Special Notice to owners of Panasonic KX-P6100 and KX-P6500 printers

--------

[Image]
        [Image] [Image]Visit our new section on Panasonic Interactive Media.
                Previously known as Panasonic Software Company, PIM develops
                software for the popular 3DO platform, as well as for PC and
                Macintosh computers. We'll be adding more and more to our web
                site, so be sure to stop by again soon!

[Image]
        [Image] [Image]Panasonic is part of Matsushita Electric Corporation of
                America, which is also the home of Technics and Quasar brand
                products. You can check these pages for consumer related
                information and customer assistance telephone numbers. You can
                also download the latest Device Drivers for Panasonic Computer
                Peripheral Products via FTP.

                [Image]For information on our U.S.-based R&D labs, click to
                Panasonic Technologies, Inc. (PTI), which is currently the
                provider of this site. Also, we invite you to visit the newly
                re-designed Home Page of the worldwide headquarters for
                Panasonic, Matsushita Electric Industrial Co., Ltd. in Osaka,
                Japan.

[Image]
        [Image] [Image]Matsushita Electric Corporation of America, established
                in 1959, is the principal North American subsidiary of
                Matsushita Electric Industrial Co., Ltd. Along with our local
                subsidiaries and affiliates, we recorded sales of $7 billion in
                the fiscal year ended March 31,1995. Matsushita employs 16,000
                people in North America and has 21 manufacturing sites in the
                United States, Canada, Mexico and Puerto Rico.

[Image]
        [Image] [Image]Please send all comments and questions to feedback.

--------

      Copyright 1996 Matsushita Electric Corporation of America. All rights reserved.
   Matsushita Electric Corporation of America | One Panasonic Way | Secaucus, NJ 07094
Panasonic and Technics are registered trademarks of Matsushita Electric Industrial Co., Ltd.
      Quasar is a registered trademark of Matsushita Electric Corporation of America.
```

Figure 1: Example of a WWW page and its textual content.

possible to adopt the high-level view that all Web images are simply 2-D arrays of pixels, there are good reasons to distinguish between the formats and their various "flavors." Such an understanding can contribute to the design of better-quality algorithms for locating and recognizing embedded text, as well as suggest new topics for inquiry.

An important consideration, independent of format, is that WWW images are designed with the intention they will be viewed on computer monitors. This manifests itself in two ways: liberal use of color (up to 24 bits deep), and low spatial resolution (72 dpi). The former can actually make up for some of the limitations induced by the latter. Contrast this with the input to traditional document analysis, which typically consists of black text on a white background, printed and scanned at resolutions of 300 dpi or higher. A 10-point character nominally measures 83 pixels tall when scanned at 600 dpi. The same character may only be 10 pixels tall when displayed on-screen and yet can still remain readable if the proper colors are used. Figure 2 illustrates the difference in spatial resolutions for a line of text typeset in 10-point Helvetica.[1]

In the remainder of this section we discuss the GIF and JPEG image formats and other related details.

2.1 GIF Format

GIF (for "Graphics Interchange Format") was originally developed by CompuServe for use in exchanging images via e-mail. It is by far the most common image format encountered on the WWW today. In particular, nearly all of the small icons one finds sprinkled across Web pages are GIF's, as well as many of the larger images.

A significant limitation with the GIF format is that a pixel is at most eight bits deep. Hence, a maximum of 256 colors can be represented in a given image. The color palette can be optimized for the image in question, however, via a color map table that is specified in a control block at the start of the file. Unfortunately, certain Web browsers support only a single, fixed 256 color palette and resort to dithering to simulate colors not present. As a result, the user of a graphic may see something different than what the designer intended (this effect is probably minor enough not to be a concern for document analysis).

To save space and conserve bandwidth, GIF incorporates LZW compression. Since LZW is a lossless scheme, the decompressed image is the same quality as the original – there is never any degradation. Because the alphabet of possible symbols (*i.e.*, colors) is relatively small, LZW works well, especially when there are large regions of uniform color.

These factors make GIF an efficient format for stylized graphics, but less appropriate for photographs. There are, in fact, two standards for GIF in use today, GIF87a and GIF89a. The former is currently more popular, but the latter includes two significant new features: support for a transparent layer and animations. A transparent layer allows the background to show through. This allows for the creation of irregularly-shaped (*i.e.*, non-rectangular)

[1] As full color images, the examples presented throughout this paper are much more legible. However, the need to provide hardcopy that can be reproduced inexpensively means we must rely on grayscale for our figures.

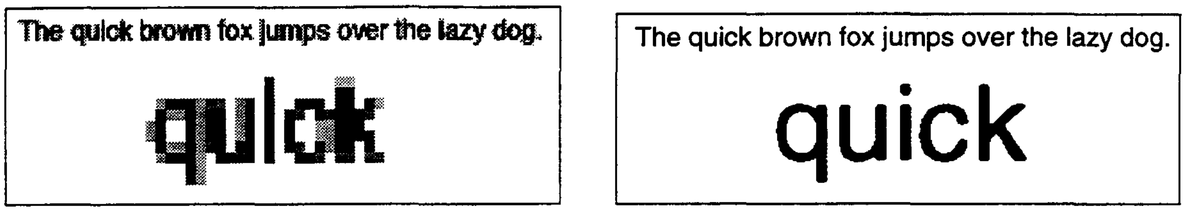

The quick brown fox jumps over the lazy dog.
quick
The quick brown fox jumps over the lazy dog.
quick

Figure 2: Text typeset in 10-point Helvetica for the Web (left) and printed/scanned (right).

images. Animations will be discussed in a later subsection.

2.2 JPEG Format

JPEG, developed by the Joint Photographic Experts Group, combines DCT-based compression in the frequency domain with Huffman coding of the resulting DCT coefficients.[2] It supports 24-bit color (i.e., any of over 16 million colors can appear in an image). This makes it the best choice for continuous tone images such as photographs. A familiar example of a JPEG image is shown in Figure 3.

Unlike GIF, JPEG is a lossy scheme. Compression is achieved by discarding the high order DCT coefficients. This saves space, but degrades the high frequency information in the image. JPEG supports variable settings, making it possible to trade off file size versus image quality. Compression ratios can approach 100:1.

JPEG is somewhat problematic when applied to images that contain text because of the artifacts it introduces. Text is, in effect, a high frequency signal. This fact combined with JPEG's block-oriented nature can lead to visible distortion in the vicinity of characters. A textured "halo" is apparent surrounding the letters in Figure 4 (especially obvious to the immediate left of the 'Y'). For this reason, JPEG is usually used for images consisting solely of photographic data (*e.g.*, the designer of Figure 9 choses to make it a GIF, most likely because of the text it contains).

2.3 Anti-aliasing

Anti-aliasing is the process of reducing the jagged appearance of a sharp edge by blending the pixels at its boundary with the background. It is independent of the image file format. Figure 5 illustrates this effect for a fragment of text originally typeset in 48-point Times.

While anti-aliased edges are smoother and more pleasing to the eye, there are in fact occasions where aliased text is preferable (see [13], page 96). This is particularly true for small font sizes – here anti-aliasing may result in excessive blurring, making the text more difficult to read (the text on the left side of Figure 2 was anti-aliased).

Since the matter is one of the designer's preference, text encountered in a Web image can be either anti-aliased or not. This issue could complicate the building of OCR classifiers for such characters.

[2]To be precise, JPEG is not a file format, it is a compression algorithm. The proper name for the associated file format is JFIF (JPEG File Interchange Format).

2.4 Spatial Sampling Effects

Lopresti, Nagy, Sarkar, and Zhou recently observed that when scanning a page, the effectively random placement of the sampling grid (*i.e.*, the CCD sensor array) relative to the page can lead to significant variation in the resulting character bitmaps [8]. An analogous effect occurs when producing a GIF or JPEG image using software such as Adobe Photoshop. Abstract characters are placed on a virtual grid with a much finer resolution than the final output. At some point, however, these must be mapped to the intended resolution; the abstract characters are sampled at the coarser grid rate, yielding variability much like the physical scanning process.

Figure 6 shows eight instances of the same 12-point Helvetica 'e' taken from an image rendered at Web resolution. The difference in the bitmaps is dramatic and due entirely to the placement of each character as the image was produced.

2.5 Dynamic Documents

Unlike scanned paper documents, Web pages can be dynamic. This nature is expressed in several different ways. Over a period of time, it is often the case that the same http pointer will refer to the most current version of a specific document. The home page for a research project, for example, might contain a list of the people currently working on the project. As people join and leave the project, the page is updated. This natural evolution over time can be captured and presented in ways useful to someone trying to understand the history of the project. Douglis and Ball describe one such system designed to track changes in the HTML text for Web pages [3]. This same notion could be extended to the analysis of Web images as well, although the technical issues would no doubt be more challenging.

Perhaps more germane to the topic at hand are the animated icons now becoming popular on some pages. These are, in fact, part of the definition of the GIF89a format. Multiple images can be stored in a single GIF file. Browsers that support this feature will then cycle through these frames in an endless loop (those that do not support it will display either the first or the last image in the sequence). Figure 7 shows three frames (out of 14) from an animated GIF.

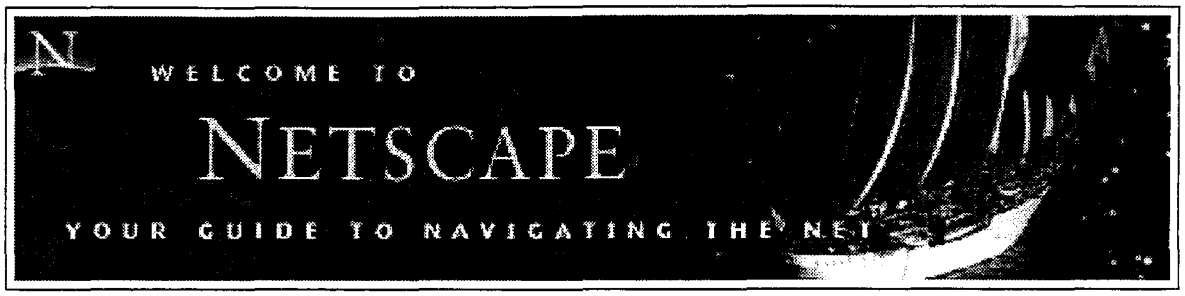


Figure 3: A JPEG image containing text.

It is quite evident to a human that this GIF contains a large letter 'e' with the word "mail" circling around it. Even ignoring the issue of detecting and representing the motion in the image, this is clearly a difficult problem. No single frame contains a clear, head-on view of all the letters together. The entire sequence of images must be analyzed and the results synthesized in order to determine the text contained in the animation.

2.6 Progressive Transmission

Bandwidth to/from the Internet is still very much limited. Because the delivery and display of image data, even when compressed, can require macroscopic amounts of time, both GIF and JPEG support the notion of progressive transmission. This allows the user to see a coarse approximation to the final image right away. The quality is then improved in successive steps as more data is received. If the user is able to recognize his/her goal before the download is complete, it is possible to jump ahead without waiting for the process to finish. This improves the interactivity of the Web.

In the case of GIF, this is accomplished using interlacing. GIF's LZW compression is raster-oriented; interlacing involves stepping through the image and transmitting every eighth row in turn. The result is a "chunky" effect until the image reaches its final resolution. For JPEG, progressive transmission involves sending the DCT coefficients proceeding from low order to high. The image resolves itself as more and more high frequency information is added back in.

While progressive transmission is important from the standpoint of on-line response, its potential impact on document analysis appears to be minor. Since image processing algorithms are computation-intensive and typically run in an off-line manner, the penalty of having a lower-quality image to work with more than offsets the small amount of download time to be saved.

2.7 Other Issues

The process of designing a Web graphic is a highly creative, labor-intensive activity. Since attracting attention is often the primary goal, a large number of different techniques are employed, some of which have already been touched on. The powers of human perception still greatly exceed what is possible using a machine. Text can be overlayed on a complex background and made so nearly transparent that it would be impossible to locate much less recognize using existing approaches. Consider, for example, the letter 'a' in Figure 8 which blends perfectly into the background texture and yet can easily be segmented out by a human. This makes the image more interesting, but much harder to analyze for its textual content. Weinman presents a good survey of the various issues that arise when designing graphics for the Web [13].

It should also be noted that the problem of locating and recognizing text in WWW images would be moot if designers always elected to include the embedded text in the source document. The HTML image tag has an "alternate text" parameter ideally suited to this purpose (the HTML comments tag could also be used). In point of fact, however, this is merely a convention; there is no way to enforce such a policy and, as we observed earlier, many Web pages simply ignore it.

Lastly, an important factor in the proliferation of image text on the Web has been the lack of control designers have over the typography of their pages. Other than specifying the font size and several other simple attributes (*e.g.*, bold, italics), a document's appearance is determined mostly by the user. To guarantee a certain look, the designer is forced to represent it in image form.

Recent extensions to HTML, however, are beginning to offer the designer much more flexibility in this regard. A number of companies have banded together to develop standards for directly supporting anti-aliased fonts, for example. Hence, one might argue that a key reason for wanting to embed text in images is about to go away, and therefore there is no need to worry about addressing the problem. To believe this reasoning, however, is to believe that the Web will become less multimedia-oriented in the future. This does not seem likely – if anything, use of the Web to deliver image data will continue to grow.

3 Text Region Location

One of the difficulties we face in recovering text in Web images is the detecting and recognizing text embedded in a complex color background. As men-

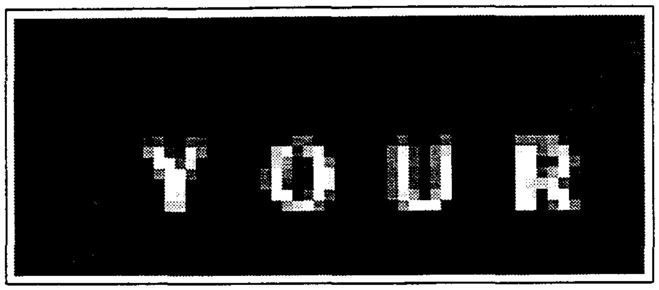


Figure 4: JPEG compression artifacts.

tioned earlier, text in Web images can have arbitrary colors and is often intermingled with differently colored objects in the background.

Locating text in complex color images has been addressed under different contexts in the literature. Most of the methods, however, are designed to process scanned images. In a paper by Zhong, Karu, and Jain the authors discuss two methods for automatically locating text in CD cover images [15]. Huang, *et al.* [5] proposed a method based on grouping colors into clusters for foreground/background segmentation of color images. In a paper by Doermann, Rivlin and Weiss [2], the authors discuss methods for extracting text (often stylized) from logos and trademark images. Another related work is the genetic-algorithm-based technique proposed by Zhou, Lopresti, and Zhou for extracting text from gray-level scanned images [18].

3.1 Color Clustering

We recently developed a text detection algorithm which is based on color clustering and connected component analysis. Conceptually, our text extraction algorithm follows a similar paradigm as Zhong *et al.*'s method. We first quantize the color space of the input image into color classes by a clustering procedure. Pixels are then assigned to color classes closest to their original colors. For each color class, we then analyze the shape of its connected components and classify components as character-like and non-character-like. Finally, character components are aligned to form text lines. In contrast to Zhong *et al.*'s method, however, we adopt a parameter-free clustering method which is intuitive and robust. In addition, a more sophisticated set of features are used for connected component classification in our algorithm.

The clustering method we used is based on the Euclidean minimum-spanning-tree (EMST) technique. The EMST-based clustering approach is a well-known method and has been researched extensively over the years. It has been shown that the method works well on a variety of distributions [14, 4].

To apply the EMST clustering algorithm to color image, we view each color as a point in a three dimensional RGB space. The distance between two colors is computed as the Euclidean distance of their RGB elements. An EMST is then constructed from these points. Edges on EMST whose distance is larger than the average are then removed from the EMST. The EMST tree thus trimmed may contain several disjoint sub-trees, each of which corresponds to a color cluster.

It is easy to see how an EMST can be used as a basis for clustering. By definition, each edge in the EMST of a set of points S is the shortest edge connecting two partitions P and $(S - P)$ of S. Intuitively, points in the same cluster should be connected by shorter edges than points in the different clusters. Hence, by removing edges with the longest distance in the EMST we separate the points of set S into disjoint clusters.

3.2 Connected Component Analysis

For colors in each cluster, we select the color which occurs most frequently in the image as the representative color of the cluster. After the color space is quantized, the next step is to find connected components belonging to each color class. For each connected component, a set of features such as the number of holes it contains, the numbers of upward ends or downward ends it has, *etc.*, are extracted from the component.

The algorithm then classifies connected components into two classes: text-like or non-text-like. The classification is based on the observation that connected components of characters have different shape characteristics than these from, say, graphical objects. A text component is often composed of a few strokes, with a relatively uniform width, and typically a small number of holes. A connected component from a non-text region, on the other hand, is irregular, with varying-size segments and many holes. Non-text components are then excluded from further analysis.

Finally, the algorithm performs a "clean-up" operation. It analyzes the geometric layout of text components. Heuristics such as text components often form co-linear lines are used during the process to eliminate spurious text components. Figure 10 shows the result of the text detection algorithm on the image in Figure 9. In this example, the original image contains 63 unique colors and the color clustering process groups them into 10 color classes (clusters). In a recent experiment, we tested the text extraction procedure on a set of WWW images extracted from the Web. The experimental results show that the text extraction algorithm works well

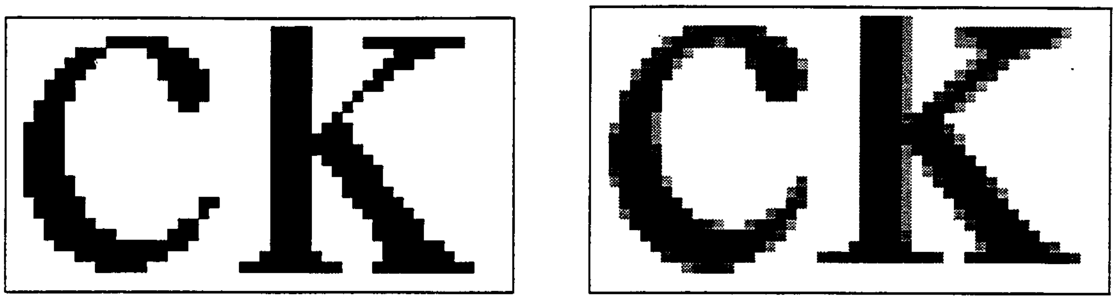


Figure 5: Aliased (left) and anti-aliased (right) text fragments.

on a variety of test images [16].

4 The Issue of Low Resolution

A major issue when attempting to OCR Web images is the ultra-low resolution at which text is usually rendered in these images. In a recent investigation, we estimated that much of the text in Web images is roughly equivalent to a 5 - 7pt font in a document scanned at 300 dpi [9]. Such character images are typically considered to be "difficult" for traditional OCR technologies.

On the other hand, text in Web images is rather "clean" – unlike scanned document, most Web images are created electronically, thus, do not contain noise such as blur, speckles. Moreover, Web images are usually free of distortions caused by skew, jitter, *etc.* This relative "cleanliness" of Web images suggests that in the development of recognition techniques, we may not need to be too concerned with the noise issue.

Nevertheless, text in Web images is not completely distortion-free. There can be distortion, for example, from compression loss, such as in JPEG compression. Another major distortion for Web image text comes from spatial sampling variation.

4.1 Color Information

Earlier studies have shown that it is possible to recognize text successfully at low resolution if color information is preserved during the digitization. For example, Lee, Pavlidis and Wasilkowski investigated the trade-off between spatial resolution and quantization resolution in signal processing [7]. It is predicted by the authors that for 10 point text the sampling rate could be as low as 100 dpi (but not much lower). Based on this result, Li and Pavlidis later proposed a character recognition technique which extracts features directly from the gray level of the scanned input image [12].

Inspired by these research results, we initially sought to apply Li and Pavlidis' idea directly on the OCR problem for Web images. We implemented a recognition process which was based on polynomial surface fitting. Our idea was to approximate a character shape by a polynomial surface function. We then extracted a set of features from the polynomial representation for recognition. This technique was later found to be unsuitable. The surface fitting method relies on the assumption that the color value being continuous. Web images, however, often use a small number of color shades. The high frequency of the intensity changes in colors makes the surface description unreliable.

4.2 N-Tuple Classifier

On the other hand, an alternative approach based on the n-tuple technique shows to be more promising for the problem. N-tuple classification is a rather old OCR technique, first proposed in 1959 [1]. It has received only scant attention since the early days of OCR research. A major weakness of the n-tuple method is its sensitivity to variations in the input pattern. In addition, the selection of "optimal" n-tuples is, in general, a computationally intractable problem. Our motivation for re-examining the n-tuple method comes from the observation that Web image data is relatively low of distortion. Finding distinguishing n-tuples may be more manageable in this case.

An n-tuple is simply a set of locations in an image bitmap. A typical statistical n-tuple classifier contains two stages: in the training stage, the color patterns at the locations as specified by an n-tuple are examined for each character class, and their corrresponding probabilities of occurrence are estimated from a sample set. For recognition, the same n-tuple is superimposed over the input image. The character class of which the color pattern appearing in the input is most likely to occur is assigned to the input. Usually, several n-tuples can be used in the classifier.

The majority of n-tuple methods reported in the literature are defined over a binary input space. An obvious way of applying such n-tuple methods to Web image OCR is to threshold the color image into a binary bitmap and treat the problem as standard n-tuple classification problem. The drawback of doing so is that the information contained in the color bits will be lost in the thresholding process. The approach we developed takes into consideration the "fuzziness" of the differentiation between foreground objects (*i.e.*, text characters) and the background in the input image. Instead of an exact "fit" our method measures the degree a tuple pattern matches the input image pattern and finds the class the tuple matches with a maximum likelihood.

To do so, we first assign to each pixel in the input

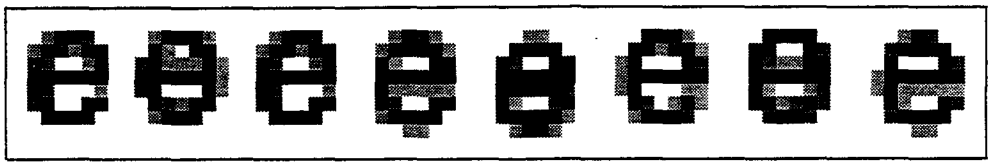


Figure 6: The spatial sampling effect for anti-aliased text (12-point Helvetica).

image a value between 0 and 1 which represents the certainty the pixel belongs to the foreground. Recall that the text location process assigns a representative color for each detected text component. We define the normalized color difference between the representative color and the actual color at a pixel as a measure of confidence the pixel belongs to the foreground.

To apply the n-tuple method, we first assess the confidence measures of pixels at selected locations as specified by an n-tuple, alternatively, being foreground or background. A joint probability distribution for all possible foreground/background configurations at these locations is then calculated. During the training process, the probability distribution is evaluated over a set of character samples for each character class. In the subsequent classification process, a Bayesian maximum likelihood classifier is used to determine the class of the input pattern. In a recently experiment, we collected a small set of character samples from Web documents and tested the algorithm. Preliminary results show that our n-tuple method worked quite well. The recognition accuracy for the sample data was 90.2% [17].

One issue which is yet to be addressed is how to select a set of appropriate n-tuples for the classifier. In the current implementation, the locations for each tuple are chosen randomly from the input image space. As it is pointed out earlier, for nontrivial recognition problems, it is usually computationally intractable to find the optimal choice of n-tuples. Nevertheless, practical searching algorithms have recently been developed for finding "good" tuples. In a paper by Jung, Krishnamoorthy, Nagy, and Shapira, the authors proposed two algorithms for generating, from a small training set, a collection of n-tuples which are "shift" invariant [6]. Such n-tuples fit each positive pattern in at least p different shift positions, and fail to fit each negative pattern by at least $n-q$ pixels in each shift position. "Shift invariance" is a desirable feature since this makes the tuples independent of sample variations. We are currently in the process of constructing a better n-tuple classifier and conducting a larger-scale test.

5 Conclusions

In this paper, we explored the issue of image processing for Web-based documents. We shown that the rapid growth of the World Wide Web, and its natural evolution towards more complex forms of multimedia, is creating significant opportunities as well as new challenges for document analysis research. In particular, we pointed out that the need for OCR still remains, and examined in some detail the problem of locating and extracting text from Web images. We described the current status of our work in this area. We also discussed various topics for further research.

6 Acknowledgments

The trademarks mentioned and/or depicted in this paper are the properties of their respective companies.

References

[1] W.W. Bledsoe and I. Browning. Pattern recognition and reading by machine. In *Proceedings of Eastern Joint Computer Conference*, number 16, pages 225–233, December 1959.

[2] D. Doermann, E. Rivlin, and I. Weiss. Logo recognition. Technical Report CAR-TR-688, Document Processing Group, Center for Automation Research, University of Maryland, College Park, MD 20742-3275, October 1993.

[3] F. Douglis and T. Ball. Tracking and viewing changes on the Web. In *Proceedings of 1996 USENIX Technical Conference*, pages 165–176, San Diego, CA, January 1996.

[4] R. Graham and P. Hell. On the history of minimum spanning tree problem. *Annals of History of Computing*, 7, 1985.

[5] Q. Huang, B. Dom, D. Steele, J. Ashley, and W. Biblack. Foreground/background segmentation of color images by integration of multiple cues. In *Proceedings of Computer Vision and Pattern Recognition*, pages 246–249, 1995.

[6] D.M. Jung, M.S. Krishnamoorthy, G. Nagy, and A. Shapira. N-tuple features for OCR revisited. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 18(7):734–743, 1996.

[7] D. Lee, T. Pavlidis, and G.W. Wasilkowski. A note on the trade-off between samping and quantization in signal processing. *Journal of Complexity*, 3:359–371, 1987.

[8] D. Lopresti, G. Nagy, P. Sarkar, and J.Y. Zhou. Spatial sampling effects in optical character recognition. In *Proceedings of the Third International Conference on Document Analysis and Recognition*, pages 309–314, Montréal, Canada, August 1995.

[9] D. Lopresti and J.Y. Zhou. Document analysis and the World Wide Web. In J. Hull and

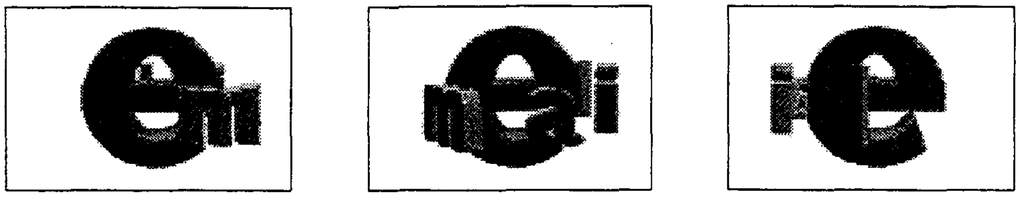


Figure 7: Three frames from an animated GIF.

S. Taylor, editors, *Proceedings of the Workshop on Document Analysis Systems*, pages 417–424, Marven, Pennsylvania, October 1996.

[10] G. Nagy, S. Seth, and M. Viswanathan. DIA, OCR, and the WWW. In H. Bunke and P.S.P. Wang, editors, *Handbook of Document Image Analysis*. World Scientific, Singapore, 1996. To appear.

[11] Kazem Taghva, Allen Condit, and Julie Borsack. Autotag: A tool for creating structured document collections from printed materials. Technical Report 94-11, UNLV Information Science Research Institute, Las Vegas, NV, December 1994.

[12] L. Wang and T. Pavlidis. Detection of curved and straight segments from gray scale topography. In *Proceedings of SPIE Symposium on Character Recognition Technologies*, pages 10–20, San Jose, California, 1993.

[13] Lynda Weinman. *Designing Web Graphics*. New Riders Publishing, Indianapolis, IN, 1996.

[14] C.T. Zahn. Graph theoretical methods for detecting and describing gestalt clusters. *IEEE Transactions on Computers*, 20(1), 1971.

[15] Y. Zhong, K. Karu, and A.K. Jain. Locating text in complext color images. *Pattern Recognition*, 28(10):1523–1535, 1995.

[16] J.Y. Zhou and D. Lopresti. Extracting text from WWW images. Technical report, Matsushita Information Technology Laboratory, Princeton New Jersey, April 1997. In submission.

[17] J.Y. Zhou, D. Lopresti, and Z. Lei. OCR on Web images. In *Proceedings of SPIE, Document Recognition IV*, San Jose, California, Febuary 1997. (To appear).

[18] J.Y. Zhou, D. Lopresti, and J. Zhou. A genetic approach to the analysis of complex text formatting. In *Proceedings of SPIE, Document Recognition III*, pages 126–137, San Jose, California, January 1996.

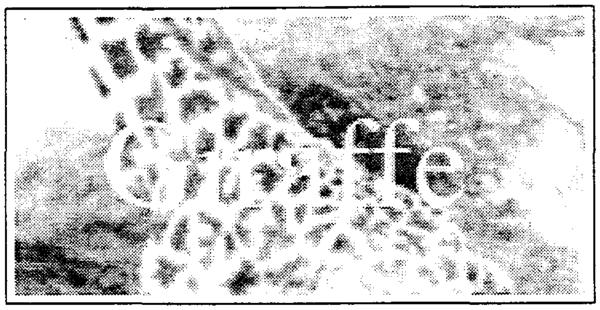


Figure 8: An image with hard-to-segment text.

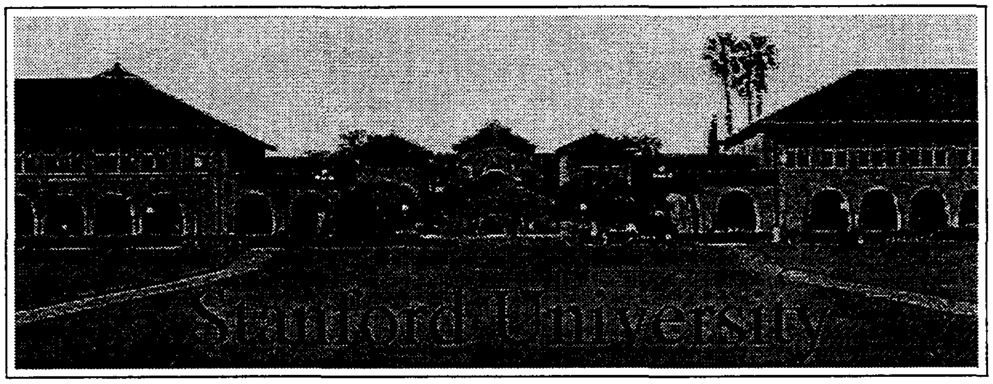


Figure 9: A Web image example.

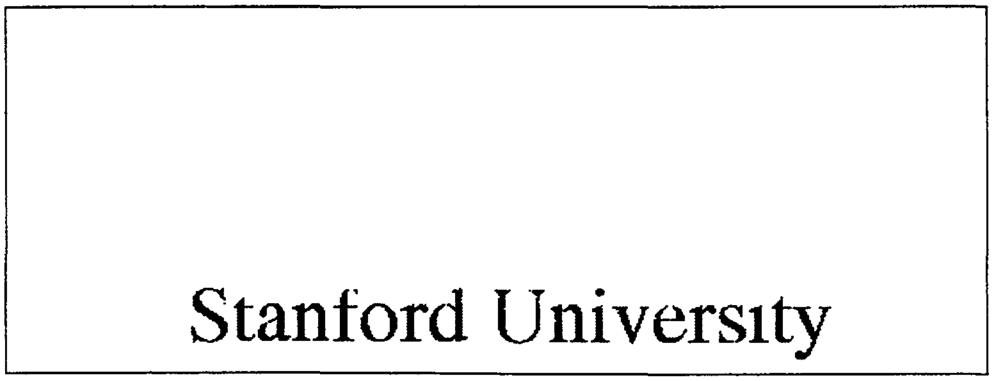


Figure 10: Result of the text detection algorithm.

Image-Based Indexing

Comparison Of OCR Versus Word Shape Recognition For Keyword Spotting

Jeff DeCurtins

Computer Engineering, Information, Telecommunications, and Automation Division
SRI International, Menlo Park, California 94025

E-mail: decurtin@erg.sri.com

Abstract

With the advent of on-line access to very large collections of document images, electronic classification into areas of interest has become possible. Classification (as we use it here) is a function of the presence or absence of selected keywords within a document's text. Two competing approaches are available for the detection of keywords within a document image. The first approach uses OCR on each document followed by analysis of the resulting ASCII text. An alternative approach is the use of whole word shape recognition (as opposed to individual character recognition) applied directly to the image. This alternative has been explored because of its speed and the expected robustness in the face of poor image quality that corrupts OCR output.

In this paper we describe the results of comparative testing between a commercial OCR engine and the word shape based system SRI has developed. The comparison discusses accuracy of keyword detection, speed of execution, and the relationship between accuracy and image quality.

1 SRI's Keyword Detection System

SRI has developed a system for keyword detection in document images. The system, called *Scribble*, uses the shape of the keyword as the primary retrieval property. For machine printed English text, the word shape is determined by the presence of *ascenders* (characters with components that rise above the height of a lowercase x) and *descenders* (characters with components that fall below the baseline of the text).

The major components of the Scribble system are shown in Figure 1.

1.1 Basic Approach

The two components of input to the system are the *lexicon* (i.e., the set of keywords to be detected) and the set of document images on which the detection algorithm will be run.

The word shape of each keyword within the lexicon is codified by a preprocess that uses models of individual characters to categorize each keyword with respect to its ascender/descender content. This process, performed once for any given keyword set, is very fast, even for very large lexicons (thousands of words).

As each document image is processed, a prerequisite to the detection of ascenders and descenders is the detection of a baseline and the measurement of lowercase height on an given line of text. An example of these survey lines is shown in Figure 2. This is accomplished by a document page decomposition algorithm that breaks the page image into regions of text, partitions each region into individual lines of text, and then partitions each line into individual words. The page decomposition process also determines document properties such as language (the system is being extended to handle non-English languages) and image quality (e.g., smearing and broken characters).

As each word is delimited, it is assumed a possible match to any of the words within the supplied lexicon. For a given keyword, (e.g. "might"), ascenders and descenders are detected (Figure 2) and their relative locations with the word are compared to the predetermined ascender/descender locations in each lexicon word. This comparison, combined with an estimate of the number of characters in the image word, typically reduces the number of possible matches by a factor of 100 or more. Once a small set of candidate matches is found, the ascenders and descenders present in the word image are segmented based on their possible character identities as derived from the

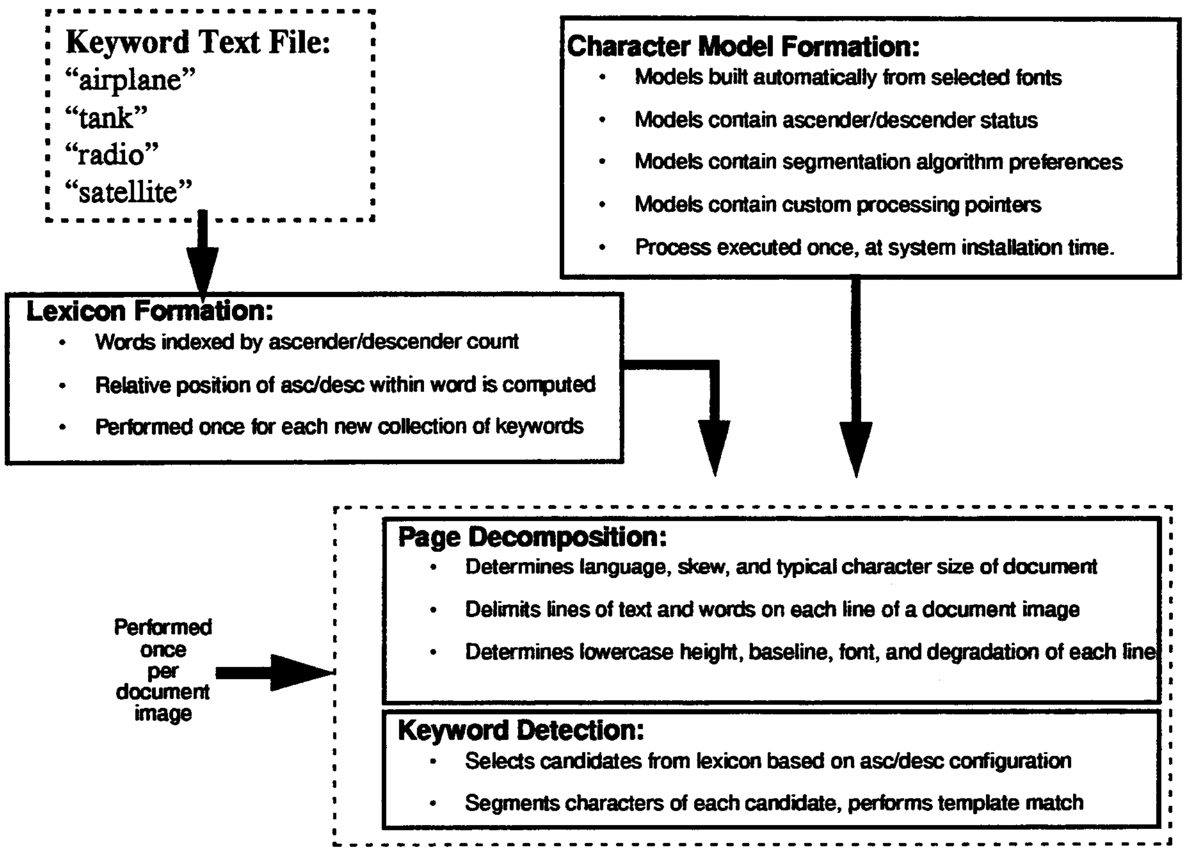

Keyword Text File:
"airplane"
"tank"
"radio"
"satellite"
Character Model Formation:
• Models built automatically from selected fonts
• Models contain ascender/descender status
• Models contain segmentation algorithm preferences
• Models contain custom processing pointers
• Process executed once, at system installation time.
Lexicon Formation:
• Words indexed by ascender/descender count
• Relative position of asc/desc within word is computed
• Performed once for each new collection of keywords
Performed once per document image
Page Decomposition:
• Determines language, skew, and typical character size of document
• Delimits lines of text and words on each line of a document image
• Determines lowercase height, baseline, font, and degradation of each line
Keyword Detection:
• Selects candidates from lexicon based on asc/desc configuration
• Segments characters of each candidate, performs template match

Figure 1. Scribble word shape recognition system components.

collection of candidate word matches. For example, suppose the image word "might" induced the collection of three lexicon candidates "might", "sight", and "capella". Character segmentation would be performed around the location of the first ascender, first assuming it was an "h" and then again assuming it was an "l". For the first (only) descender, the segmentation algorithm would assume a "g" and then try again assuming a "p" (see Figure 3). Note that because a specific keyword match has been hypothesized, the most appropriate segmentation algorithm can be selected for each edge of the character. The character segmentation procedure contains approximately a dozen such character-pair segmentation algorithms. In the case of "g" in "might" we segment assuming the "g" is preceded by an "i" and followed by an "h", thus the selection of the "Bottom Drop" and "Top Rise" segmentation algorithms (see Figure 4) When an hypothesized character has been segmented, its image region is mapped to a 16x16 pixel array and compared to a 16x16 generic (template) image of the given character (see Figure 5). The generic templates are derived from synthetic images of each character in different fonts. If the match is successful, the next ascender/descender is segmented and matched according to the keyword hypothesis. If all characters in the keyword are matched successfully, the document page is marked as containing the keyword.

2 Comparative Test Results

The performance comparison tests used a total of 380 document images. These images were derived from the University of Washington CDROM English Document Image Database I. To focus the comparison on the word recognition aspect of both Scribble and the OCR engine, zones containing only single columns of text within each of the 380 images were selected. These selected zones were then formed into new subimages from each original.

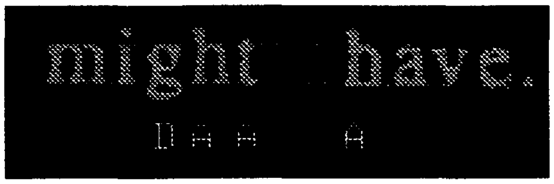


Figure 2. Baseline and lowercase measurement, ascender and descender detection.

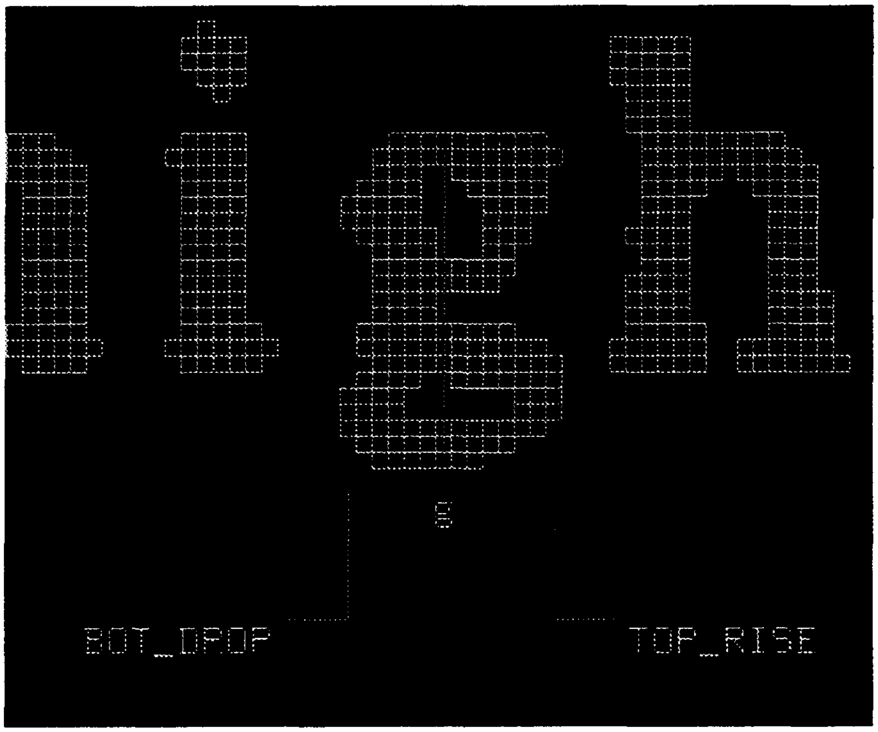


Figure 3. Segmentation based on hypothesis of character identity.

The image set contained a range of font styles and sizes as well as zone types (e.g., bibliographic references, titles, simple text lines).

2.1 Lexicon Construction

The set of images came from four groups of the images on the CDROM. For each image group, two sets of keywords were derived from the textual ground truth associated with each selected zone of each image. The first set of keywords for a group consisted of all words of four or more characters which appeared at least three times within the document group. The second set of keywords consisted of words at least eight characters in length which appeared at least three times within the group. The requirement that a keyword appear at least three times within the document group ensures that our statistics will reflect the search for the same word in different image contexts. The number of images in each group and the number of words in each of the two lexicons associated with each group is given in Table 1.

2.2 Overall Accuracy

The first test compared the accuracy of Scribble versus OCR. Recall that the basic task is the classification of document pages as "interesting" versus "not interesting" based on the occurrence of keywords. Thus the accuracy of a particular Scribble or OCR output for a given document was defined as the percentage of keywords correctly reported (Eq. 1). The complement of this measure is the false alarm rate, i.e., the number of times a keyword is reported as present

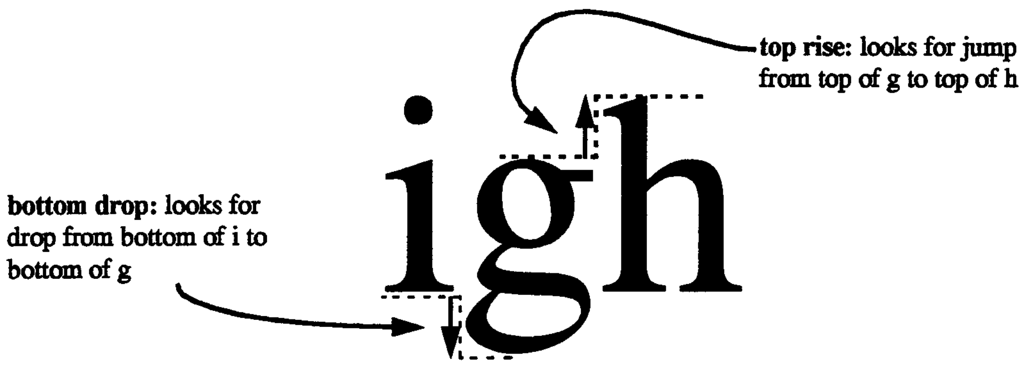

top rise: looks for jump from top of g to top of h
bottom drop: looks for drop from bottom of i to bottom of g
igh

Figure 4. Image features used for segmentation determined by hypothesis of character pair.

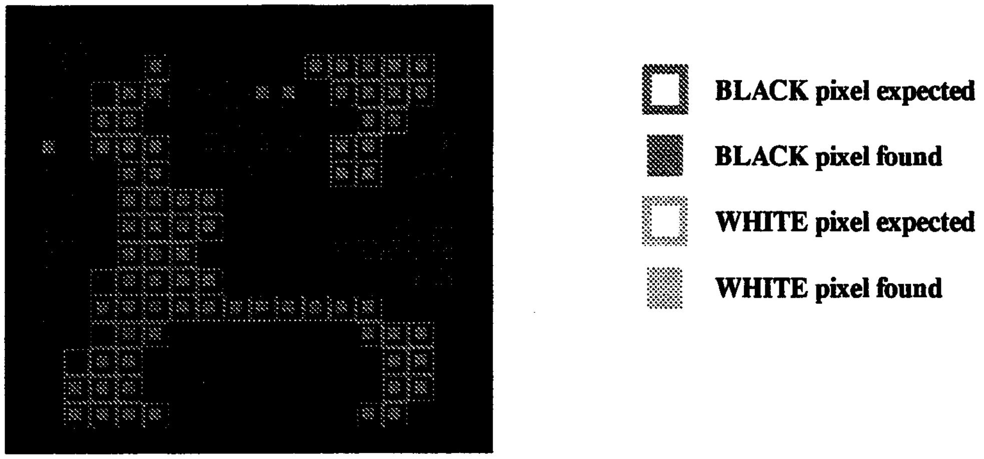


Figure 5. Template match for character g

$$\%\text{ACCURACY} = 100 \times \left(\frac{\#\text{ correctly reported}}{\#\text{ should be reported}}\right) \quad (1)$$

when it is not. Because both Scribble and OCR have low false alarm rates (below 3%), this measure was not tabulated.

When Scribble recognizes and reports a keyword, it does so based on the whole image of the word. In contrast, the OCR output is a text file which must be searched for the presence of expected keywords. Because OCR is a character (rather than word) based process, a single misrecognized character will render the keyword not found. One could allow for single or multiple missed characters in both the OCR output and in the Scribble process. This would increase the number of declared hits (and thus the accuracy as defined above) but at the cost of a higher false alarm rate.

The OCR engine used in these tests had a tendency to recognize the lower case letter "l" as the number "1". This is understandable as in many fonts these two glyphs look very much alike. To enhance the OCR performance levels, the keyword check of the output text file allowed for the replacement of an l with a 1. This added about 20% to the accuracy level of the OCR output. Similar problems were noted for the character combination "ri" which was often reported as an "n". The analysis of the OCR text output also allowed for this confusion.

Histograms of the accuracy of Scribble and OCR performance are given in Figure 6 and Figure 7. The results are summarized in Table 2. Note (in the

Table 1. Lexicon size for each image group

Group	# of images	# of words (>3 chars)	# of words (>7 chars)
A	95	1539	646
C	71	1119	472
D	89	1157	473
E	125	1847	794

histograms) that both OCR and Scribble failed catastrophically on some document images. Examples of images that caused these failures are shown in Figure 8.

2.3 Accuracy Versus Image Quality

One of the motivations for the use of word shape recognition techniques was the assumption that such techniques would remain viable in the face of poor image quality. To test this assumption, the average Scribble and OCR performances were plotted as a function of increasing image degradation (i.e., decreasing image quality).

Before this plot could be made, it was necessary to obtain some measure of image degradation. Although the Scribble page decomposition algorithm makes some estimates of the image quality of each textline, these estimates primarily measure the amount of smear (touching characters) in the image. Analysis of the poorly scoring images from the overall accuracy test indicated that poor performance was also linked to broken characters. Finally, the discrepancy between the OCR text output and the ground truth text on a character by character basis was used as a measure of image degradation. Although this means (almost by definition) that the OCR performance would decrease as this measure of degradation increased, we are only interested in discovering the tendency of the Scribble performance as the OCR performance changes.

Samples of images with high measured degradation are shown in Figure 8. Plots of accuracy versus image degradation for the two lexicons are given in Figure 9 and Figure 10. These plots support the initial presumption that the word shape algorithm is more robust than character based algorithms of standard OCR. In addition, the difference is more pronounced for longer keywords.

2.4 Execution Times

Three factors made it difficult to compare the execution times of Scribble versus the OCR engine.

First, the image files were supplied by a file server that was also servicing many other clients, thus response time was a function of the overall system load. This problem was mediated by running late at night while the system load was minimal.

Second, the two executables were on different machines, Scribble on a Sun Microsystems Ultra 1, the OCR engine on a Sun Microsystems Sparc 1. Both machines had comparable memory and access to the same file servers. Benchmarks suggest that the Ultra is approximately an order of magnitude faster than the Sparc. Thus an approximate comparison can be made by dividing each document's OCR time by 10 to obtain the execution time on an Ultra.

Finally, because Scribble incorporates knowledge of the keyword lexicon directly into the image analysis, Scribble's execution time is a function of the size of the lexicon itself. More words in the lexicon mean, on average, more possible character segmentations and matches to evaluate at each image word. The OCR search consists of two steps (the OCR process itself and the analysis of the output text file). Only the second step is a function of the lexicon size. The time taken for this second step was small relative to the OCR process time and so was not measured during the tests. For very large lexicons this text search time may become significant.

Accounting for the above mentioned factors, the execution time comparison is given in Table 3.

Table 2. OCR and Scribble Accuracy

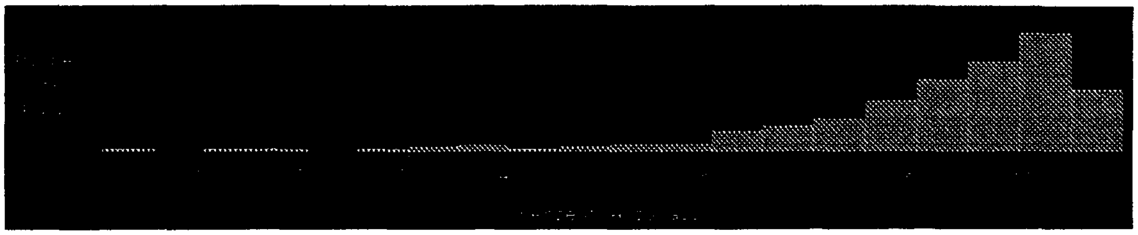
ENGINE	LEXICON	AVERAGE	MEAN
OCR	>3 chars	81%	86%
SCRIBBLE	>3 chars	87%	90%
OCR	>7 chars	74%	79%
SCRIBBLE	>7 chars	84%	88%

3 Conclusions

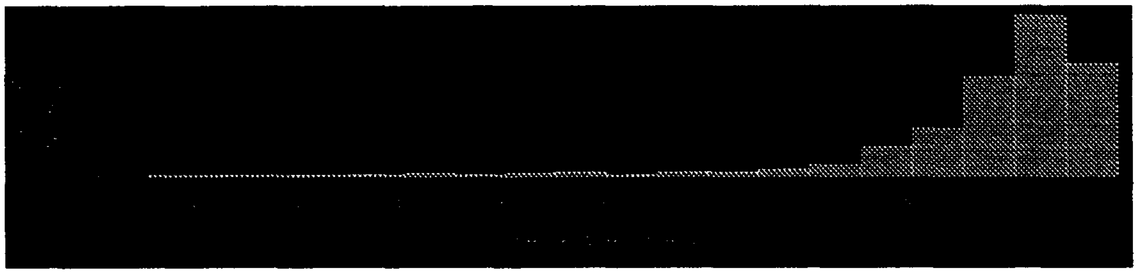
The assumption that motivated the development of word shape recognition systems for use in keyword spotting was that the performance of such systems would match or exceed that of traditional OCR, especially on images of poor quality. The tests described in this document have verified the assumption. On average, the Scribble word shape system was between 6% and 10% more accurate than OCR.

An additional benefit has been the reduced execution time per document. Scribble's average time per page was 50% to 80% faster than OCR. The worst case time for Scribble (which is a function of the number of textlines on a page) was only 4 times the average. For OCR, the worst case was more than 35 times the average.

It should be noted, however, that both Scribble and OCR encountered a few pathological cases in which performance of one or the other was exceptionally poor. Examination of the set of images that confused OCR did not reveal any clue to the OCR behavior. The images that caused Scribble to fail generally had lines that were misinterpreted at the page decomposition stage. Because such misinterpretations induce large fluctuations in Scribble's image property measurements, the fluctuation of these measures can be used to flag such documents for human attention off-line.

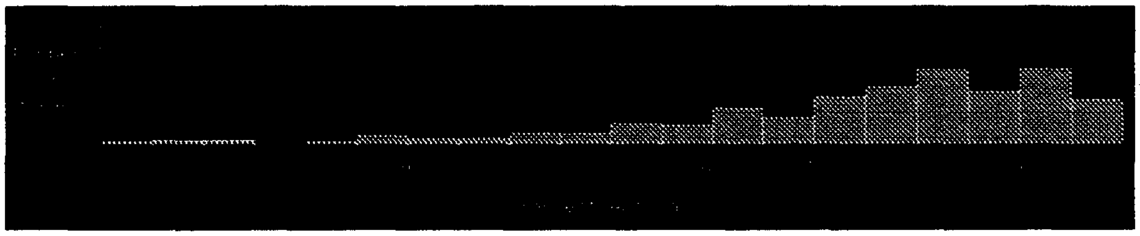


(a) OCR
Average = 81%, Mean = 86%

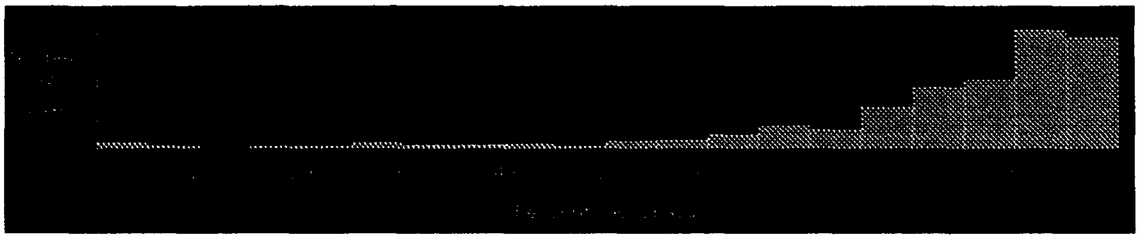


(b) Scribble
Average = 87%, Mean = 90%

Figure 6. Accuracy of OCR and Scribble on lexicons with keywords of 4 or more characters.

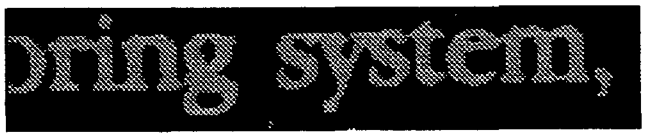


(a) OCR
Average = 74%, Mean = 79%

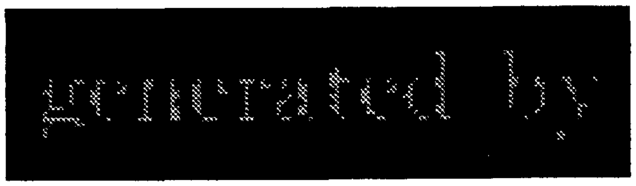


(b) Scribble
Average = 84%, Mean = 88%

Figure 7. Accuracy of OCR and Scribble on lexicons with keywords of 8 or more characters.



(a) touching characters



(b) broken characters

Figure 8. Samples of poor quality images.

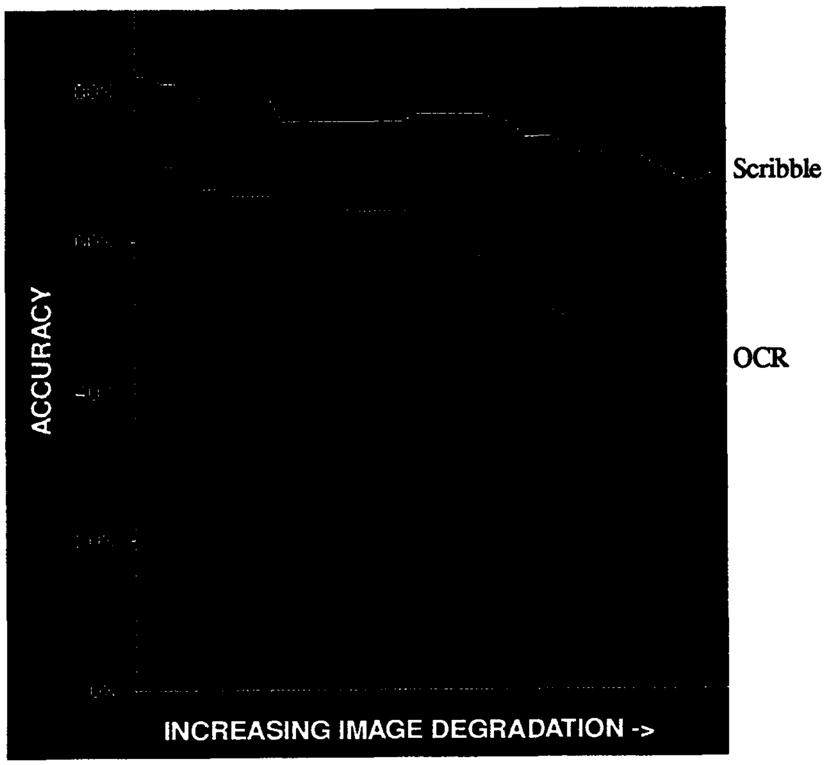


Figure 9. Accuracy versus image degradation for lexicon of keywords of length 8 or more.

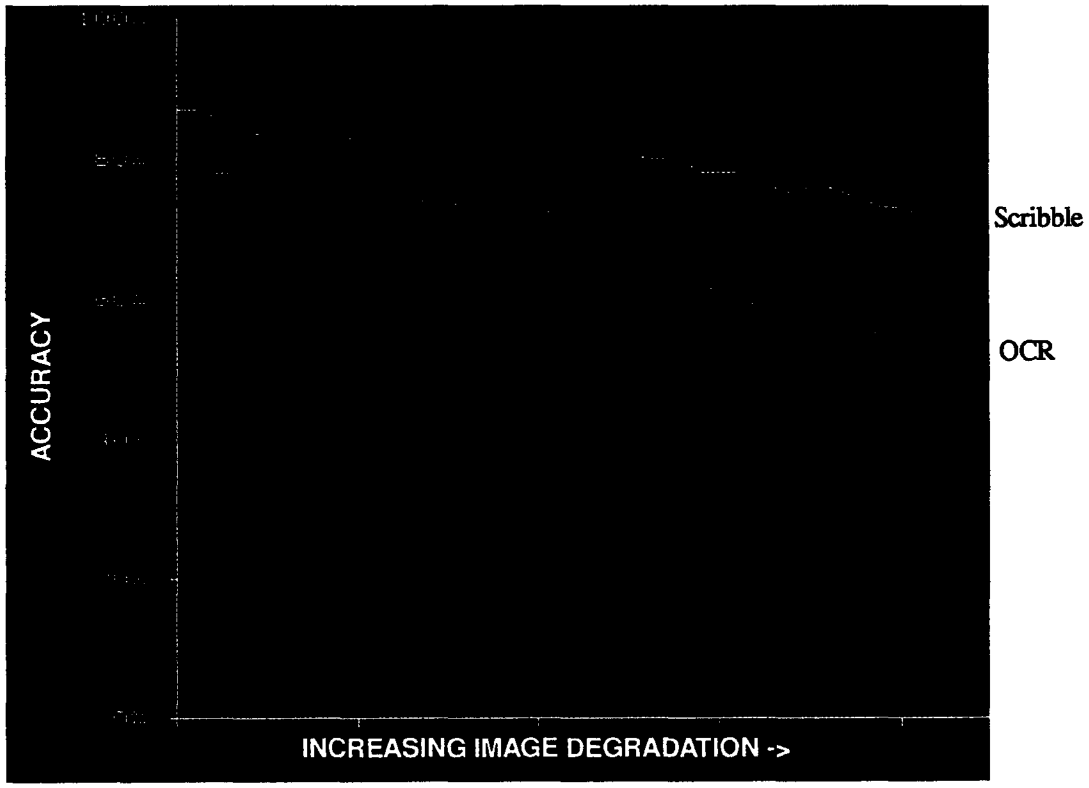


Figure 10. Accuracy versus image degradation for lexicon of keywords of length 4 or more.

Table 3. Execution Time[a] Comparison

ENGINE	LEXICON	AVERAGE TIME	MINIMUM TIME	MAXIMUM TIME
OCR	either	13 seconds	< 1 second	496 seconds
SCRIBBLE	>3 chars	6 seconds	< 1 second	24 seconds
SCRIBBLE	>7 chars	2 seconds	< 1 second	6 seconds

a. Times are for execution on Sun Ultra 1.

Word Spotting in Bitmapped Documents

William J. Williams, Eugene J. Zalubas and Alfred O. Hero, III
Electrical Engineering and Computer Science Dept
University of Michigan, Ann Arbor MI 48109

Abstract

Images and signals may be represented by forms invariant to time shifts, spatial shifts, frequency shifts, and scale changes. Advances in time-frequency analysis and scale transform techniques have made this possible. However, factors such as noise contamination and "style" differences complicate this. An example is found in text, where letters and words may vary in size and position. Examples of complicating variations include the font used, corruption during fax transmission, and printer characteristics. The solution advanced in this paper is to cast the desired invariants into separate subspaces for each extraneous factor or group of factors. The first goal is to have minimal overlap between these subspaces and the second goal is to be able to identify each subspace accurately. Concepts borrowed from high-resolution spectral analysis, but adapted uniquely to this problem have been found to be useful in this context. Once the pertinent subspace is identified, the recognition of a particular invariant form within this subspace is relatively simple using well-known singular value decomposition (SVD) techniques. The basic elements of the approach can be applied to a variety of pattern recognition problems. The specific application covered in this paper is word spotting in bitmapped documents.

1 Introduction

The recognition of specific signatures in images and signals has long been of interest. Powerful techniques exist for their detection and classification, but these techniques are often defeated by changes or variations in the signature. These variations often include translation and scale changes. Methods exist for transforming the signal/image so that the result is invariant to these disturbances. Translation and scaling are well understood in a mathematical sense, so it is fairly straightforward to design methods which yield a transformed form of the data wherein these effects are removed. There are other variations which are not well described mathematically or are not mathematically tractable in terms of reasonable transformations. This paper describes a combination of techniques which allow scale and translation invariant transformations to be used as one step of the signature recognition process. This is followed by an approach which separates the entities to be classified into a number of subsets characterized by additional variations. A new method is introduced to identify the subset to which the specific entity at hand belongs so that classifiers specific to that subset can be used. A two dimensional image is the basic starting point for the technique. This may be the actual image of an object or the two dimensional form of a signal representation such as a time-frequency distribution.

Classification of words appearing in different fonts and sizes serves to illustrate the methods developed. However, the approach is quite general and may be applied to a variety of problems and signals.

A representation termed the Scale and Translation Invariant Representation (STIR) is introduced here. It has desirable properties for pattern recognition under certain conditions. The object to be recognized is assumed to have consistent shape and appear on a constant intensity background. Using autocorrelation and the scale transform, one may produce STIRs which are identical for examples that have been translated on the background or scaled (compressed or dilated) along one or more axes.

Concepts borrowed from high-resolution spectral analysis, but adapted uniquely to the problem of classifying these STIRs have been found to be useful in this context. In high resolution frequency estimation, the noise subspace eigenvectors of the autocorrelation matrix are used. Pisarenko harmonic decomposition [1] employs the orthogonality of the noise subspace to the signal vectors to estimate sinusoid frequencies. This idea is used in the classification of signals following STIR processing.

A standard approach is to use the training data to generate templates for each class. A similarity measure, such as correlation coefficient, between the processed test data and each template is calculated and the test data is declared to be in the class corresponding to the largest similarity measure. In this noise subspace approach, an orthogonal subspace is created for each class of training data. A measure of the projection of the test data onto each of these subspace is calculated. Test data matching a given class should be orthogonal to the noise subspace for that class and yield a small projection measure.

The STIR and noise subspace classification method are applied to the example of word spotting in bitmapped documents. For a bitmap word input, the data are represented invariantly to translation and size, then submitted to a word detection algorithm. This combination of methods is applicable to many pattern recognition and detection problems of any dimension.

2 Background

The approach presented in this paper appears to be quite novel. There is not a lot of previous work that needs to be cited in order to build up to the approach. However, a few words on related approaches are appropriate. In addition, the time-frequency motivations behind the present work reveal the progression of the concepts and how they came to be applied to the word spotting problem.

2.1 Hidden Markov Model Approaches

Word and character spotting or recognition in documents have been a topic of interest for many years. A recent comprehensive review covers the field well[2]. The directions being taken now depart from historical approaches which depended on template matching and edge tracing. Many of these approaches have historically involved the "direct approach" wherein one attempts to capture obvious features. An approach based on hidden Markov processes (HMM) is more in line with the statistical approaches that we wish to employ, however. Recent reports[3, 4] might be considered to reflect the present success using this technique in word spotting in documents. The system was evaluated on a synthetically created database that contains about 26 000 words. They achieve a recognition accuracy using a 2-D HMM of 99% when words in testing and training sets are of the same font size, and 96% when they are in different sizes. In the latter case, the conventional 1-D HMM achieves only a 70% accuracy rate.

Word spotting in speech has also been accomplished using HMM techniques[5]. These studies were carried out using radio dialog. Both the HMM document studies and the HMM radio speech studies form an appropriate benchmark for our research. It seems that HMM techniques are widely regarded to be one of the best present approaches available.

2.2 Time-frequency concepts

The recognition of specific signatures in images and signals has long been of interest. Powerful techniques exist for detection and classification, but these techniques are often defeated by changes or variations in the signature. These variations often include translation and scale changes. Methods exist for transforming the image/signal so that the result is invariant to these disturbances. Translation and scaling are well understood in a mathematical sense, so it is fairly straightforward to design methods which yield a transformed form of the data wherein these effects are removed. A general philosophy of time-frequency representation termed the Reduced Interference Distribution (RID) approach has been developed by our group at Michigan[6, 7]. An illustration of the importance of this concept is shown in Figure 1. The spectrogram is an important tool in speech and other areas, but this figure reveals some of its problems which can be understood in terms of a full time-frequency conceptualization.

A considerable amount of work has recently been carried out on time-frequency distributions (TFDs), yielding newer TFDs which offer a considerable improvement over more traditional TFDs such as the Wigner distribution and the spectrogram[8, 9]. One can now readily design TFDs which represent the joint energy of a signal as a function of time and frequency or space-frequency distributions which represent the joint energy of images as space-spatial frequency distributions (two spatial variables x and y and two spatial frequency (wavenumber) variables Ω_X and Ω_Y). Furthermore, with careful design, these joint distributions can exhibit proper covariances with time, frequency or spatial shifts such that the representation shifts in accordance with these shifts but does not change in its configuration[10]. The well-known spectrogram has been extensively used in speech analysis and it has these useful properties. A shift in time or a shift in frequency of the signal[1] will shift the representation appropriately in time and frequency. However, the spectrogram does not exhibit the proper characteristics in response to *scale* changes in the signal. That is if $x(t)$ becomes $x(at)$, the Fourier transform of $x(t)$ changes from $X(\omega)$ to $\frac{1}{a}X(\frac{\omega}{a})$. This is illustrated in Figure 1.

One can readily see that the spectrogram results simply shift in time and shift in frequency in accordance with time and frequency, but the scaled

[1]Within reasonable bounds that do not induce aliasing or some other undesirable effect.

and time shifted click produces a somewhat distorted version of the representation with extra shaded regions above and below the main components as well as some changes in the main components. The RID result shown in Figure 1c illustrates the same basic invariance in the representations at the proper time and frequency shifts as does the spectrogram, but produces a result in keeping with the Fourier theory when scaling is applied. The RID result compresses in time and expands in frequency in accordance with the Fourier theory. The well-known wavelet transform would represent the original click and its time shifted and scaled version properly in that the scaled version would simply shift in scale. However, the representation produced by the time and frequency shifted click would be distorted in this case.

The spectrogram, the Wigner distribution and other time-frequency distributions such as the RID are members of what is generally known as Cohen's class of distributions. The RID enjoys many desirable properties that one would wish to have in time-frequency distributions, as does the Wigner distribution. However, the Wigner distribution exhibits troublesome cross-terms or interference terms between signal components. The RID reduces these interference terms considerably while retaining most of the high resolution of the Wigner distribution. The RID has revealed many hitherto unknown phenomena in many types of signals, including radar signals, sonar signals, machine signals, vibrational signals, speech, biological signals and biomedical signals to name a few (See Figure 1, for example). We have been quite active in pursuing the development and uses of the RID. One example of the RID that is often cited in the literature was developed here by Choi and Williams. We call this the Exponential distribution or ED. One designs TFDs by choosing certain *kernels* in Cohen's class of distributions. The general class is defined by:

$$C_x(t,\omega;\phi) = \iiint e^{j(-\theta t - \tau\theta + \omega u)}\phi(\theta,\tau)\, x(u+\tau/2)x^*(u-\tau/2)\,du\,d\tau\,d\theta \qquad (1)$$

where $x(t)$ is the time signal, $x^*(t)$ is its complex conjugate, and $\phi(\theta,\tau)$ is the **kernel** of the distribution.[2] The choice of the kernel dictates the properties of the resulting TFD. The kernel for the Wigner distribution is unity. The kernel for the spectrogram is a two dimensional filter based on the analysis window and the kernel for the RID is specially designed to retain many desirable properties while retaining high resolution in time-frequency.

Since there is an extensive literature in this area and a number of comprehensive reviews of this literature, we will not present the theory or discuss the applications in detail in this paper. There are several other useful TFDs which do not strictly conform to the constraints placed on the RID and thus do not enjoy all of its useful properties. Some are RID variations and others are developed under different assumptions.

[2]The range of integrals is from $-\infty$ to ∞ for this paper unless otherwise indicated.

We have carried out some initial work toward the goal of extending time-frequency techniques into the spatial realm. In this case, 2-D images in x and y are converted into 4D representations, representing not only the original x and y variables, but also their spatial frequencies as well. These results show that additional information about the image is indeed revealed by these representations which are an extension of TFDs. Some previous work on applying Wigner distribution concepts to images are very interesting and seem to have quite a lot of promise[11–14]. Advances in the theory and computer technology may now make it worthwhile to revisit such ideas.

Speech would seem to be a natural application of RIDs. A few studies of speech involving RIDs have been reported in the literature. We have carried out some preliminary work which is promising in terms of speaker recognition[15]. Combining RID ideas with the recently developed scale transform and some modified signal subspace concepts has led to a highly effective method of classifying signals and images. The scale transform must now be introduced so that its role in the approach can be appreciated.

2.3 The Scale Transform

The scale transform has been described by Cohen[16] to be:

$$D(c) = \frac{1}{\sqrt{2\pi}}\int_0^\infty f(t)\frac{e^{-jc\ln t}}{\sqrt{t}}dt \qquad (2)$$

The scale transform is of interest to this paper because it is able to remove the effect of scale in the images. There is an analogy to the Fourier transform. The Fourier transform of a signal, $x(t)$ and the Fourier transform of a shifted version of that signal, $x(t-t_o)$ differ only by a phase factor.

$$F[x(t-t_o)] = X_o(\omega) = X(\omega)e^{-j\omega t_o} \qquad (3)$$

so that

$$|X(\omega) = |X_o(\omega)| \qquad (4)$$

In a like manner, the scale transform of $x(at)$ differs from the scale transform of $x(t)$ only by a phase factor, so that the magnitudes of the scale transform of $x(t)$ and $x(at)$ are identical. Thus the effect of size changes can be removed by using only the magnitude of the scale transform.

$$|D(c)| = |D_a(c)| \qquad (5)$$

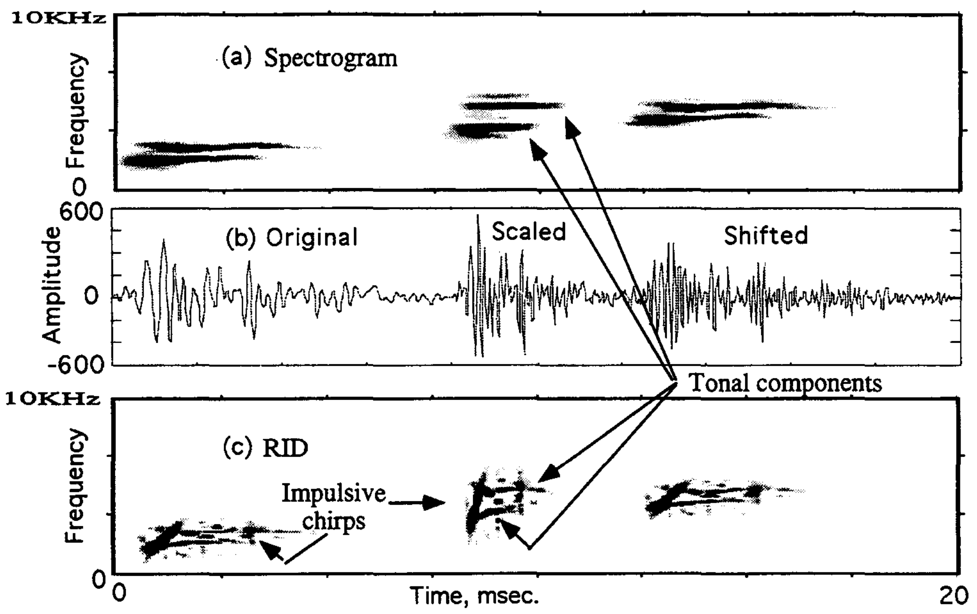


Figure 1: TFD results for time shifted, frequency shifted and scaled dolphin click. (a.) Spectrogram. (b.) Original click, scaled and time shifted click, time shifted and frequency shifted click. (c.) Reduced Interference Distribution (RID) click results for the same time shifts, scaling and frequency shift. Reproduced from[10].

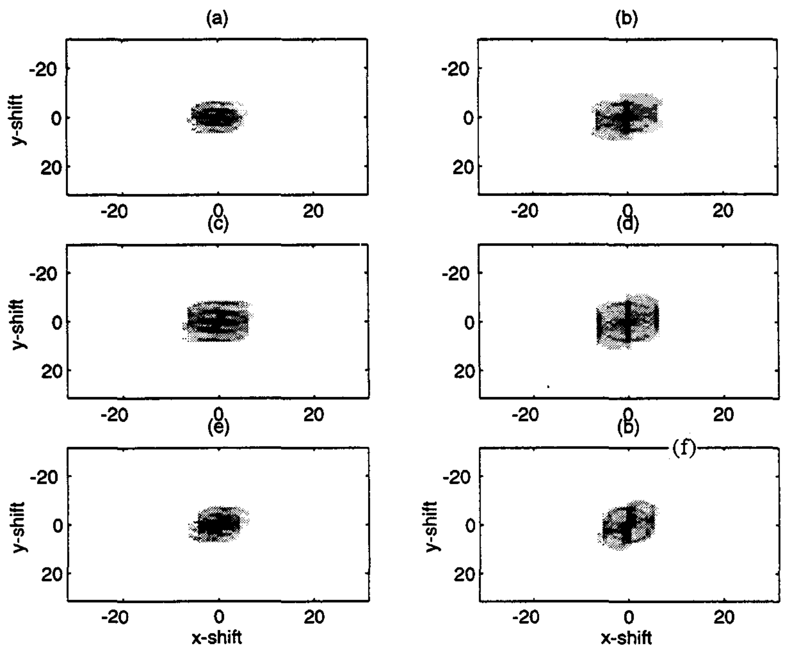


Figure 2: Typical 2-D autocorrelation result for a 63x63 pixel bitmap of two lowercase letters. (a) 'a' in Courier(12pt) , (b) 'b' in Courier(12pt),(c) 'a' in Helvetica(12pt), (d) 'b' in Helvetica(12pt), (e) 'a' in Times(12pt) , (f) 'b' in Times(12pt).

We have developed a discrete form of the scale transform[17, 18] which can be computed efficiently. One might question the use of the scale transform rather than the more well-known Mellin transform. One reason is that the standard Mellin transform weights signal components in lower time more than in higher time. A second reason is the relationship of scale to wavelet concepts and the insights it brings in this light. We have adapted Mellin transform ideas from the paper by Zwicke and Kiss[19] for our scale transform applications.

2.4 Scale transform applications to speech

Cohen, in his theory of scale, treats it as a physical variable just like frequency. Marinovich et al[20], have shown that the concept of scale can be profitably used to understand the nature of the speech signal. To illustrate the main issue that motivated the work consider the example of a grown person and a child making the sound "ah". The time-frequency structure of these two sounds is different. The frequency bands, the formants, are at different locations; the frequency spacings between the formants are different. However, one interprets the sounds as the same. How is that possible? If the two sounds had spectra that were scaled versions of each and the auditory system unscaled them, they would be perceived the same. Indeed it has been experimentally shown for vowels that sounds which one categorizes as the same but produced by different size vocal tracts have spectra that are scaled versions of each other, where the scaling occurs in the frequency domain.

We are able to cope with the various types of transformations performed on the signals and images which occur in our environment - time shifts, frequency shifts, translations in space and scale changes. In order to devise systems which will perform under these conditions it is necessary to cope with these changes in the design of the algorithms.

2.5 Tools for Invariant Image Representation

Several tools have been developed for representation of the images we seek to classify. In this section we will confine ourselves to 2-D images. These results were obtained when we decided to "back-off" from the full 4D representations mentioned previously for a time due to their representational and computational complexities. The idea was to gain insight in a simpler setting and then return to the more complex representations as experience dictates. The more complicated 4D functions required for the 4D representations were thus temporarily replaced by 2-D autocorrelations. The steps in the image processing algorithm which are reflected in the character, word and sound classifications are:

- Autocorrelation of the 2-D representations to remove translational effects.
- 2-D scale transformations of the the autocorrelation result to remove scaling effects.
- Partition of the results into subsets which reflect extraneous variations of the data.

Classification of the image involves two steps. These are:

- Determine the subset to which the unknown image belongs.
- Use the classifier designed for that specific subset to classify the image.

2.6 Computation of the 2-D Autocorrelation

The autocorrelation function of the signal provides the stable origin needed by the scale transform. Since the autocorrelation simultaneously sums over all points of a function, shifting of a signal over the plane does not affect the values for each lag. It is well known that autocorrelation removes translational effects in images and specifically in optical character recognition (OCR) methods[2].

The 2-D discrete autocorrelation may be carried out as follows[3]:

$$A(k_1, k_2) = \sum_{n_1} \sum_{n_2} a(n_1, n_2) a(n_1 - k_1, n_2 - k_2) \quad (6)$$

where $a(n_1, n_2)$ is the image. The image need not be centered within the bitmap representation, which has finite support in n_1, n_2. The bitmap is assumed zero outside of the specific bitmap support region chosen.

The 0,0 lag point provides an origin from which the autocorrelation function scales. Another feature of the 2-D autocorrelation function is the symmetry $A(k_1, k_2) = A(-k_1, -k_2)$. Hence, the first and fourth quadrants together contain complete information about the entire autocorrelation lag plane. This attribute will be used in applying the scale transform. An autocorrelation function for a bitmap of the word VAN is shown in Figure 5

For pattern recognition purposes, one must be aware of the loss of information which results from obtaining the autocorrelation of the signal. The goal

[3]The reader will probably recognize that this computation can be done in the frequency domain by taking the squared magnitude of the 2-D Fourier transform of the image with subsequent 2-D inverse Fourier transforming

here is to remove only translation effects. Unfortunately, due to the symmetry of the autocorrelation function, an ambiguity in the orientation of the original image is introduced. The autocorrelation of an image is indistinguishable from the autocorrelation of a 180 degree rotated version of the image. This is due to the masking of phase information when the autocorrelation is applied to a signal.

2.7 2-D Direct Scale Transform

The scale transform is based on exponential sampling relative to the origin,so the entire autocorrelation plane cannot be dealt with at once. Since both lag values in the first quadrant index from zero in the first quadrant, the scale transform may be directly applied. The lag axes in the fourth quadrant, however, aren't both positive, so reindexing is necessary. For each quadrant the axes must be included, since the scale transform indexing is based relative to the origin.

Hence, define two discrete quadrant functions of the Autocorrelation plane as follows:

$$Q_1(k_1, k_2) = A(k_1, k_2) \quad \text{for } k_1, k_2 \geq 0 \tag{7}$$

$$Q_2(k_1, k_2) = A(k_1, -k_2) \quad \text{for } k_1, k_2 \geq 0 \tag{8}$$

A 2-D scale transform approximation is implemented by applying the 1D scale transform algorithm in (2) first to the rows then to the columns of a matrix of values. Applying such a 2-D scale transform to Q_1 and Q_2 and taking the magnitude of the result yields two 2-D matrices of scale coefficients. The size of these matrices is determined by the number of row and column scale values selected. These two matrices are called ACFSXs (Autocorrelation Functions - Scale Transformed).

Since the autocorrelation function input was not energy normalized, normalization of the scale magnitudes is required for a scale invariant representation. Since the scale transform is a linear transform, normalization may be done by a variety of methods to generate an appropriate result.

The two ACFSX matrices together represent a STIR of the original 2-D input. Since it is not possible to calculate the scale coefficient $D(c)$ for every c, a set of scales is chosen for computation of the scale transform coefficients. Hence, the transform is not invertible. In addition to providing a scale invariant representation, other signal information is lost. The usefulness of the STIR is dependent on its implementation and application. For the very common case of a 2-D function sampled into a matrix of discrete values, we have developed a classification scheme which can be used with STIRs as the inputs.

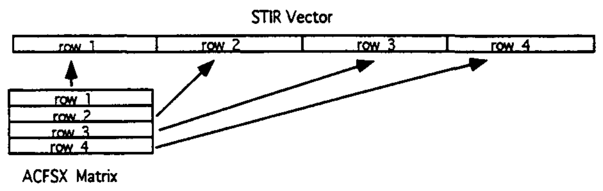


Figure 3: Forming one of the components of the STIR Vector from one of the ACFSX matrices

The novel image classification approach involves two steps. These are:

- Determine the subset to which the unknown image belongs.
- Use the classifier designed for that specific subset to classify the image.

2.8 Designing the Classifier

Suppose that the invariant form is characterized by a two dimensional representation $\Delta(p, q)$. This 2-D representation may be decomposed using eigensystem techniques as

$$\Delta(p, q) = \sum_j a_j \beta_j(p, q) \tag{9}$$

where the $\beta_j(p, q)$ are eigenimages and the a_j are the eigenvalues of the decomposition. The eigensystem decomposition is carried out on a collection of $\Delta(p, q)$ examples coming from the classes of objects (signals or images) that are of interest. The eigensystem decomposition then provides an ordered set of eigenimages ordered according to their eigenvalues. Although the eventual goal is to use true two dimensional eigenimage analysis, suitable algorithms to accomplish this have not been identified. One may utilize a simpler one dimensional approach which lends itself to readily available algorithms.

2.9 Classification of Patterns

Our technique for pattern classification uses STIR images decomposed into an orthonormal set of descriptors, using a concept borrowed from Pisarenko's harmonic decomposition [1, 18]. The Singular Value Decomposition (SVD) is used here. The STIRs of each exemplar in a class are shaped into a row vector by concatenating vectors formed from the rows of the two ACFSX matrices (See Figure 9). These row vectors are stacked to form a matrix representing the class. The SVD is then applied to extract essential features of the set of vectors. Provided that a sufficient number of scale coefficients are calculated, singular values of zero will result. Right singular vectors corresponding to zero singular value define a subspace orthogonal to the class of STIR vector representations.

In classifying a test signal, generate its STIR vector. Compute for each class the sum of inner product magnitudes of the STIR vector with the orthogonal subspace vectors. If the sum is zero, then the test signal must be a member of the corresponding class. In practice, one does not obtain a zero sum with the proper subspace class, but the sum resulting from the proper class has the smallest magnitude relative to sums from calculation with other class subspaces.

In addition to the invariances, STIRs have the desirable property that for a fixed set of row and column scales the sizes of all STIR matrices are identical, regardless of the size of the input matrices. Hence, inputs from different sources may be treated identically once processed into STIR images.

2.10 Classification of Characters

The initial approach which was taken was to decompose the STIR images via singular value eigendecomposition (SVD) in order to provide an orthonormal set of descriptors. In order to accomplish this, the STIR images of the characters were reshaped into a single vector by concatenating the rows of the STIR matrix to form a STIR vector. A new matrix consisting of all of the characters of interest for a range of sizes and the three fonts was formed from these vectors. The SVD was applied to extract essential features of the of the set of vectors. Several singular vectors with the largest singular values were chosen as features. Unfortunately, classification results were not impressive. The font variations were sufficient to reduce classification accuracy below acceptable levels. In order to combat this problem, the results were separated into subsets, with one subset representing each font type. Then, a novel orthogonal noise subspace method was used to identify the specific font used to produce the unknown character.

Almost all of the undesired variation due to shift and scale may be squeezed out of the final invariant form. There may still be some residual effects due to discretization and computation.

The $N \times M$ STIRs may be converted into vectors of length $N \times M$ by either concatenating the rows or columns. Then, readily available Singular Value Decomposition (SVD) techniques may be applied to the vectorized set of images. Suppose there are several different extraneous variations in the supposedly invariant representations caused by a variety of factors. Representation by a variety of font types and pixelation errors as well as FAX noise are examples of these extraneous variations. Conversion of the STIR matrices into STIR vectors is illustrated in Figure 9. Two such conversions, when concatenated form the STIR vector.

3 Noise Subspace Method Details

The $N \times M$ STIRs have a large number of elements. Usually, for classification methods to work, one wishes to have a considerably greater number of representations of the signal vectors than there are elements in those representations. Here, we have exactly the opposite. There are many more elements in the vectorized 2-D forms than there are vectorized 2-D forms. This is usually a statistical nightmare. However, suppose there are K examples ($K << N \times M$). Then the SVD produces $N \times M$ orthogonal eigenvectors, the first K of which form a complete orthonormal basis for the vectorized invariant forms. The remaining SVD eigenvectors (the noise eigenvectors) must be orthogonal to all of the original vectorized invariant forms. Suppose that we now obtain a new example[4]. Convert it into the STIR form and then, vectorize it to form the STIR vector. If it belongs to the set of STIR vectors used to produce the SVD results, then it should be **orthogonal** to all of the noise eigenvectors produced by the SVD. Therefore, its projection on any of the noise eigenvectors should be zero. If we have carried out the whole process through the SVD for a number of different sets of signals, we should find the projection of the STIR vector of the unknown signal on the noise eigenvectors of each set of signals. The smallest result will be theoretically obtained when this is done using the noise eigenvectors of the set to which the signal belongs. This idea may be expressed more formally. The SVD is performed on the set of matrices formed by STIR vectors for each font. Denote this matrix to be Q_k, where k is the kth font. The variety of the STIR vectors which form the rows of this matrix should cover a full range of variations of all of the characters represented by that font. Suppose there are N_c characters of interest. This could be all letters, special characters and numbers. This number would be multiplied by the number of font sizes of interest to yield N_{tot}. Application of the SVD yields

$$U_k S_k V_k' = Q_k \tag{10}$$

The N_{tot} columns of V form a basis for the rows of Q_k. If there are columns left over they will be orthogonal to all of the rows of Q_k. This may be accomplished by designing the STIR representations such that the STIR vectors meet this condition by having more elements than there are rows in the Q_k matrix. For illustration, suppose one picks a "noise vector" Z from this set of orthogonal columns, call

[4]If images are the inputs proceed, if a signal convert it into a TFD image and proceed

it Z_k. It is a column vector. Then,

$$Q_k Z_k = O \quad (11)$$

Where "O" is a row vector of zeros. However, for another font representation, Q_r,

$$Q_r Z_k \neq 0 \quad (12)$$

This provides a powerful means of determining whether or not a STIR vector belongs to the subspace of interest. Find

$$s_u = STIR_u Z_k \quad (13)$$

If the selection value, s_u, belongs to the kth subset, it projection will be zero. If not, its projection will most likely be large. Thus one may detect the font representation by this means.

3.1 Partition into Font subsets and Detecting Font

The difficulties due to font differences have been solved by first detecting the font in which the unknown character is represented. Ideally, the subspaces represented by the different fonts would be disjoint, so that one may discover which font the unknown is in and then chose a font specific classifier to home in on the character. This is not quite true, but the subspaces for the different fonts are sufficiently distinct to provide good font detection.

3.2 Font Specific Classification of Characters and Words

Suppose one has the bitmap of an unknown character. The STIR representation of the character is projected onto the subspace of each font. We can thus find a selection value that determines the font class membership of the unknown character. Next, the classifier designed for the specific font subspace is used to classify the character. This subspace is built from all of the characters in the search set represented over a number of sizes. We have used a wide range of font sizes from 10 to 50 in building these font specific subspaces. This method works very well using a number of classification methods. We use the most important features extracted from the SVD decomposition of the STIR vector. Typically, we obtain 100 percent correct character classification, even with fax corrupted bitmaps[18]. On rare occasion incorrect font subspaces are selected. Nevertheless, the correct character classification still results. This is the ultimate goal, after all. Also, on rare occasions, "b" is confused with "p" and "u" is confused with "n" in some fonts represented by noisy bitmaps. This is due to strong symmetry for such characters in the STIR representation. It should be easy to confirm the correct result by other simple means in such cases.

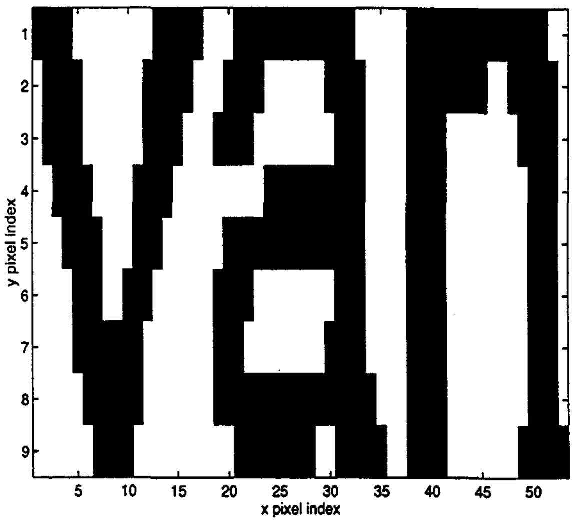


Figure 4: Bitmapped Representation of VAN

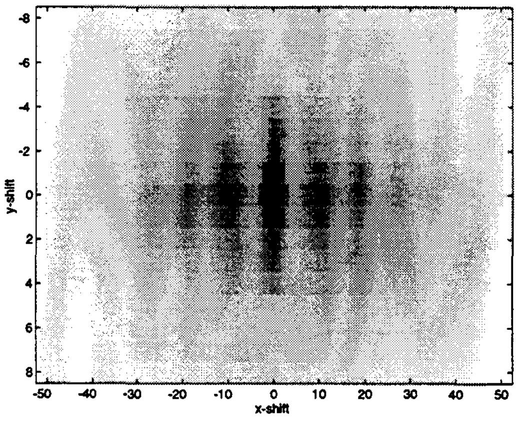


Figure 5: 2-D Autocorrelation of VAN

4 Application: Wordspotting

This example shows how STIR and the SVD noise subspace index are combined to perform as a size independent word recognition classifier. A complete document identification system incorporates much more than the pattern classifier presented here. This application shows the viability of the method for pattern classification. Performance on multiple fonts is a straightforward extension of the single font wordspotter.

STIRs and noise subspace methodology are used to spot a word in text independent of size or translation. Omitted is the task of segmenting an image into individual word bitmaps. Given the additional white space between words in a document, the segmentation task is much easier than character segmentation. For many documents, this may be simply performed by breaking text at intervals of white space which exceed a given distance.

Each segmented bitmap is considered as an isolated recognition task. Contextual information such as positioning within a line, height/width ratio, and pixel density is not used.

The word set consists of three letter words all in lowercase. The word to spot was VAN. 75 words other than VAN were generated by altering one letter of the three. Helvetica was the font examined. Text in sizes 9, 11, 12, 13, 14, 15, 16, 17, 18, 19, and 20 point was used for training. Bitmaps from faxed versions of clean printed copy were used as the input signals. The training text consisted of five instances of the word to be spotted in each size. A test bitmap of the word VAN in 10 point appears in Figure 7.

The classification methodology was tested on bitmaps of 10 point words in constructed from faxed images. Hence, the recognition tool is being tested on a size of text different from any size used in training. In this character recognizer, Font is determined first. For each font, exemplars in the four training sizes are available for each of the 26 characters, a total of 104 training characters.

Every STIR row vector is generated by the steps of autocorrelation, scale transform, and reshaping to a vector. To illustrate, consider a 9 point bitmap of the desired word 'van'. Figure 5 shows its autocorrelation. The first and fourth quadrants are scale transformed using an interval distance $T = 1$ with row and column scale values of 0.1,0.4,0.7,1.0,1.3,1.6,1.9,2.2,2.5,2.8. Figure 8 shows the matrices of magnitudes of these scale transform coefficients, the STIR values. Note that the difference in scale values between quadrants is very small. This similarity is exhibited in the scale coefficient magnitudes for most data encountered. Another notable feature is that the scale magnitudes generally show a roughly exponential drop off. These coefficient magnitudes reformed as an STIR row vector give the appearance shown in Figure 9.

Since, in this example, only one word is to be spotted, STIR training vectors formed only one matrix. This implies an SVD on 45 STIR vectors since we are using 5 instances of 'van' in each point size 9, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20. The length of each row is determined by the number of row and column scales chosen for calculation. The STIR row vectors each have 200 elements because, choice of row and column scales in the scale transform dictates a 10 by 10 matrix output for each autocorrelation quadrant, regardless of the size of each autocorrelation quadrant. Thus, the SVD for each font will yield noise vectors corresponding to $200 - 45 = 155$ singular values with zero magnitude. Calculating the sum of inner product magnitudes between these orthogonal vectors and a test STIR vector yields a selection value for each font. If the result is zero, then the unknown character must be represented in that font. In practice, one does not obtain a zero inner product with the correct font noise vectors. However, the correct font should correspond to the matrix generating the smallest selection value.

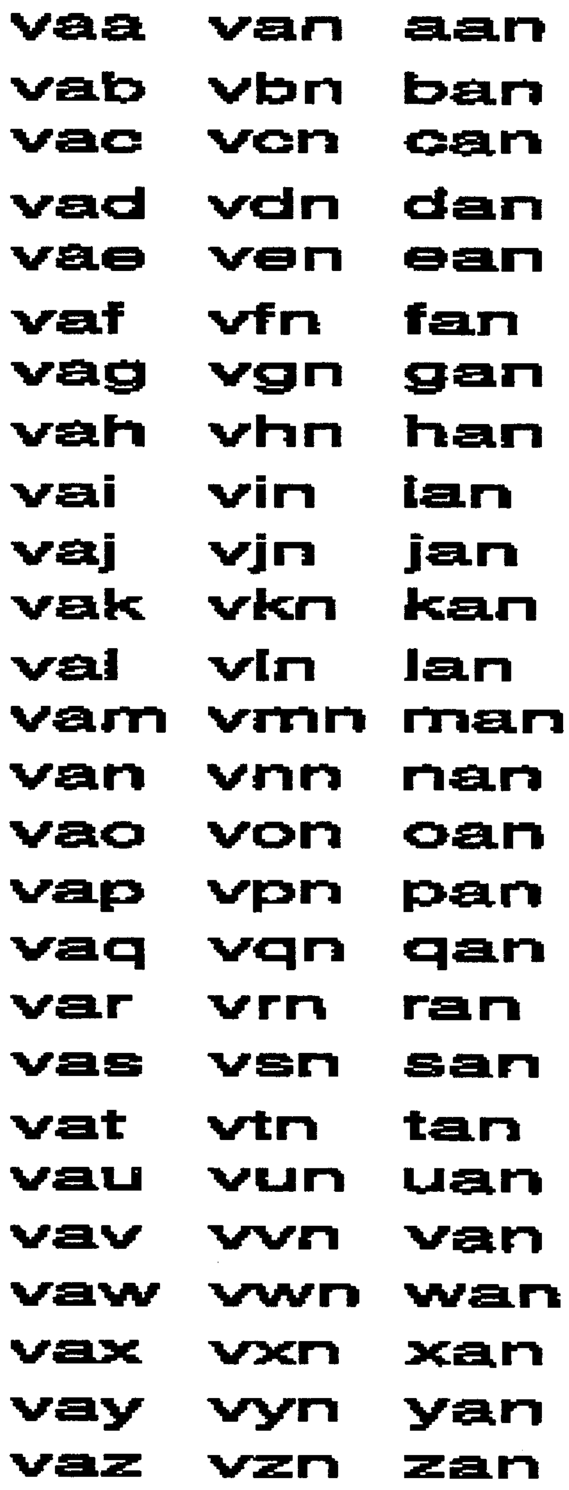
The similar words in 10 point, an untrained size, were processed into STIR vectors and selection value(SV) were calculated. The SVDs of the instances of 'van' character sequence did indeed show low values. The ROC curve is shown in Figures 11 and 10 respectively. Two sets of words were used the "Waldo Words", those that were close to VAN and the "Placebo Words", those consisting of variations of VAN. These words are shown in Figures 6,7. These words were constructed from faxed images. One can readily see the fax corruption.

The 2-D autocorrelation of VAN is shown in Figure 5. The 2-D scale transforms of two unique quadrants of the 2-D autocorrelation are shown in Figure 8. The STIR vector produced by concatenating the rows of the 2-D scale transforms is shown in Figure 9.

It was assumed that an effective word segmentation algorithm had been applied to the bitmapped page. This is not a trivial problem, but some fairly straightforward software was developed to accomplish this task quite well. Sophisticated methods are available which should be considered in a fully developed system[21–23]. In order to increase the difficulty of classification the word set used with VAN and VAX was constructed such that the other words were very close to VAN and VAX, differing only by a single letter. Results were quite good. VAN appeared three times in a series of 78 words. The algorithm found VAN each time (3 Hits) and mistakenly identified another word as VAN one time (1 False Alarm) for the threshold shown. In order to assess the method more fully, the threshold was varied over a range of values to generate a Receiver Operating Characteristic (ROC) shown in Figure 11.

5 Discussion

The examples provided show the potential for application of the STIR and noise subspace discrimination methods to character recognition. A selection value threshold could be added to reject symbols which are not among the valid set of characters. Simulations involving variations such as spotting words trained on multiple fonts and added shot noise show that the method degrades gracefully. Classification errors increase proportionally with amount of speckle. We are investigating a method based on moments which performs very well in speckle noise[24]. In addition, the detection method might be improved considerably. Some impressive results have been obtained by Warke and Orsak[25] in classifying faces using an information theoretic method.



vaa	van	aan
vab	vbn	ban
vac	vcn	can
vad	vdn	dan
vae	ven	ean
vaf	vfn	fan
vag	vgn	gan
vah	vhn	han
vai	vin	ian
vaj	vjn	jan
vak	vkn	kan
val	vln	lan
vam	vmn	man
van	vnn	nan
vao	von	oan
vap	vpn	pan
vaq	vqn	qan
var	vrn	ran
vas	vsn	san
vat	vtn	tan
vau	vun	uan
vav	vvn	van
vaw	vwn	wan
vax	vxn	xan
vay	vyn	yan
vaz	vzn	zan

Figure 6: Waldo Words in 10pt Representation

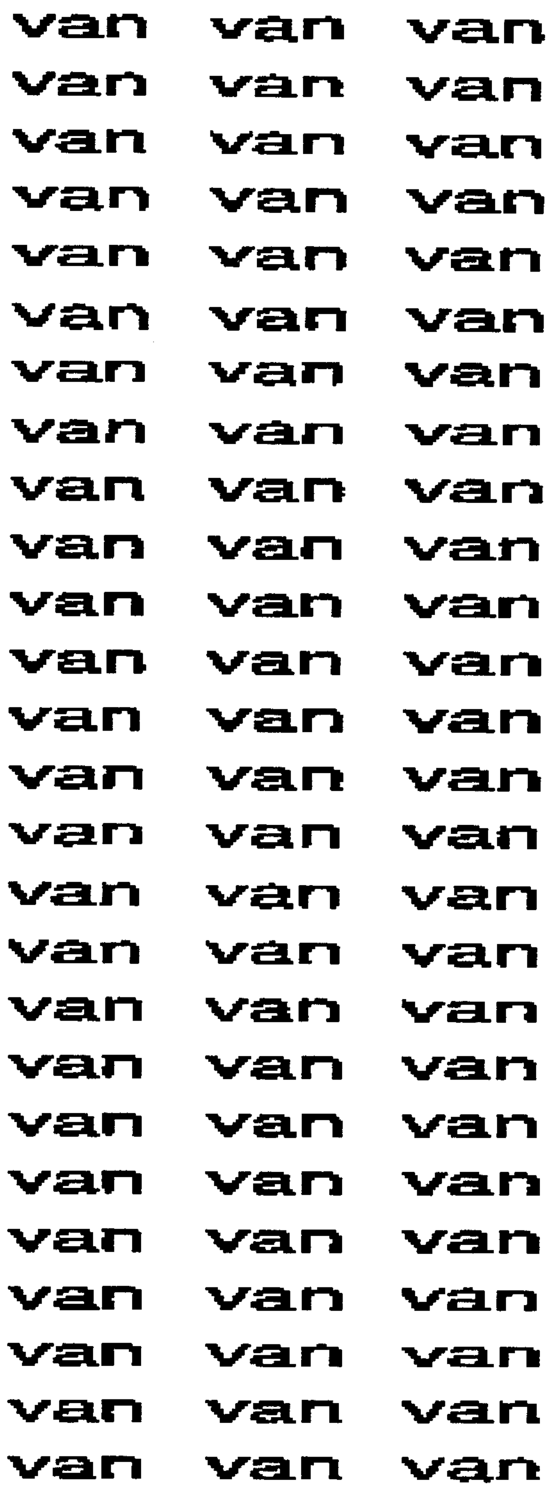


Figure 7: Placebo Words in 10pt Representation

5.1 Conclusions

The methods described in this paper provide a very robust means of identifying target words in a bitmapped document. The various parts of the algorithm are not unique in and of themselves. The scale transform (or Mellin transform) is an obvious way of removing scale effects. Autocorrelation has been used in many applications to dispense with absolute time or displacement. The noise subspace concept is used in many modern methods for high resolution spectral analysis. However, taken together, in the form suggested in this paper, yields a powerful new method for character and word spotting in bitmapped documents. The method is very robust and performs well even under a considerable amount of fax or speckle noise corruption. There are many points of refinement which remain to be investigated. Each step in the process offers considerable opportunity for refinement.

These methods need not be confined to character and word spotting. Any bitmapped representation may be brought into this methodology. Logos, Chinese characters, images of ships and planes faces, sounds (via time-frequency representations) and many other types of signals and images might be profitably identified using this approach. The methods may also be readily extended to higher dimensional spatial-temporal-frequency-wavenumber representions wherein complex objects may be identified. A major problem in doing this is the size of the representations produced. However, with rapidly increasing RAM allocations in computers, this too may be possible.

Acknowledgments

We gratefully acknowledge the support of this research via ONR Grant N000014-97-1-0072 and DOD Contract no. MDA904-95-C-2157. Thanks also to Ronit Reger, Ingbert Schmidt and Randy West for preparing the faxed bitmaps used in this work.

References

[1] V. F. Pisarenko. The retrieval of harmonics from a covariance function. *Geophys. J. Royal Astron. Soc.*, 33:347–366, 1973.

[2] S. Mori, C. Y. Suen, and K. Yamamoto. Historical review of OCR research and development. *Proc. IEEE*, 80:1029–1092, 1982.

[3] O. E. Agazzi and S. Kuo. Pseudo two-dimensional hidden Markov models for document recognition. *AT&T Technical Journal*, 72:60–72, Sept./Oct. 1993.

[4] S. Kuo and O. E. Agazzi. Keyword spotting in poorly printed documents using pseudo 2-D hidden Markov models. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 16(8):842–848, August 1994.

[5] C. Yen and S. Kuo. Degraded gray-scale text recognition using pseudo–2D hidden Markov models and N-best hypotheses. *Computer Speech and Language*, 9:381–405, 1995.

[6] J. Jeong and W. J. Williams. Reduced interference time-frequency distributions. *IEEE Trans. on Signal Processing*, 40(2):402–412, February 1992.

[7] W. J. Williams and J. Jeong. Reduced interference time-frequency distributions. In B. Boashash, editor, *Time-Frequency Signal Analysis: Methods and Applications*, chapter 3, pages 74–98. Longman and Cheshire, 1992.

[8] L. Cohen. Time-frequency distributions – a review. *Proc. IEEE*, 77(7):941–981, July 1989.

[9] L. Cohen. *Time-Frequency Analysis.* Prentice Hall, Englewood Cliffs, NJ, 1995.

[10] W. J. Williams. Reduced interference distributions: biological applications and interpretations. *IEEE Proceedings*, 84(9):1264–1280, September 1996.

[11] G. Cristobal, J. Bescos, J. Santamaria, and J. Montes. Wigner distribution representation of digital images. *Pattern Recognition Letters*, 69:215–221, 1987.

[12] L. Jacobson and H. Wechsler. A paradigm for invariant object recognition of brightness, optical flow and binocular disparity images. *Pattern Recognition Letters*, 1:61–68, 1982.

[13] L. Jacobson and H. Wechsler. A theory for invariant object recognition in the frontoparallel plane. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 6(3):325–331, 1984.

[14] L. Jacobson and H. Wechsler. Derivation of optical flow using a spatiotemporal-frequency approach. *Computer Vision, Graphics and Image Processing*, 38:29–65, 1987.

[15] W. J. Williams, E. J. Zalubas, R. M. Nickel, and A. O. Hero III. Scale and translation invariant methods for enhanced time-frequency pattern recognition. *Multidimensional Signal Processing*.

[16] L. Cohen. The scale representation. *IEEE Trans. on Signal Processing*, 41(12):3275–3292, December 1993.

[17] E. J. Zalubas and W. J. Williams. Discrete scale transform for signal analysis. In *Proc. of the IEEE Int. Conf. on Acoust., Speech, and Signal Processing*, volume 3, pages 1557–1561, 1995.

[18] W. J. Williams. Separating desired image and signal invariant components from extraneous variations. *SPIE: Advanced Signal Processing Algorithms, Architectures and Implimentations*, 2846:262–272, August 1996.

[19] P. E. Zwicke and Jr. I. Kiss. A new implementation of the mellin transform and its application to radar classification of ships. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 5(2):191–199, March 1983.

[20] N. Marinovich, L. Cohen, S. Umesh, and D. Nelson. Scale-invariant speech analysis via a joint time-frequency-scale processing. *Proc. Int. Soc. Opt. Eng.*, 2569:522–537, 1995.

[21] K. Etemad, D. Doermann, and R. Chellappa. Page segmentation using decision integration and wavelet packets. In *Proc. of the 12th IAPR International conference on Pattern Recognition*, volume 2, pages 345–349, 1994.

[22] K. Etemad, R. Chellappa, and D. Doermann. Document page segmentation by integrating distributed soft decisions. In *Proc. of the IEEE International conference on Neural Networks*, volume 6, pages 4022–4027, 1994.

[23] T. Reed and H. Wechsler. Spatial/spatial-frequency representations for image segmentation and grouping. *Image and Vision Computing*, 9(3):175–193, 1991.

[24] J. C. O'Neill, A. O. Hero, and W. J. Williams. Word spotting via spatial point processes. In *IEEE International Conference on Image Processing*, pages 217–220, 1996.

[25] N. Warke and G. C. Orsak. An information theoretic methodology for noisy image classification with application to face recognition. In *Proc. Conf. on Information Science and Systems*, Princeton, NJ, 1996.

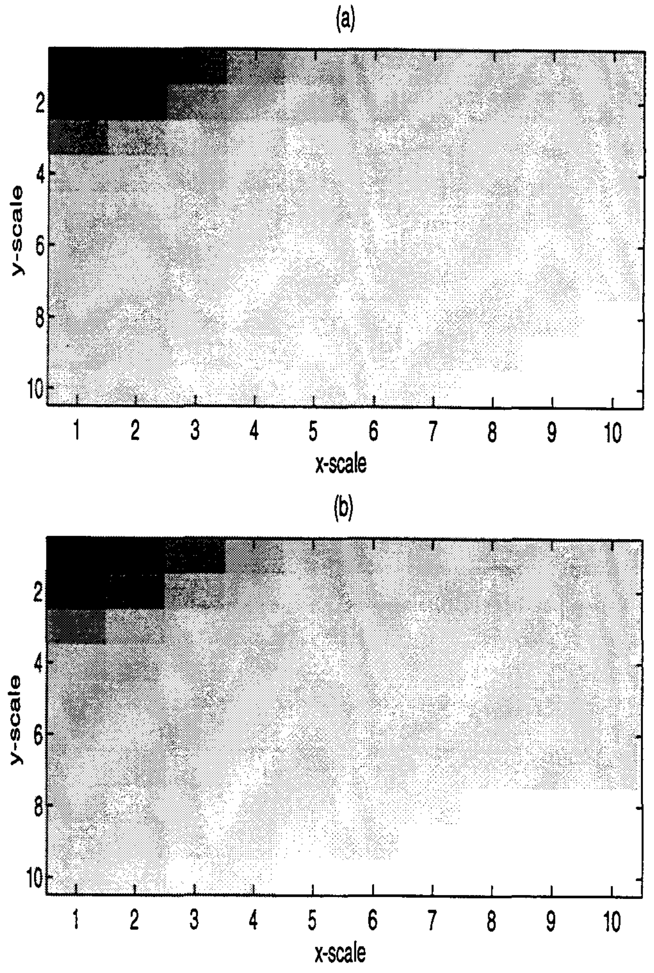


Figure 8: 2-D Scale Transforms

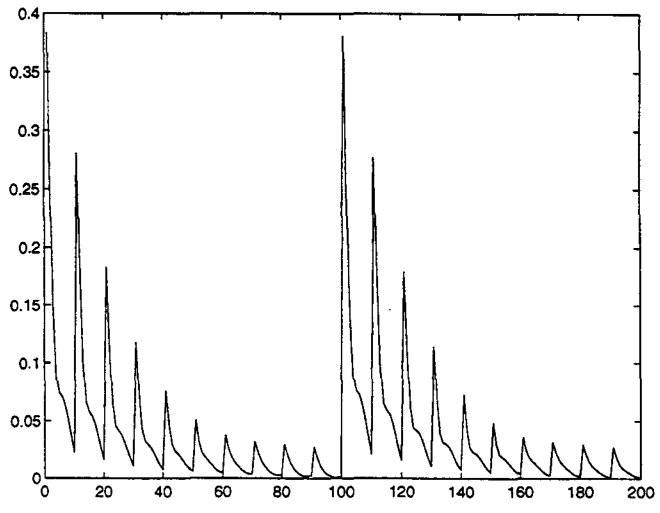


Figure 9: The STIR Vector for VAN

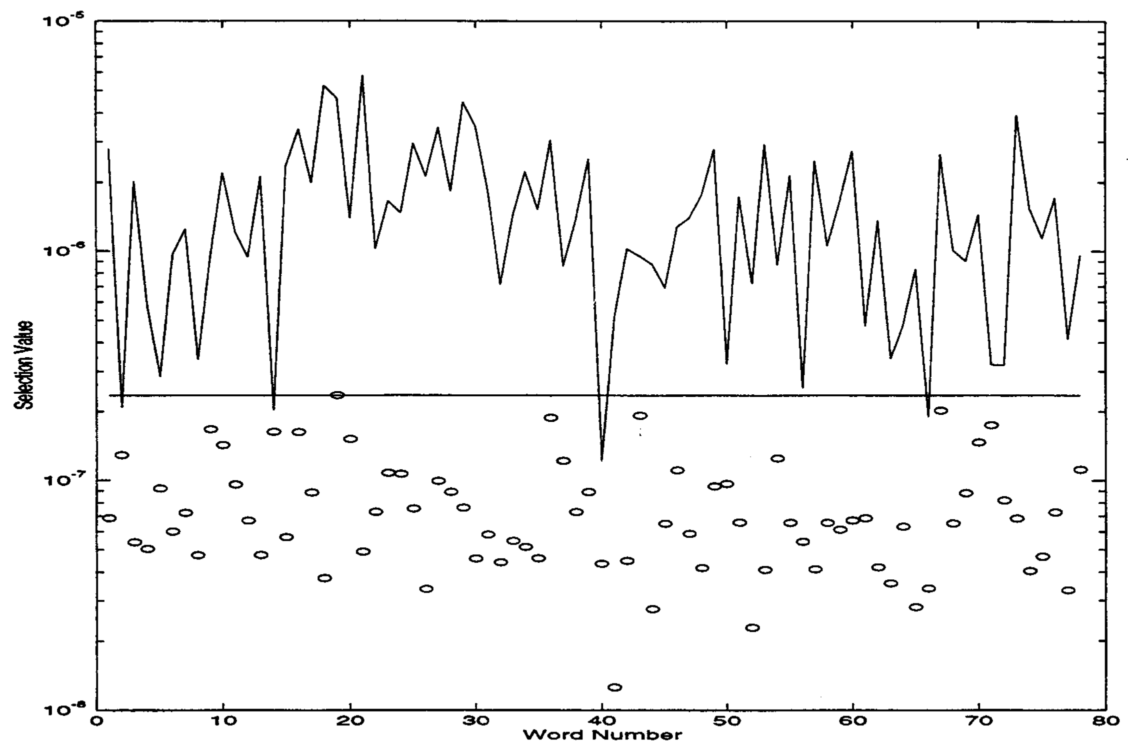


Figure 10: Selection value results. The upper plot represents the Waldo word selection values. The circles represent the Placebo word results. The horizontal line represents the threshold chosen

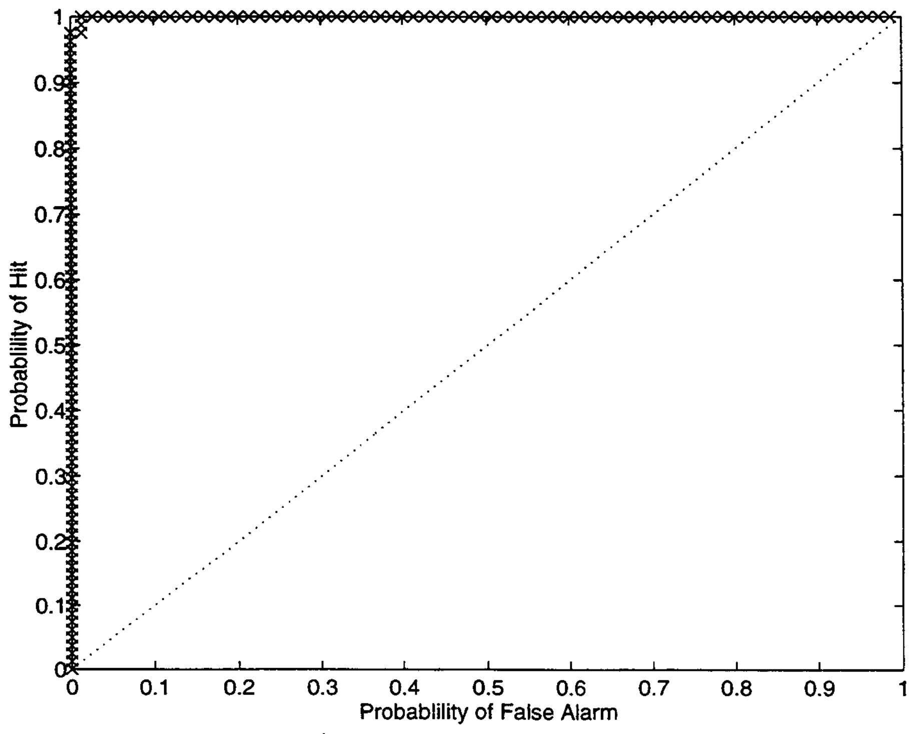


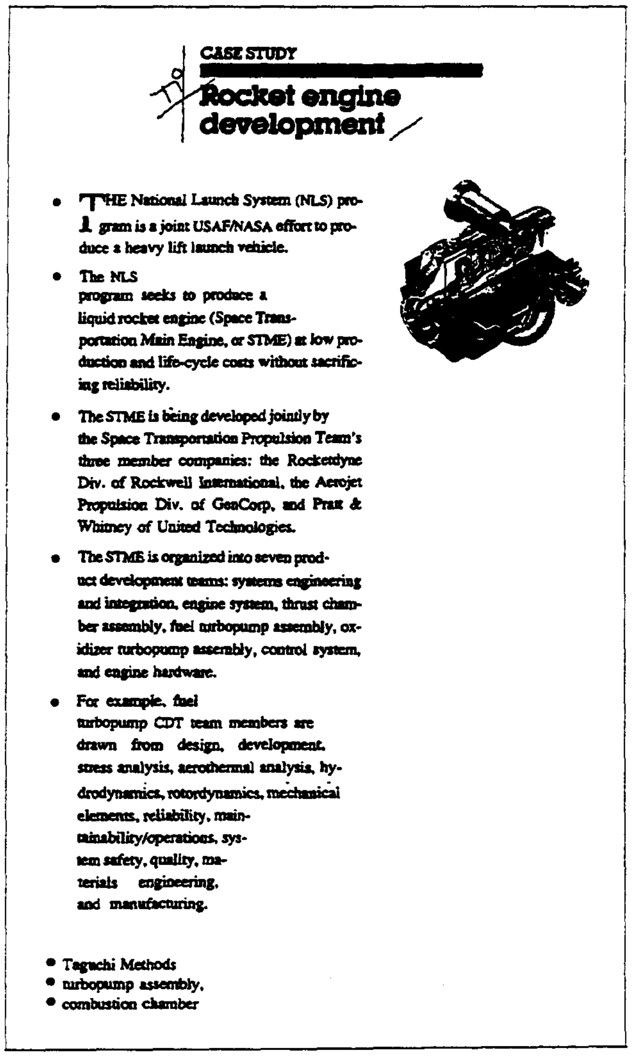
Figure 11: ROC results.The word results are x's. The dotted line represents chance performance

Image-based Document Summarization

Francine R. Chen **Dan S. Bloomberg**
Xerox Palo Alto Research Center
3333 Coyote Hill Rd.
Palo Alto, CA 94304
{fchen, bloomberg}@parc.xerox.com

We have developed a system, DIMSUM, for performing Document IMage SUMmarization. Our approach to this task is to select extracts from imaged documents for composition into a summary. Similar to work by others on ASCII text (*e.g.,* [5] [4]), our approach to image-based summarization does not rely on language understanding or generation. In contrast to the *text*-based techniques developed by other researchers, sentences and keyphrases for creating a summary are automatically selected from an *imaged* document without recognition of the characters in each word. The summary sentences and keyphrases are then combined with other extracts, such as graphics, halftones, and/or a table of contents derived from the headings to compose the final summary.

A summary of an imaged document can include extracted information unavailable in simple text documents, such as figures and graphics, that together communicate a sense of the document. An example of a collection of automatically selected summarizing extracts for a three page imaged document is shown in Figure 1. The original document[1] is shown in Figure 2. The summary page contains the first identified heading, usually the title of the document, a set of summary sentences, keyphrases, and one or more figures. The set of summary sentences are presented as bulleted items to give the reader the sense that the sentences should be regarded as highlights, rather than a coherent paragraph. The set of bulleted keyphrases tend to be complementary to the sentences. The selected image(s) are extracted from the document and scaled to fit on the page. The sentence excerpts are produced for presentation by extracting the regions of each page image which contain the summary sentences. Because a sentence is extracted from the image line by line, and then assembled without regard to relative location of the lines, characters that span several lines, as in the first



CASE STUDY

Rocket engine development

- THE National Launch System (NLS) program is a joint USAF/NASA effort to produce a heavy lift launch vehicle.
- The NLS program seeks to produce a liquid rocket engine (Space Transportation Main Engine, or STME) at low production and life-cycle costs without sacrificing reliability.
- The STME is being developed jointly by the Space Transportation Propulsion Team's three member companies: the Rocketdyne Div. of Rockwell International, the Aerojet Propulsion Div. of GenCorp, and Pratt & Whitney of United Technologies.
- The STME is organized into seven product development teams: systems engineering and integration, engine system, thrust chamber assembly, fuel turbopump assembly, oxidizer turbopump assembly, control system, and engine hardware.
- For example, fuel turbopump CDT team members are drawn from design, development, stress analysis, aerothermal analysis, hydrodynamics, rotordynamics, mechanical elements, reliability, maintainability/operations, system safety, quality, materials engineering, and manufacturing.

- Taguchi Methods
- turbopump assembly,
- combustion chamber

Figure 1: Sample collection of summarizing extracts. Summary sentences, keyphrases, and selected images are displayed.

[1]K.N. Butler, Rocket engine development, *Aerospace America*, pp. 28-30, April 1993.

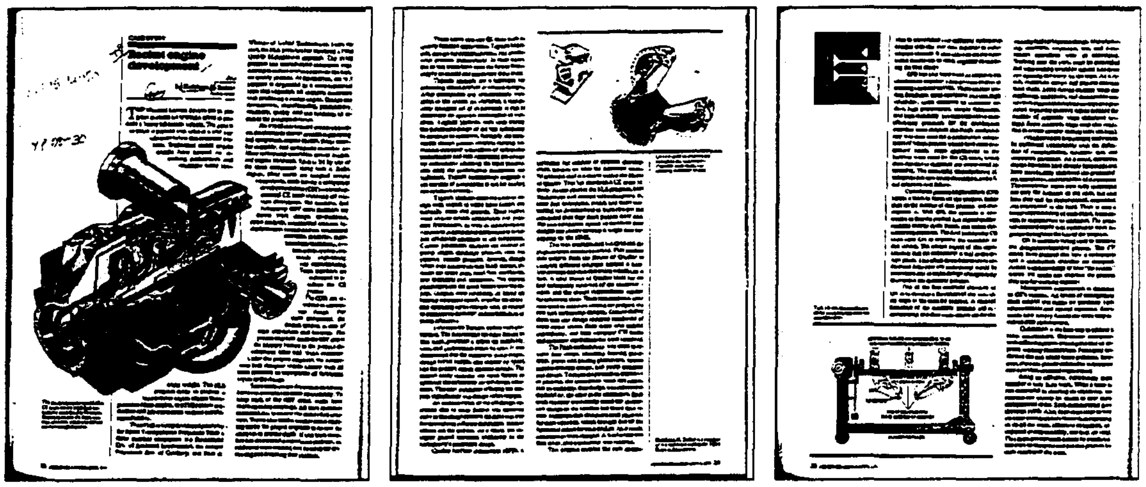


Figure 2: Three page document summarized in Figure 1

sentence, are broken in presentation. Currently, no reformatting of the text is performed, although such reformatting could be done to produce aligned text. The keyphrase excerpts are created by extracting a representative of each selected phrase from the page images.

In selecting regions for extraction, there are a number of acceptable excerpts, as well as some unacceptable excerpts. We have taken the approach that the highest priority should be to avoid selecting an unacceptable excerpt, followed by selection of the best of the acceptable excerpts. Another design guideline in our work is to identify computable solutions. This leads to the use of approximations to an ideal solution, but can result in faster performance. In empirical tests over 11 documents, we have observed that performing optical character recognition (OCR) takes approximately an order of magnitude longer than performing segmentation and identification of equivalence classes in a document. In either case, the processing time for selection of excerpts is negligible as compared to image analysis.

Referring to Figure 3, we outline the steps in performing document image summarization. Details may be found in [1] and [2]. To identify a set of summarizing excerpts, halftones, graphics, and text are first identified in the imaged document. Word boxes in the predominant font are extracted from the text regions.

Computation of sentence features requires estimation of term frequencies and identification of *stop* and *content* words. In text-based systems, character sequences are compared both to identify words that are the same *term*, that is composed of the same character sequences, and to identify stop words. But without OCR, the characters composing a word are unknown. Instead, in image-based summarization, word-box *equivalence classes*, that is words that are instances of the same term, are identified based on shape similarity. From the word-box equivalence classes, term frequencies can be estimated. Next, sentence and paragraph boundaries are located. Using sentence locations and the statistical characteristics of the word equivalence classes in each document, stop words are identified. All words that are not identified as stop words are considered to be content words. Sentence features are computed for each sentence in the document from the content words and the sentence and paragraph boundaries.

Summary sentence selection is performed using a statistical classifier to determine the likelihood of each sentence in a document being a summary sentence. The sentences most likely to be included in a summary are then selected for extraction. This method parallels the approach to text summary sentence selection described in Kupiec *et al.* [4]. A training corpus containing *indicative* abstracts by professional abstractors was used; thus the summary sentences tend to be indicative in nature, although some are *informative* [6]. Keyphrases are identified as sequences of frequently occurring content words. Our method of keyphrase identification has similarities to the method of text-based keyphrase identification by Justeson and Katz [3]. However, our method is simpler in that it relies on statistical characteristics only; part-of-speech tagging is not performed. The first heading is identified from the text blocks not in the predominant font. The largest halftone image and the largest lineart image, when present, are selected for presentation. Finally, the sentence and keyphrase excerpts are composed with the selected heading and images to create a summary.

In closing, a summary page for a five page article[2]

[2] J.R. Koelsch, Turning Pointe, *Manufacturing Engineering*, March 1993, pp. 59-63.

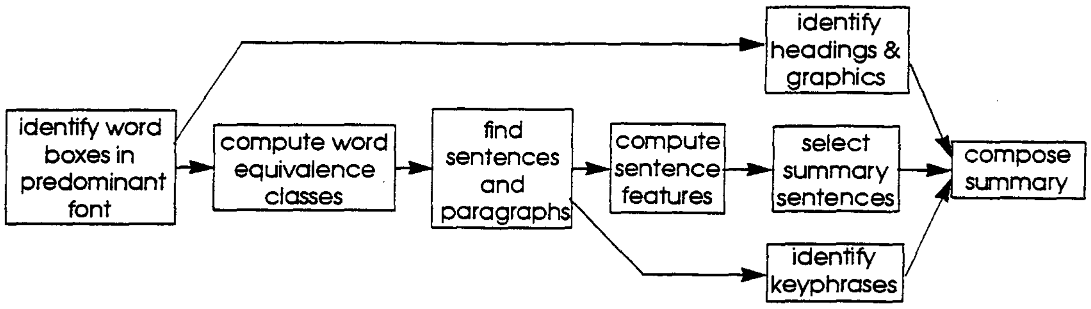
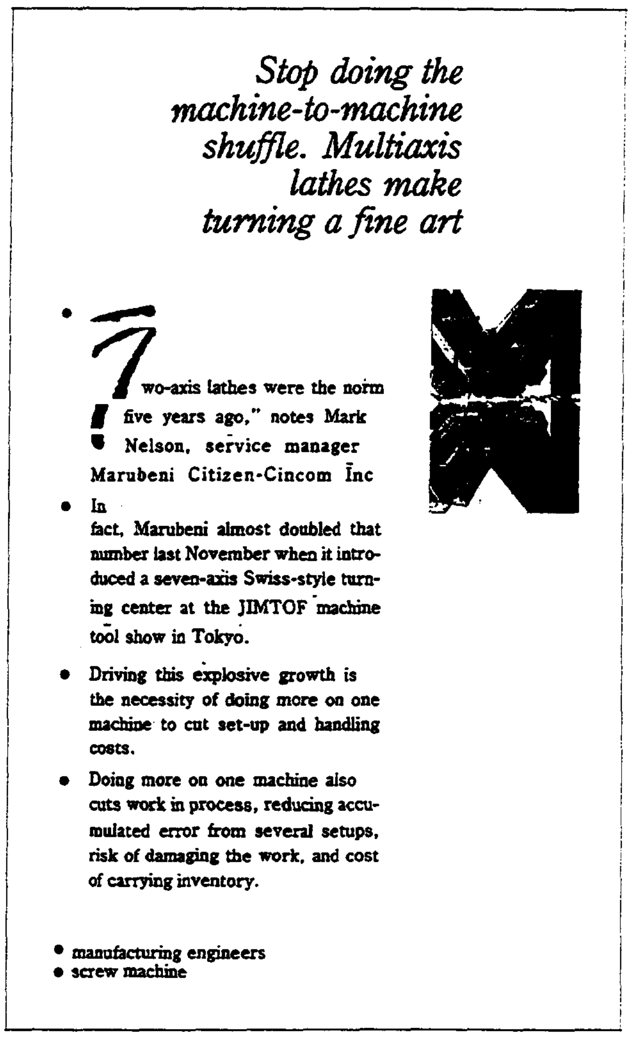

identify word boxes in predominant font
compute word equivalence classes
find sentences and paragraphs
compute sentence features
select summary sentences
identify headings & graphics
identify keyphrases
compose summary

Figure 3: Document image summarization system.



Stop doing the machine-to-machine shuffle. Multiaxis lathes make turning a fine art

- "Two-axis lathes were the norm five years ago," notes Mark Nelson, service manager Marubeni Citizen-Cincom Inc
- In fact, Marubeni almost doubled that number last November when it introduced a seven-axis Swiss-style turning center at the JIMTOF machine tool show in Tokyo.
- Driving this explosive growth is the necessity of doing more on one machine to cut set-up and handling costs.
- Doing more on one machine also cuts work in process, reducing accumulated error from several setups, risk of damaging the work, and cost of carrying inventory.

- manufacturing engineers
- screw machine

Figure 4: A summary page for a 5 page article with a script title.

is shown in Figure 4. In this article, the title is in script, rather than printed and was identified as line art. However, the text in the largest font provides concise summary information. Creating a summary from an imaged document permits a richer collection of information in the summary than is available from the text alone, and can be done more quickly than first performing OCR and then summarizing.

References

[1] D.S. Bloomberg and F.R. Chen, Extraction of text-related features for condensing image documents, in *SPIE Conf. 2660, Document Recognition III*, San Jose, CA, Jan 29-30, 1996, pp. 72-88.

[2] F.R. Chen and D.S. Bloomberg, Extraction of thematically relevant text from images, in Proceedings of the Fifth Annual Symposium on Document Analysis and Information Retrieval, Las Vegas, NV, pp. 163-178, April 1996.

[3] J.S. Justeson and S.M. Katz, Technical terminology: some linguistic properties and an algorithm for identification in text, *Natural Language Engineering*, **1**, 1995, pp. 9-27.

[4] J. Kupiec, J. Pedersen and F. Chen, A trainable document summarizer, in *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, Seattle, WA, pp. 68-73, 1995.

[5] H.P. Luhn, The automatic creation of literature abstracts, *IBM J. Res. Develop.*, vol. 2, pp. 159-165, 1959.

[6] C. Paice, Constructing literature abstracts by computer: Techniques and prospects, *Information Processing and Management*, vol. 26, pp. 171-186, 1990.

Media Processing Research at Maryland

David Doermann
Laboratory for Language and Media Processing
University of Maryland, College Park, MD 20742
doermann@cfar.umd.edu
http://documents.cfar.umd.edu/LAMP

1 Introduction

The Laboratory for Language and Media Processing at the University of Maryland has a concentrated effort in document image understanding and video processing. The laboratory was formed in the fall of 1996 to bring researchers from Linguistics together with researchers who have expertise in document, image and video analysis. Research is being carried out in a number of key areas including:

- Language Processing
  1. Automating the Acquisition of the Lexicon for NLU and MT
  2. Automating the Development of Broad Coverage Parsers and Lexicons for NLU and MT
- Document Analysis
  1. Document Page Segmentation
  2. Document Page Decomposition and Classification
  3. Document Image Compression
  4. Document Image Retrieval
  5. Document Image Databases
- Video Analysis
  1. Video Segmentation
  2. Video Browsing, Indexing and Retrieval
  3. Analysis of Human Actions

In this text, we provide an overview of several recent document related projects of possible interest to the community. Additional information is a available, in published papers, from our home page or upon request.

The support of this research by the Department of Defense under contract MDA 9049-6C-1250 is gratefully acknowledged. Researchers in LAMP involved with these projects include Omid Kia, Christian Shin, Huiping Li, Katherine Guo, Azriel Rosenfeld, and Ehud Rivlin.

2 Document Image Retrieval

Although the concept of a document image database is attractive, a comprehensive solution which does not require complete conversion to a machine-readable form continues to be elusive for practical systems. Since complete conversion is not always possible, we believe that a working solution lies somewhere in the continuum which exists between conversion- (or recognition-) based approaches and what can be considered image-based approaches. Relevant work has appeared on the problems of retrieving OCR degraded text ([28]), searching for keywords using image properties ([5–7, 43]), matching document image ([17]), indexing using the output of a document analysis system ([16, 47]) querying graphical documents ([14, 26, 30]) and querying image collections using captions ([45, 46]).

In this text we describe the general framework for IDIR, a system for Intelligent Document Image Retrieval. Our discussion of research problems will be limited to retrieval of document images, including feature extraction and querying. In Section 2.1 we describe the feature extraction capabilities and highlight some of the database issues. In Section 2.2 we describe the text-based query language and graphical query interface, and depict several query options. In Section 2.3 we present some preliminary results from our work with query by example.

A full discussion of the systems issues and architecture is available in the full paper [12].

2.1 The Document Database

In order for the document images to be utilized by different components of the document management system (transmission, storage, retrieval, organization and OCR), IDIR operates on both the image and the output of the document image analysis processes. After analysis, the aim is to present the document in a controlled form, where the actual image data (pixels) are linked with the resulting representations in a database. To represent the document efficiently both the physical and logical components of the document's structure must be made available [40] (see Figure 1).

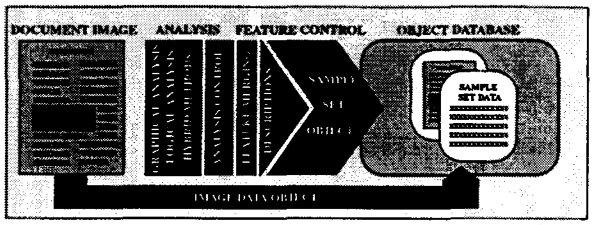


Figure 1: The construction of a document image representation in the database.

2.1.1 Feature Extraction

Feature extraction modules can make effective use of the results of image and signal processing and document analysis (physical and logical). The success of queries by the IDIR system depends greatly on the quality of the description that is available from the images in the database. The results of feature extraction (calculated results of low-, mid- and high-level features) at multiple levels are exploited most efficiently when they are combined to present higher abstractions of document characteristics. These can then be utilized by a query language that offers flexible means to define document content using its syntax to retrieve the desired document.

Texture Features - characterized by a non-uniform or varying spatial distribution of intensity [13].

Global Geometric Features - locally calculated black to white ratio, statistical distributions of geometrical properties, and geometrical measures attached to pixel clusters

Local Component Features - low-level features such as texture and color, functional features such as component type (list, table, or drawing) or logical features which are class-dependent, such as names, titles and abstracts.

Structural Features - logical and physical layout.

Content Features - via a template library based approach to extracting keywords from a "compressed image" as described by Kia [21, 22].

2.2 Document Image Querying and Retrieval

It is clear that the most natural way to query a general image database is to use features that might otherwise be used to verbally describe the image instance. Fqor example, names of objects, colors, positions, actions, etc. In documents the same principle holds true. In this section we describe the basic architecture for incorporating query capabilities and demonstrate a simple matching technique for queries by example. We also describe a GUI interface which can be used to form spatial queries.

2.2.1 Query Capabilities

The basic syntax of the SQL-like query mechanism is as follows:

```
find <attributes>
from <objects>
with <query_conditions>
     [group by <object_type>]
     [order by <attribute>
           [<sort_order>]];
```

where `<attributes>` defines our search criteria and `<objects>` defines the space in which we are looking.

Query by Document Structure

Query by document structure will allow us to formulate queries based on both the existence of particular structural features in the document as well as relations between them. For fixed attributes such as the document type, location of text and graphic blocks and font size, the database is populated with the output of the document image analysis system. A query for retrieving all journal article pages with at least one table and one image may appear as follows:

```
find p.page_image
from document d, page p, block b
with
  d.document_type = 'journal_article' and
  p.document_id = d.id and
  b.content_type = 'image' and has_a(p,b) and
  b.content_type = 'table' and has_a(p,b);
```

Similarly, we can specify specific configurations of elements using spatial functions including

```
direction_from(<object>,<object>)
distance_from(<object>,<object>[,<distance_unit>])
column(<object>,<number>,<from_direction>)
```

which are more easily described with the graphical query interface described in Section 2.2.2.

Query by Text Content

Query by text content can be implemented in a number of ways and there sexists a wide spectrum of possible representations which can be used. At one level, we may consider OCR with a "fuzzy retrieval" algorithm on the resulting text.

The function

```
keywords_in(<object>,<keywords>)
```

will return documents which contain the supplied keywords.

We are considering a shape-based method which was first proposed by Tanaka and Torii [48] and then refined and demonstrated for information retrieval by Spitz [43], although the query language is independent of the choice of representation and technique. Currently, the system provides exact and partial matching of keywords with text using GLIMPSE.

Query by Document Example

We have found that in the document domain, we may consider defining similarity both content-wise and structurally. We will assume a Boolean model for keyword similarity as defined above (it only matches if the keyword exists in the image) and concentrate on structural similarity. Ideally, the similarity measure will have several components tunable by the user, including similarity with respect to the types of components present, their attributes (e.g. font size or line spacing for text), their relative locations (allowing for figures and blocks of different lengths) and their absolute locations. To date we have only implemented global exact match and weighted similarity measures. Structural similarity is defined as follows:

Structural Similarity

The structural similarity of two documents can be approximated using a measure of their constituent regions and their types (text, graphics or image). For each region R_i in the query image Q_i, we match R_i to each region of the database image D_j of the same type and overlapping it. If there is no region of overlap, the region R_i is mapped to NULL. We therefore have a directional graph from each query image region to a possibly empty subset of the database regions.

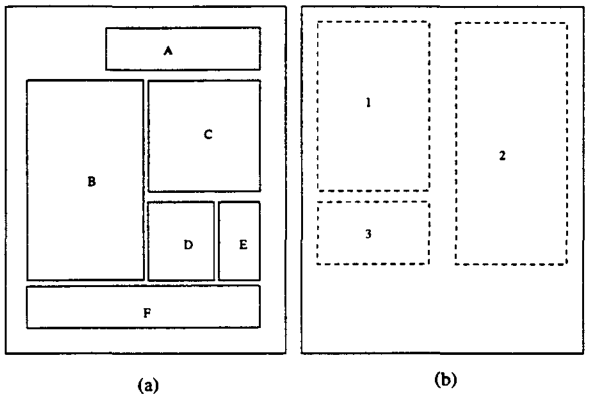
Once this first correspondence has been established, an evaluation mechanism is used to refine and measure the quality of the match. It is clearly possible for a single region in the query image to be mapped to multiple regions in the database image and vice versa. There are several situations where such a mapping is not desired, and must be refined.

The first restriction is that no region should be mapped to two or more regions in the horizontal direction. This would occur, for example, if a page with a single block of text were mapped to a page with two columns of text. Splitting a block horizontally may occur between paragraphs, for example, but vertical splitting is typically an intentional structural occurrence. For query regions which map to more than one corresponding database region, a subset of regions which have maximal intersection but do not neighbor horizontally, is chosen and the remaining regions are removed from the mapping. For query image regions which overlap a single database region, the correspondence is trivial (but we must later consider a symmetric case where multiple query regions correspond to a single database region).

Once this condition is satisfied for all query regions, the symmetric case, where a single database region corresponds to multiple query regions, is still possible. Using the restricted mapping, a reverse mapping is constructed from the database image to the query image and the condition is checked again. Regions which violate the vertical split are again evaluated and the subset of maximal overlap is kept.

We can, for simplicity, demonstrate this using a set of known text regions found in Figure 2. The initial match produces a graph with the following relations:

```
A -> 1,2   B -> 1,3
```



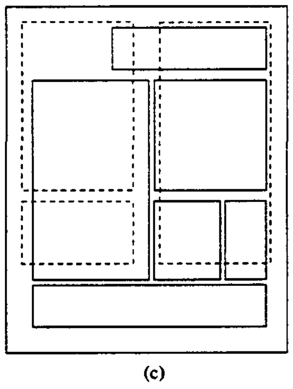


Figure 2: Example of matching: (a) the query structure, (b) the candidate structure, and (c) overlay of the two structures.

```
C -> 2     D -> 2
E -> 2     F -> NULL
```

Since A maps to two regions in the database image which neighbor horizontally, the maximal subset is chosen which does not overlap in the horizontal direction. The maximum is defined on the area of overlap, and corresponds to region 2 in this example (1 is removed from the mapping). All of the other mappings are acceptable. Computing the inverse match, we obtain

```
1 -> A,B
2 -> A,C,D,E
3 -> B
```

Again, we find that 2 violates the horizontal mapping rule and the maximal subset, corresponding to A, C, D, is chosen and preserved.

Once the best match is found, the percentage of each region in the query image which matches the database image is computed, and the total is summed for all regions. Regions which remain unmatched in the model image are accumulated separately. Since documents tend to have relatively small numbers of regions, an exhaustive search can be performed to establish the correspondence, and the regions ranked by similarity.

Various other mechanisms for querying are being explored including querying by graphic content. In

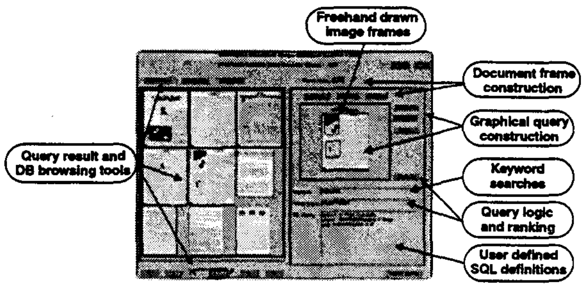

Freehand drawn image frames
Document frame construction
Graphical query construction
Keyword searches
Query logic and ranking
User defined SQL definitions
Query result and DB browsing tools

Figure 3: The IDIR query interface.

particular, some work has been done at the University of Maryland on a mechanism for querying maps by content [41].

2.2.2 Graphical Query Interface

Figure 3 shows the graphical query interface (GQI) which can be used to simplify the creation of spatial queries for document images. The main window shows a document instance which serves as a frame for spatial layout. Any number of frames can be included with the query precedence being used to set the logical relations between frames. A single frame is intended to be applied at the page level, but for non-spatially relevant queries, it can be applied at the document level—for example, a query which requests a document in which a given set of regions exists.

A set of query objects (or blocks) can be placed on the document with spatial relevance either on or off. A set of type-dependent attributes is associated with each block to set—for example, font size for text regions, or number of columns for a table. Attributes can be edited by selecting the attributes button on each block. On each frame, the scope of the query (i.e. restriction on the document type) and the logical relation between any two blocks in the frame are set. Spatial relations between any two blocks are set by selecting a relation link between any corresponding pair of blocks.

The spatial query is translated on the fly and appears in the query translation window. This SQL-like query is run against the database, and the results returned. The query results can appear in two ways. For images, a set of thumbnails is displayed, and for data, the raw text results appear. A query results history function saves the recent queries and can be refined by dropping the query onto the new frame button.

2.3 Query by Example Results

2.3.1 Local Features

A first set of experiments was performed using a limited set of features which have previously been applied to scene image retrieval[20]. Low-level global features including texture orientation, gray level difference histogram, and color features of the image were extracted when the image was inserted into the

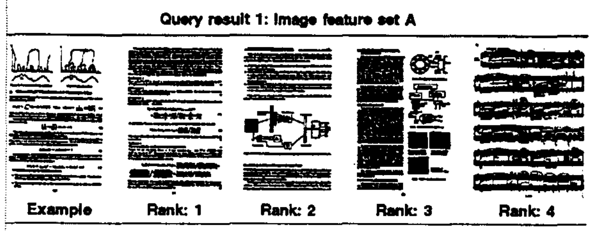


Figure 4: Results: Global measures.

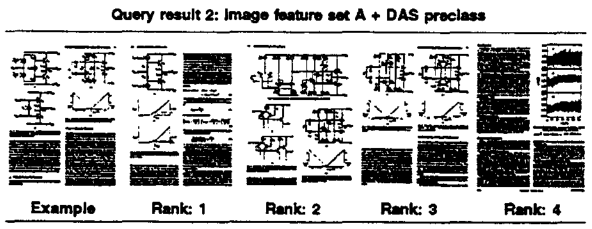


Figure 5: Results: Local measures based on page segmentation.

database. The features are localized pixel-level measures which can be attached to page segmentation results [39] These features have undergone preliminary tests with the sample document images and produce quite specific document-content-related information.

Images are ranked by their degrees of similarity with the user-specified query by calculating differences between them using the Euclidean distance for scalar features and the G statistic [36] for distribution features. Simple Boolean operators can be used to merge queries, prior to ranking.

The document image query using the features described above was tested in two different test settings. We used a document image test database with over 250 images of various types. The first test used only global image features without document analysis results. An example of a query result is shown in Figure 4.

The second test was to use the same features with document segmentation results computed a priori. An example of a query result is shown in Figure 5.

The image analysis features perform well with scene images, achieving a fairly high retrieval accuracy ratio (85-98%). When using only the image analysis features, the similarity of the document images is approximately the same as with the scene images (judged by human perception), but this is still undesirable since document images are typically fairly similar to begin with (horizontal text, similar size pages, etc) As one can see from the result, the use of document properties is essential to achieve similarity in the document structure (physical and logical). In the second set of results (Figure 5) the global structure (i.e. number of columns, size of text, etc.) and content level similarity is apparent. By using the two feature domains, the document retrieval performance is substantially enhanced and more ac-

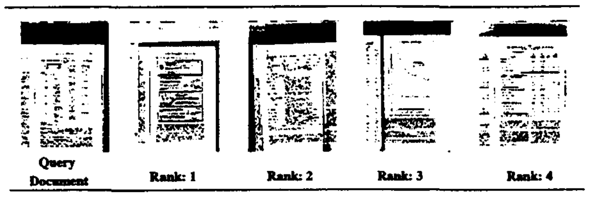


Figure 6: Results for querying with a graphics page.

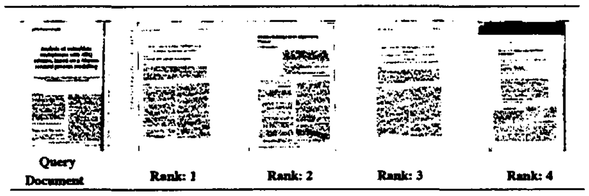


Figure 7: Results for querying with a title page.

curate results can be achieved.

2.4 Structure

A second set of experiments was performed independently using the algorithms presented in Section 2.2.1 on features extracted from the segmented page such as locations and sizes of blocks and their spatial relationships. In addition, content features such as content type, text point size and font style for each spatially matching block were used to refine the matching score. The query capabilities were tested using 293 images from the University of Washington image database.

First, the size of the content body of each document was used to normalize (scale and translate) each image a priori. As described in Section 2.2.1, the similarity between two document images is measured by computing the total content area overlap between regions of the same type. When there exists an area overlap between blocks of the same content type, content features are also used to weight the structural similarity.

For text regions, our algorithm reduces the matching score by 90% for each conflict in either font size and font style. The texture measure from the previous set of experiments can also be used as a content descriptor to scale the matching score. As a result, both structural and content features contribute to the final similarity measure between two document images.

Figures 6 and 7 show an example page taken from the database and the top four matches. Figure 6 contains a query image with a large "graphic" component and several smaller text regions. Figure 7 shows a title-page query characterized by large text blocks spanning the top of the page, and body text at the bottom. We are currently developing ground truth so that we can run precision and recall experiments on the entire database, and we are also extending the matching scheme to consider "nearby" regions for matching.

3 Duplicate Document Image Detection

In this project, we propose and implement a method for detecting duplicate documents in very large image databases. The method is based on a robust "signature" extracted from each document image which is used to index into a table of previously processed documents. The approach has a number of advantages over OCR or other recognition based methods including speed and robustness to imaging distortions.

To justify the approach and test the scalability, we have developed a simulator which allows us to change parameters of the system and examine performance for millions of document signatures. A complete system is implemented and tested on a test collection of technical articles and memos.

3.1 Introduction

Consider the situation where thousands of documents are being imaged and added to a database, possibly from a distributed environment. If multiple instances of the same document exist, they may be re-entered into the database unnecessarily. This may not be desirable for a number of reasons, including increased storage cost, difficulties in maintaining database integrity, increased processing costs for database operations and cost of indexing multiple images with the same underlying content.

We are currently addressing only the problem of detecting duplicate image-variant documents, documents where multiple instances of an effectively identical original source are possibly copied, written on, or otherwise degraded, and then scanned for inclusion into the database. At a physical level, pages may be missing, a cover may have been added, or notes may have been written on the pages. The document may have been copied repeatedly, so different-generation copies are involved. At the image level, the document may have been scanned at different times and on different devices, so resolution, illumination, and contrast are also issues. Similarly, skew and translation of the page may add additional distortion. The goal of this project is to track documents which are being added to a database, so that when variations of an existing document are presented, the system is able to identify the duplicates, and not process them further.

We provide a brief overview of the problem and our approach in Sections 3.2 and 3.3 and its feasibility in Section 3.4. In Section 3.5 we describe the results from a simulator which allows us to test the indexing mechanisms with millions of indices and in Section 3.6 we provide details of the implementation.

3.2 Problem Overview

Depending on the information available in the database system, when a document arrives for processing, the problem of duplicate detection can be approached in a number of ways. If, for example, basic index information such as the document number, date, title, authors or number of pages is entered manually prior to scanning, this information could serve as a preliminary filter for duplicates. In most cases, however, high-volume operations prohibit this manual entry prior to scanning. Instead, we would like to identify duplicates from their images prior to any manual entry.

A second possible solution to consider is to apply Optical Character Recognition (OCR) to the document image and match as much text as possible between the documents. Although the matching can be done relatively quickly, OCR performance suffers on degraded documents both in terms of accuracy and speed. For this reason, we do not feel that OCR is a feasible first level filter, but it may be used as a secondary filter, to reduce the possible matches from several hundred to tens of documents.

The overall goal of a duplicate detection system should be either to 1) determine that the document is not a duplicate or 2) to accurately identify 10-20 documents which could contain a duplicate. In the second case, other methods of analysis such as OCR, structural analysis or a human verifier, can be used as post-processors to eliminate non-duplicates.

Our preliminary experiments have allowed us to draw some very general conclusions about the types of algorithms that will and will not work. First, algorithms which attempt to do any in-depth analysis of the document (such as OCR or structural analysis) are not ideal because they cannot extract the features (recognized characters in the case of OCR) robustly enough for degraded documents and the index information would be too large. Second, after features have been extracted, any indexing scheme which requires the comparison of the extracted features to a significant portion of the database will fail because of the computational requirements.

We have developed an approach which promises to fulfill a majority of the above criteria. The approach is based on a surprisingly robust characterization using shape codes of textual components. In the next section, we describe the approach, some experiments we have run to show its effectiveness, and research issues which must be addressed to develop a working system.

3.3 Basic Approach

Our approach is based on the conversion of a representative line of text in a document image to a *signature* using a shape coding technique used by a number of authors including Tanaka [48], Kia [24] and Spitz [43] for related document analysis applications. Shape coding attempts to label symbols in a line of text based on very simple shape properties, such as whether they are ascenders, descenders, limited to the x-line, multi-component, or punctuation, for example. These properties are much more robust to noise then features necessary for OCR, and can be extracted fairly rapidly.

To extract the signature, the document is scanned

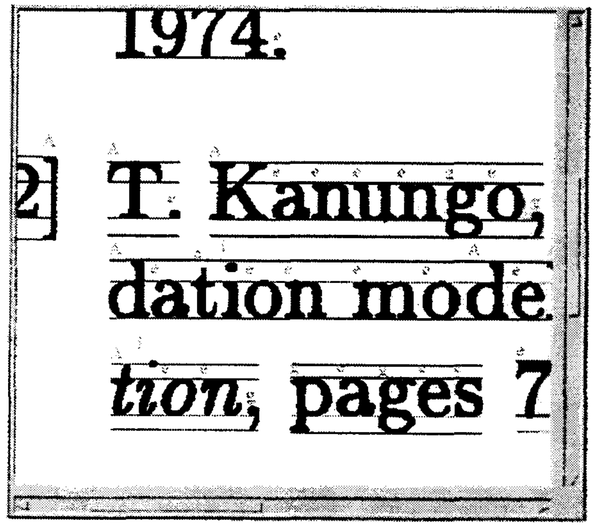


Figure 8: Sample character shape code assignment

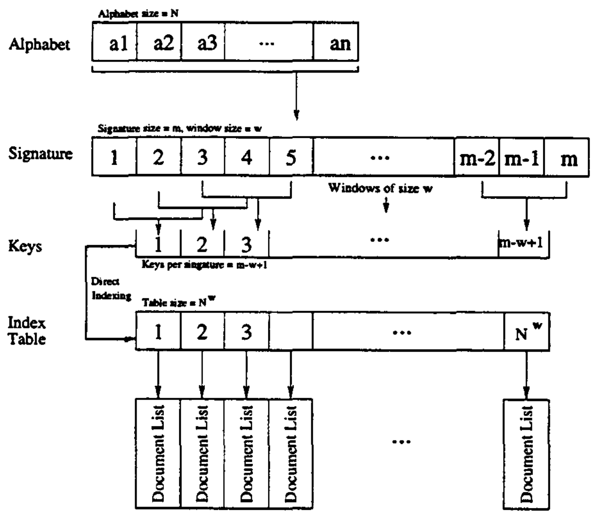


Figure 9: Overview of indexing scheme.

for a representative sample of text, typically on the order of 30 symbols, on a single line or across several lines. For this text, the base-line, x-line, ascender-line and descender-line are identified, and each character component assigned an appropriate shape code as shown in Figure 8.

The string of shape codes is then a signature for the document, and is used to index into a large table of previously processed documents. A second level of robustness is added by indexing based on n-grams of the signature, rather then attempting to use a line index based on the entire string. Each of the shape coded n-grams is extracted from a sliding window of size w across the signature and each serves as an index *key* into the database. A single dropped or inserted code will affect a small number of keys and will not affect the entire signature. Figure 9 shows the relationship between the signature, its keys, and the database.

Clearly, a number of additional issues must be addressed in a complete system to satisfy criteria set forth above. These include:

- the use of global classifiers - number of pages, page component statistics, etc. as first-level filters to reduce the duplicate search space.
- the choice of a shape code alphabet - selection of the features to incorporate into the document-specific signature which provide maximum discrimination.
- the extraction of features - how to identify the signature in the image of the document.
- the database organization and indexing - how to create efficient ways to index into large collections.
- verification of duplicate candidates

All of these issues are addressed in the design phase.

3.4 Feasibility Analysis

Before implementing and testing our approach on real data, we performed a theoretical analysis to see if the design is realistic and if it is robust to anticipated errors in the signature extraction. The analysis was performed assuming we have extracted a candidate signature. The parameters which can be controlled include system dependent variables such as file size limitations of the operating system, disk access time and disk transfer rates; algorithm dependent variables such as the number of documents, the size of the index table and the average size of the documents; and independent variables such as the size of the signature alphabet, the size of the signature and the n-gram window size.

The analysis was carried out for indexing and performance of the system, and resulted in the ability to produce a qualitative estimate of the expected size of the database, the computational requirements for matching signatures and the number of missed and false duplicate detections as a function of database size. Overall, it showed that the system could be implemented with basic hardware.

The details of the analysis and a complete discussion of the findings can be found in [9].

3.5 Coding and Indexing Simulator

To develop and demonstrate the feasibility of our approach to shape coding we have set up several experiments using ideal and corrupted shape code data. A simulator was developed which allows us to explore a number of different coding, indexing and database organization scenarios, without specifically addressing the feature extraction issues. The goal is to be able to show that signatures can be obtained which are unique enough to be used for indexing and that the database of indexes can be created and will scale appropriately.

To do this, the simulator takes as input *ideal* ASCII text and provides a mechanism to map the incoming characters deterministically into appropriate

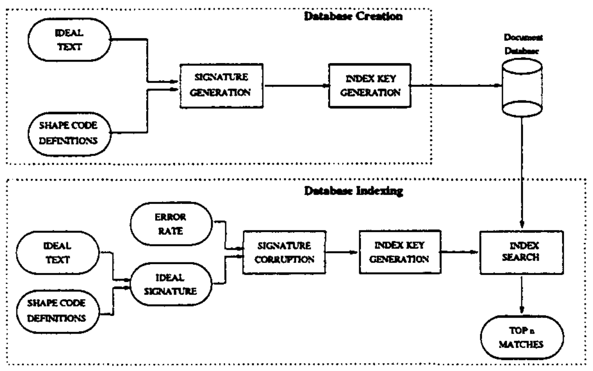


Figure 10: Simulator overview

shape codes, thus simulating perfect feature extraction (see Figure 10). From the resulting signature, we can explore various indexing options, and test the uniqueness of the shape coding on large databases. Since text databases are widely available, a large-scale system can be simulated at relatively low cost, with the primary remaining issue being the robustness of the feature extraction.

To address robustness, we have built a noise or degradation model into the system by corrupting the ideal signature, thus simulating errors in shape coding. This is accomplished by randomly perturbing a fixed number of symbols in the signature.

Added Errors	Top 1	Top 2	Top 5	Top 10	Top 20
0†	2416	2443	2458	2465	2467
5	2379	2419	2447	2457	2463
8	2059	2202	2327	2390	2431
10	1403	1678	1905	2039	2181

Table 1: Number of duplicate candidates detected in 2500 queries from a pool of 1 million documents.

Several experiments where were conducted on both corrupt and uncorrupt data. Overall, we have shown that a large percentage of the documents can be identified. Table 1 shows the results of the number of duplicate documents detected which appear in the top 1, 2, 5, 10, and 20 matches. Figure 11 shows the percentage of duplicates (recall[1]) as a function of the number of documents retrieved. We can see that even in the case of 10 errors, for a signature size of 50 symbols, we can detect duplicates more than 86 percent of the time.

[1]Recall can be defined as the number of relevant documents which are retrieved divided by the number of relevant documents in the entire database. In this case recall is the number of duplicates identified, divided but the total number which are considered duplicates.

†: Some of the lines in the original database occur more than 20 times, so the "true" duplicate may not appear in the top 20. That is why for the 0 error case we did not get 2500 lines all detected.

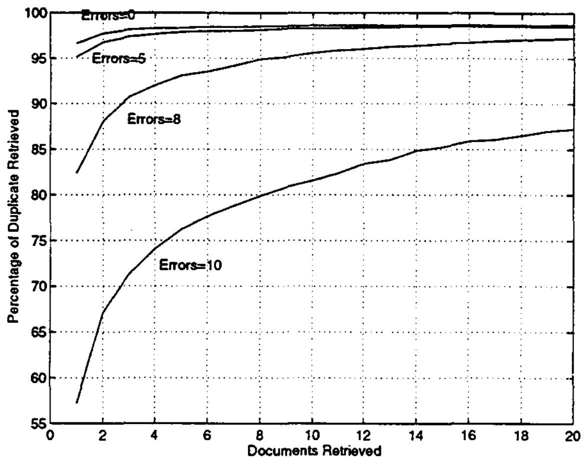


Figure 11: Percentage of duplicate documents identified in the top n candidates

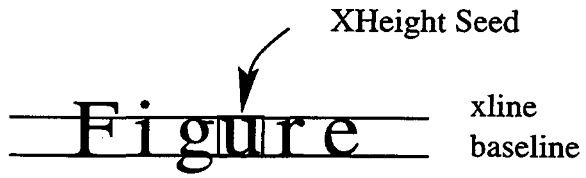


Figure 12: Example xheight hypothesis, xline and baseline

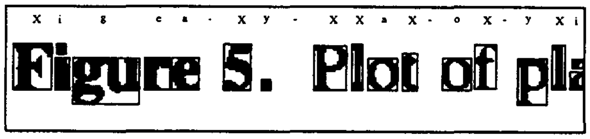


Figure 13: Shape coding results for part of a line.

3.6 Implementation

Having tested the signature matching capabilities, the remaining task was to implement and test the line extraction and signature coding. With degraded documents, the most sensitive part of the system is the ability to extract the same representative line from multiple instances.

The representative line is extracted from each image as a text line which has a sufficient number of characters (i.e. > 50) to be used as a signature. For efficiency we divide the page image horizontally into thin "zones" and perform the analysis on these zones in order from top to bottom. When a representative line is found, we do not have to process any portion of the image below it.

We first use a single pass connected component algorithm to identify components within the zone of interest. To deal robustly with noise, we attempt to use a filter to eliminate components which result from copier noise and graphics. By using a conservative threshold on the components size, we eliminate components whose actual size is less than 7pts or greater than 14pts. Although we may loose some punctuation and other small symbols, they should be lost in both the original and candidate documents.

Next, the symbols are grouped into words using a smearing distance which is a function of the average character width. A second filter is then used on the height and width of words, eliminating words what would not likely contribute to a unique signature.

Using a second smearing, we group words into lines. In order to avoid problems with running heads, we skip the first two valid reference lines, and consider the third as the signature for the page. If this is done consistently, it will provide us with a meaningful signature.

Once a valid reference line is found, the next step is to extract the signature. Distortion of several sorts may cause difficulties at the line level, so we extract the shape code signatures at the word level. Although we can assume that the image has been de-skewed to less than a degree, a word level shape code procedure is more accurate.

We begin by roughly classifying the characters in a word into three groups by height – small symbols (like punctuation), medium size symbols (such as 'a' or 'c') and large symbols (ascenders, descenders, parentheses, etc). We then choose a medium size symbol from the middle of the line as an xheight seed hypothesis, using the bottom position of the symbol as the baseline and the top as the xline (See Figure 12). From this seed symbol, we encode the line outward in both directions. When classifying the neighboring character, if it is a punctuation symbol (i.e. does not span the region between the baseline and the xline, or extends into both the ascender and descender region), we simply code it appropriately and do not adjust the x or baselines. If the symbol is an ascender, we classify it and adjust the baseline to the bottom of the symbol, if it is a descender we classify it and adjust the xline to the top of the symbol, and if the symbol covers on the xheight region, we adjust both the xline and baseline.

After the word is classified, holes are identified and the classes are refined. Figure 13 shows an example coded.

3.6.1 Accuracy

The shape coding is very accurate for clean data which is not skewed significantly. The one problem which is present is that when many characters start to touch, the system may skip an otherwise valid line. This tends to occur when documents are photocopied repeatedly. For example, when the original is printed with a laser printer, then we attempt to process a third generation photocopy as a duplicate candidate, enough symbols touch to put us below the limit of signature length. As shown in the previous section, even the introduction of 10-15 shape code errors, we will still detect the duplicate the duplicate, but only if we can identify the line.

3.7 Experiments

To experiment with real data, we used images from the University of Washington Document Image Databases [34]. The data consisted primarily of technical journal pages scanned from first generation photocopies of the original documents, and memos scanned directly from the originals.

In the initial experiments, 1035 documents were added to a database. As each candidate document was added, the top 20 database matches were returned, along with their similarity scores. As expected, the average similarity for these unique documents was a low 20%, and the average difference between the similarity of the top matched document and the second second matched document was about 4%. This suggests that in general, no single document stood out as significantly more similar.

We then took 307 duplicate documents which were the result of third generation photocopies of the originals (also present on the UWASH CDROM). Some of the documents were significantly degraded. For these, the verification procedure was followed, without adding the documents to the database. As in the database construction, the top 20 database matches were returned for each candidate document, along with similarities. For the duplicates, the average similarity score for the top match was over 75% and average difference between the similarity of the top matched document and the second second matched document was over 48%. The average similarity for the second ranked document was about 20%, similar to that of the top ranked non-duplicate above. This suggests that when a duplicate was identified, it tended to be a significantly better match then the second closest match. Table 2 shows the distribution of rankings for the 307 duplicates tested.

All of the errors in the top matched document were due to a significant number of merged characters. We are currently attempting to use character width statistics to provide a symbol segmentation scheme which is not based entirely on white space. We anticipate even a naive segmentation will reduce such errors by as much as 75%.

Given that we can reduce the number of duplicate document candidates to a manageable number, a more refined algorithm which directly compares the two images can be used, or an operator can be rapidly presented with thumbnails of the top 20 candidates to verify that a duplicate exists. One important piece of information which can be taken into consideration is the ordering of the n-tuples. To index millions of documents it is much faster to consider them independently, but when the number of candidates is less then 20, we can consider comparing documents directly.

4 Document text image compression

In this section we describe a compression and representation scheme which exploits the component-level redundancy found within a document image. The approach identifies patterns which appear repeatedly, represents similar patterns with a single prototype, stores the location of pattern instances and codes the residuals between the prototypes and the pattern instances. Using a novel encoding scheme, we provide a representation which facilitates scalable lossy compression and progressive transmission, and supports document image analysis in the compressed domain. We motivate the approach, provide details of the encoding procedures, report compression results and describe a class of document image understanding tasks which operate on the compressed representation.

4.1 Introduction

Technological advances in processing, storage, and visualization have made it possible to maintain large numbers of documents in digital image form and make them accessible over networks. In order to do this effectively, three primary concerns must be addressed. The first is document size. An ASCII version of a document page can easily be stored in 2-3 KB, whereas a typical scanned page may result in an image which requires between 500 KB and 2 MB. A single book stored in image form can fill a CD-ROM to capacity. If we are to maintain documents in image form, an efficient compression scheme is essential for both storage and transmission.

The second concern is that of efficient access to the compressed images. Traditional compression techniques used for document images have been successful in reducing storage requirements but do not provide efficient access to the compressed data. It is desirable to use a compression method that makes use of a structured representation of the data, so that it not only allows for rapid transmission but also promotes compressed-domain processing and allows access to various document components in the compressed domain.

The third concern is that of readability. Most lossy compression and progressive transmission techniques use resolution reduction or texture-preserving methods that might render a document image unreadable. It is desirable that a document be readable even at the highest levels of lossy compression and at the start of a progressive transmission; it can then be augmented by subsequent information for better rendition. This is much preferable to a scenario in which the highest resolution is the only readable resolution. In this paper, we provide a single coding scheme which addresses all three of these issues.

4.2 Approach

Symbolic compression can be described in 3 parts: 1) segmentation and clustering, 2) residual coding and 3) data representation and compressed-domain processing.

4.2.1 Segmentation and clustering

The first step in our symbolic document image compression method is to find an initial set of patterns in the image which can be used to form a library. In the case of Latin text, performing a connected component analysis on the binary image provides a reasonable starting set. For connected scripts

	Top 1	Top 2	Top 5	Top 10	Top 20
Number of Documents	286	296	298	302	307
% of Total Doc Count	93.2	96.4	97.1	98.4	100.0

Table 2: Results for matching 307 duplicate document images against a database of approximately 1000 documents.

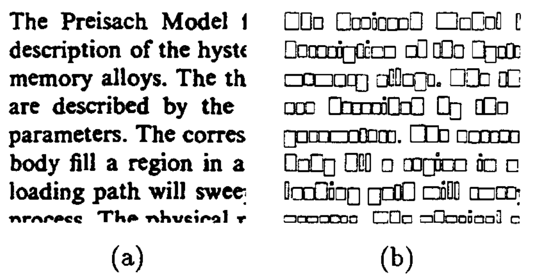

The Preisach Model 1
description of the hyste
memory alloys. The th
are described by the
parameters. The corres
body fill a region in a
loading path will swee
process The physical r

Figure 14: a) A sample document image; b) the bounding boxes of its connected components.

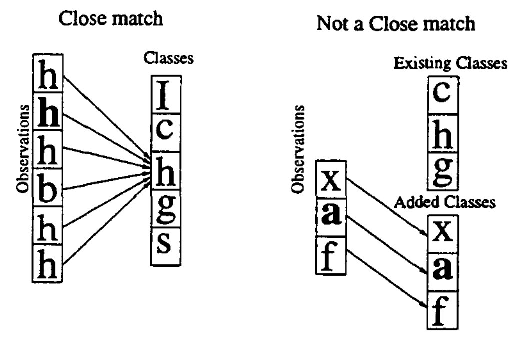


Figure 15: Clustering by pattern matching

or text in which the basic units are disconnected, more extensive segmentation would be necessary.

A small portion of a typical document is shown in Figure 14a, and the bounding boxes of the connected components are shown in Figure 14b. Because these patterns tend to appear repeatedly in the image, they form a basis for compression. In cases where multiple characters touch to form a single component, or where a single character is split into multiple components, representing them as a new component lowers the compression factor only slightly. If salt and pepper noise is present, it may give rise to small components; this will reduce compression but will have little or no effect on the readability of the document.

We treat each component as an observation and try to determine a best set of classes (clusters) of the components and to generate a prototype image for each class. We begin by comparing each observed component to all previously generated prototypes (Figure 15). If a match exists, we assign the observation to that class and refine the prototype. If no match is sufficiently close we regard the observation as defining a new class and take the observation as a prototype for that class. After all observations have been processed, the collection of prototypes typically looks like the one shown in Figure 16. The shapes shown in Figure 16 resemble English characters because of the primarily English content of the original document. For a document rich in mathematical symbols, some of the prototypes would resemble mathematical symbols, and similarly for non-Latin languages the prototypes would capture their symbol content. A number of matching algorithms can be used to group similar patterns into clusters, including simple XOR, weighted XOR, Boolean AND-NOT, and compression-based template matching.

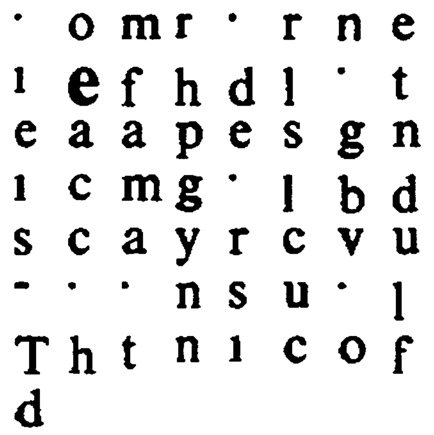


Figure 16: Typical cluster prototypes from a textually rich image

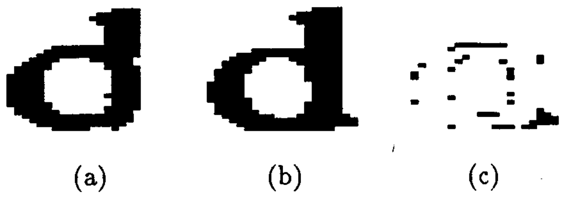


Figure 17: a) A component; b) a prototype; c) the residual map.

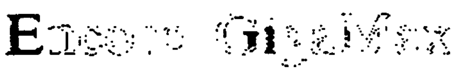


Figure 18: Example of ambiguous membership and resulting residual.

4.2.2 Residual coding

Each cluster of components is represented by a prototype. Depending on the sample space, there may exist some amount of variability in the clusters. For some clusters, the amount of this variation may be large enough that some of the components differ significantly from the prototype and extra information must be recorded to remedy this. A residual map, the difference between a given component and its prototype, is preserved and used to recover the components in a lossless form when necessary. Examples of a component, a prototype, and the corresponding residual map are shown in Figure 17. The replacement may give rise to three types of residuals. The first type is the most common variation from the prototype; it takes the form of a silhouette around the prototype, and usually does not effect the readability of the document, as seen in most of the residual maps of Figure 18 (an example also appears in Figure 17c). The second type results from ambiguity between two similar components. An example of this is shown in Figure 18 where the symbol 'C' was used as a prototype for the pattern 'G' in "Encore GigaMax". The third type arises when no appropriate clusters are available, as shown in Figure 18 for the letter 'E'. This case arises when the overhead of creating a cluster is higher than just tagging the component as a graphic entity and representing the entire component in the residual map. In a large document set, the number of instances of this case will be small, since each symbol will usually occur multiple times.

Motivated by image degradation models, we code the residue based on their contribution to the structure of the image component. Work done by Kanungo et al. [19], Nagy [33], and Sarkar [38] suggests document images degrade faster around the edges of characters. Since most degraded documents are still readable it suggests that pixels close to edges do not contribute to correct recognition and we extend to the idea that farther pixels contribute the most toward correct recognition of components. Ordering in distance along with structural prediction of pixels protruding outward from prototype component forms the basis of our residual coding.

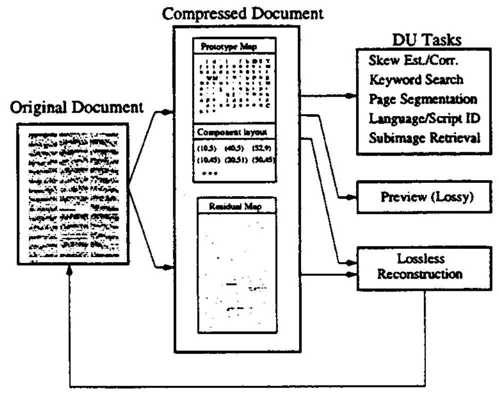


Figure 19: Representation of a compressed document image with relevant processing access.

4.2.3 Data representation and processing

In our work special attention is given to the ability to perform document analysis tasks in the compressed domain, without full decompression. In many situations only parts of the encoded information pertaining to the given task require decompression.

Although some work has been done on the processing of compressed document images [31, 42, 44] the outlook for performing such processing on standard pixel-based compressed representations does not appear promising. Our approach makes a strong case for the use of symbolic compression in document image coding. We use an indexable representation [23] composed of independent streams for the prototypes, the locations of symbol instances, and the residual maps. We will show that it is possible to code the dominant symbol shapes and their locations within the image using only about one percent of the original image data. Using only this encoding it is possible to implement a large number of common document analysis tasks [1, 29], Optical Character Recognition (OCR) [32], and layout analysis [35, 37].

Our contribution is in the development of a compression system that promotes compressed-domain analysis by allowing symbol access. Although the approach works best with clean images where multiple patterns repeat, it is flexible enough to adapt to situations where the components correspond to arbitrary patterns in the image as opposed to symbols, or where many symbols are mis-clustered. The overall encoding scheme is shown in Figure 19.

In order to provide spatial access to image components in the compressed domain, it is convenient to encode four types of information in four independent sections: the file header section, the prototype section, the symbolic section (see below), and the residual section. The file header contains general information which provides indexes to the prototype,

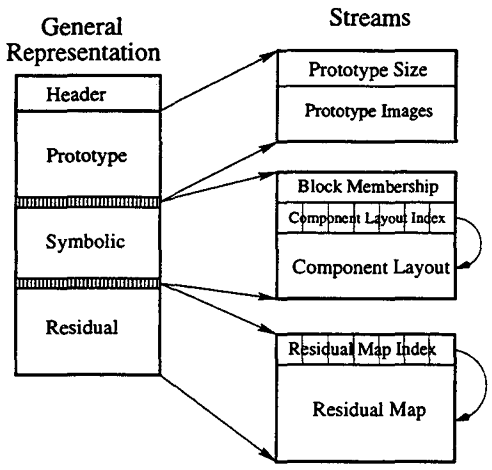


Figure 20: Data structure organization

symbolic, and residual sections and an index to the header section of the subsequent page in a multi-paged document. The prototype section is a collection of prototype bitmaps and their sizes. The symbolic section provides an encoding of the locations of components in the image and the prototype of which each component is an instance. The residual section contains the residuals produced after substituting the prototypes for the actual components. Each of these four sections is encoded as a set of streams. Figure 20 shows the organization of the streams.

Access to the components and use of the residual coding allow a number of desirable applications directly on the compressed image. Lossy compression can be achieved with an associated entropy rate and used to determine size of the information record. Progressive transmission is achieved by hierarchical residual coding. Skew estimation and correction can be performed by manipulating component locations, Sub-image retrieval can be performed with processing which is linearly dependent on the sub-image size, and keyword searching can be carried out by determining character shapes and comparing them to an input query.

5 Document Functionality

The purpose of a document is to facilitate the transfer of information from its author to its readers. It is the author's job to design the document so that the information it contains can be interpreted accurately and efficiently. To do this, the author can make use of a set of stylistic tools. In this work we introduce the concept of document functionality, which attempts to describe the roles of documents and their components in the process of transferring information. A functional description of a document provides insight into the type of the document, into its intended uses, and into strategies for automatic document interpretation and retrieval.

To demonstrate these ideas, we define a taxonomy of functional document components and show how functional descriptions can be used to reverse-engineer the intentions of the author, to navigate in document space, and to provide important contextual information to aid in interpretation.

5.1 Documents as Message Conveyors

Written documents have long been the preferred medium for the transfer of information across both time and space. In this sense, the general purpose or "function" of a document is to store data produced by a sender in a symbolic form to facilitate transfer to a receiver. Traditionally, the data takes the form of a set of markings on a page, with the sender corresponding to the "author", and the receiver to the "reader". We limit ourselves to the understanding and interpretation of these "traditional" 2D documents which the reader receives visually.

When documents are regarded as message conveyers, we can classify them according to the type of message that is conveyed. We will differentiate between three classes of messages: informational (report, dictionary, newspaper, novel, catalogue), instructional (recipe book, a do-it-yourself manual, roadsign), and identificational (a street sign, a car license plate, a name tag).

The types of messages describe above were formulated from the author's point of view. The reader, the receiver of the document, may have different goals, and may abstract the document's contents at many different levels. Readers can become quite skilled at abstracting task-dependent information from a document and using this information to establish a context for further interpretation. For example, when looking for documents created on a specific date, an experienced reader can rapidly locate the dates of documents such as business letters and forms without reading them entirely. If it is then decided to "read" the document, the context helps with its correct interpretation and provides a framework in which to proceed through it in an orderly fashion. We can distinguish three basic ways of doing this:

- **Reading** - which usually involves examining the document from beginning to end. This mode is ordinarily used for letters, articles, and many types of books. The examination may be more or less thorough, ranging from proofreading to skimming.
- **Browsing** - which involves examining only selected parts of the document to determine if more in-depth examination of these parts is required. This mode is ordinarily used for newspapers, magazines, and journals.
- **Searching** (or referencing) - which involves looking for a specific piece of information in the document. This mode is ordinarily used for reference books such as dictionaries, encyclopedias, directories, manuals, handbooks, catalogs, etc.

These modes of interaction with a document apply not only to text-intensive documents; they can also apply to documents which are primarily representational, such as maps and drawings. However, the processes used to read, browse, or search a document depend on the document type. For example, browsing a newspaper and browsing a map have the same basic goal of examining only selected parts, but the methods which are used to accomplish this are quite different. Similarly, searching a phone book and searching a map both require "navigating" and making decisions based on partial information, but they involve different processes. For phone books, one uses index terms and alphabetical relationships; for maps, one uses symbols or landmarks and spatial relationships.

A great deal of work has been done on the analysis of document structure. Almost all of this work, however, has involved models for specific classes of documents. We believe that significant progress in the automated analysis of general classes of documents depends on the development of a general framework for describing document structure. This paper attempts to develop a such a framework.

5.2 Document Structure

In this section, we first consider traditional views of document organization and show how a document's functional organization (i.e. organization in information transfer terms) is related to its geometric and semantic organizations (Section 5.2.1). We then illustrate how the author and the reader are able to use the design of a document to impose functional organization on the document (Section 5.2.2).

5.2.1 Levels of Document Organization

In document understanding, documents have traditionally been viewed according to their geometric and semantic organizations[2]. Both organizations have a common *content* which represents a base level of data (typically text, but also possibly including graphics or images). The content's *geometric* nature refers to how it is presented on the page (for example, typeface and font size, for text; line widths and symbols, for graphics), and its *semantic* nature refers to its meaning.

Similarly, a document has both geometric and semantic *structure*. The *layout* structure corresponds to the organization of the document into geometric groupings such as characters, lines, blocks, columns, etc. It describes the relationships among these components and the relationships of the individual components to the entire page. The *logical* structure, on the other hand, organizes the content according to the interpretation of the reader, and also provides global relationships such as reading order. The logical structure corresponds to the document's semantic or conceptual organization.

[2]This is the view taken in the ODA standard [18].

We claim that there is a level of document organization, which can be regarded as intermediate between the geometric and semantic levels, that relates to the efficiency with which the document transfers its information to the reader. We refer to this level as the *functional* level.

A document obeys conventions such as the use of an alphabet and a language common to the author and reader, and the use of standard presentation rules such as word and line spacing, punctuation, etc. As the information content of the document becomes more complex, these conventions may no longer be adequate for efficient information transfer. Appropriate structures can be used to enhance efficient transfer of information and reduce its ambiguity. For example, an author may use page or section headers to "summarize" content; ordered lists to enumerate or itemize information; separators to "punctuate"; attachments (such as footnotes and sidebars) to subordinate; tables or graphs to present numeric data; maps to present spatial data and their interrelationships. (Note that graphs and maps involve augmenting the basic language with more expressive constructs.) Figure 21 shows some examples of such structures.

As an illustration of the relationship between the geometric, functional, and semantic organizations of a document, consider a **block** of text at the top of a page. Its dimensions and location on the page, as well as properties of its components, are geometric or layout attributes. The fact that we have grouped the components together to form the block is based on geometric proximity. We can use the block's attributes (position, size, etc.) in a class-independent manner to conclude that the block is a **header**; this describes it functionally. If we make a class-dependent identification of the block as a **title**, we have given it a semantic description. Note that a similar block could be a running head or a letterhead in a different context.

The functional description of a document is often independent of document type and can be derived from geometric considerations. Headers, footers, lists, tables, and graphics are examples of generic structures which can be common to many types of documents. Such functional structures will be referred to as class-independent.

If the type of the document is known (for example, business letters or memos, forms, advertisements, or technical articles), a component can have functionality with respect to the documents of that type. For example, in a letter, functional components may include the sender, receiver, date, and salutation. Such functional components will be referred to as class-dependent. The formats used in documents of specific types, such as business letters or journal articles, also serve to enhance information transfer by helping to organize and prioritize the information.

Structure	Example	Use
header	centered	relative importance, focal point
list	enumerated itemized	conveys temporal sequence suggests similar level of descriptiveness
separator	white space or rule line	physical and possibly semantic dis-association
attachment	footnote boxed text sidebar	supplemental information under some semantic hierarchy
illustration	table figure	supplemental information - preserves 2D associations graphical representation of information

Figure 21: Some structures and their uses

5.2.2 Functional Document Design

Because the transfer of information to the reader of a document is done using vision as a medium, documents should be designed in accordance with basic perceptual principles such as the principles of Gestalt [25]. When we use white spaces as separators, the principle of proximity, which states that elements which are closer together tend to be grouped together, is being applied. According to this principle, the space between lines should be greater than the average space between words and letters. The principle of good continuation, according to which elements that lie along a common line or smooth curve are grouped together, causes the white spaces that border a column to be seen as units, thus separating the column from its neighbors. The principle of similarity, which states that elements that are similar in physical attributes, such as color, orientation, or size, are grouped together, causes words in boldface to group together.

The author of a document can take advantage of these principles to design the document so that the reader can use it effectively. Authors typically use combinations of layout and emphasis to convey an intended organization, or to assign priorities to specific components.

Within a document, structures such as those shown in Figure 21 can be used as aids in the organization of information. A list, for example, suggests a meaningful temporal or set relationship between its items. A figure and the corresponding caption are interpreted as an illustration of some concept or fact in the text. Higher-level constructs such as sections/subsections, columns, indices, or running heads aid in organizing a document at a more global level.

Other techniques can be used to attract (or suppress) a reader's attention. At a page level, an author can use headers and increase their point size, use all caps, and/or center them to make them more prominent. At a word or phrase level, the author can use bold-face or italic fonts in a similar way to draw attention. Text which is seen as unimportant can be put in "fine print" with opposite results.

5.3 Exploiting Function

In order to effectively process a document, most document image understanding systems rely on relatively specific information about a restricted domain in order to accurately model the expected document class(es). This allows the system to richly interpret the document, and extract detailed information about its content. For example, in the domain of business letters, a great deal of work has been done on both their structural and logical interpretation ([2], [3], [8], [27], [35], [50], [51]). Unfortunately, for less homogeneous environments this approach cannot be effectively applied. As the set or stream of documents becomes more diverse (both intra-class and inter-class), the formulation of models becomes more difficult. Functional interpretation of documents can greatly facilitate tasks associated with their classification and use. In the following paragraphs we give three examples of tasks which can be addressed by identifying functionally meaningful constructs in documents.

Use Classification: In Section 5.1, we identified three major ways in which a reader can use a document: reading, browsing, and searching. Documents designed for these purposes can be grossly characterized by the size and organization of their information units, which can be identified by repetitive patterns in the document. For example, reading documents such as journal articles tend to have a single read-order and large information units; browsing documents, such as newspapers or popular magazines, tend to have multiple head-body structures, since their designer's goal is to give the reader quick access to the contents with "handles"; and searching documents tend to have many small information units such as the entries in an index or phone book. An instructional document intended for modification by the reader, such as a form, is characterized by small, blank information units such as horizontal line segments or boxes (including small check boxes).

Type Classification:
Simple functional features such as head/body

pairs and the locations of handwritten regions allow us to distinguish between document types such as letters and memos. Using functional features, we can achieve a gross categorization of the documents in a database. Given a large heterogeneous database of documents, this allows us to provide groups of documents which are likely to contain some piece of requested information, even if we cannot provide the specific information. An experiment demonstrating this method of type classification will be described in Section 5.4.3.

Functional Enhancement: We can use the functional organization of a document to help decide which portions of it should be presented to a user and which can be ignored or considered as lower priority. The extraction of functional constructs allows this to be done without the need for content-level reasoning. In fact, many of the relationships which are explicit in the structure cannot be found at the content level; examples are the ordinal relationship between items in a list, or the spatial relationships between columns in a table. Based on these ideas, techniques can be developed to present document images to users who want to browse collections of documents. Such techniques, as illustrated in Section 5.4.4, make it possible to provide documents to a user in a way which is consistent with how the documents were intended to be used, or which is consistent with the goals of the reader. We believe that this will be very helpful in gaining acceptance for electronic representations of documents, since the electronic representation allows the mode of presentation of a document to be modified easily.

5.4 Experiments

In this section we describe some experiments on document use and type classification, and briefly outline some methods of functional enhancement. These tasks rely heavily on the identification of information units, information structures and their properties. The first step, therefore, is a segmentation of the document into appropriate information unit primitives whose properties can be used for classification or enhancement.

5.4.1 Extracting Information Units and Structures

In our experiments, we will consider characters, graphics blocks, and image blocks to be the basic information units. We assume that the document has been separated into text, graphics and image regions, and we then further decompose the text regions. The extraction of information units is related to the Gestalt principles, as discussed briefly in Section 2, and we rely on this in our approach to text segmentation. Proximity grouping of text is performed bottom-up to obtain a component hierarchy, and similarity grouping (boldface, italics and text size) and "good continuation" segmentation are then computed top-down.

Segmentation of Text

Text-based information units vary with physical scale and are dependent on the application at hand. We therefore must be able to represent multiple levels of information units. For text, the hierarchy typically consists of characters, words/phrases, lines, blocks, etc. Other units and levels are typically application-dependent - for example, strokes for handwriting, serifs for font identification, and sentences for content analysis. Details of the text segmentation can be found in [10].

Properties of Text Units

A second level of characterization is based on information unit properties. First, a gross characterization of the text height is made for each block. The height of each line's bounding box is computed, and the average height of all the lines in all multi-line blocks is computed as the average text height, based on the assumption that multi-line text blocks are a good indication of the standard "body" text of a document. Text blocks are then characterized as large or small when they vary by more then 25% from the average.

Words are also identified as italic or boldface. Italic words are identified by the following algorithm. The minimum upright bounding parallelogram (i.e., a parallelogram with horizontal base and top) is constructed for each component and the slant measured relative to the vertical axis. Since it is difficult to make an accurate determination of the angle from short characters, symbols taller then the average are weighted more heavily. Words in which 50% of the characters have slants greater than δ degrees are classified as italic (Figure 22). We have used $\delta = 11$ in our experiments.

Boldface is also identified at the word level, but using a morphological approach applied to individual blocks (Figure 22). An opening transform is applied in an attempt to eliminate or severely distort non-boldface text. An erosion transform is applied until more than 80% of the pixels have been eliminated, at which point a dilation is applied for an equal number of steps. When the resulting image is compared to the original image, words which are not in boldface have very limited similarity to the original while boldface characters tend to remain intact. Note that boldface can be detected only in the presence of normal-weight characters, and the number of erosion steps is dependent on the scanning resolution and the size of the characters. By operating on the block level, problems caused by a wide variety of text sizes, as well as inconsistent illumination, are reduced.

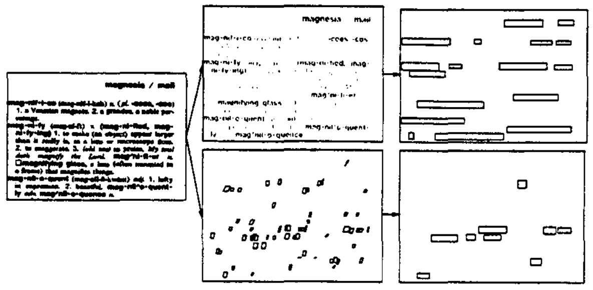


Figure 22: Boldface (top) and italic (bottom) word detection.

5.4.2 Use Classification

As suggested in Section 3, the population of text blocks and their descriptions can be used to classify a document into the usage categories of reading, browsing, and searching (and modifying).

The following heuristics can be used to identify these classes:

Reading documents are characterized by a relatively small number of large text blocks on each page. The majority of the document is composed of text that has a single point size.

Browsing documents tend to have medium to large text blocks, and small text blocks of a larger point size which act as focal points for the reader. Although readable documents have similar handles, browsable documents typically have many such handles.

Searching documents are characterized by small, repetitive text blocks.

We use of these criteria in the block-level segmentation of a browsing document.
Some of the specific properties which can be used include:

- Number of text blocks
- Distribution of the geometrical sizes of the blocks
- Number of words and lines per text block
- Geometrical arrangement of the blocks
- Existence of multiple point sizes
- Existence of graphic and image components

Using a set of very simple criteria, based on a subset of the above properties, we were able to classify approximately 80% of a 100-document database correctly, with approximately 5% being unclassified. The criteria used were as follows:

- In a searching document, no more than 25% of the text blocks should have more than five lines. There should be no image components, and few or no graphic components.
- A browsing document must have at least three head/body pairs. A head is in an emphasized font (boldface, italics, or a large font) and has no more then two lines. A body is standard text with more then two lines.
- A reading document must follow a strict (one- or two-column) column structure and must have large text blocks, primarily of a standard point size.

These criteria will not perform well on very complex structures. One of the difficulties is that many documents belong to more than one use class. Consider, for example, the "yellow pages" of a telephone book. The individual line listings are clearly designed for searching, but they are intermixed with "advertisements" which have browsing characteristics. Similarly, a journal article's bibliography exhibits both reading and searching characteristics.

5.4.3 Type Classification

Type classification is a refinement of use classification; the type of a document refers to a more specific document-level characterization such as journal article or newspaper article, or a page-level characterization such as title or contents page.

We can use function-based analysis as a basis for type classification. As an example of how to perform classification at this level, we attempt to classify individual journal pages as being title, reference or body.

A set of 59 journal page images from the University of Washington English Document Image Database-I was used for training and testing. This database contains images of pages as well as page- and zone-level ground truth for each page. Each description includes general characteristics of the page and characteristics of each zone on the page. The page characteristics include, for example, "dominant-font-size", "dominant-font-style", and "number-of-columns", while the zone characteristics include, for example, "type", "location", "text-alignment", and "dominant-font-style". The classification of pages into the three categories was not provided in the ground truth, and was performed manually.

The complete database was converted to Document Interchange Format (DIF). In this format, each page is described by specifying general information about the page, and a list of zone descriptions.

To classify the pages, we used a small set of attributes of the zones. The most discriminatory attributes turned out to be the number of vertically neighboring zones with consistent height and the average size of the zones.

Using rules based on these attributes we were able to classify journal page images with an accuracy of over 90%. The rules are intuitively plausible and highly consistent with our functional principles. The number and average size of the information units (zones) play major roles in the rules.

Examples of documents that were classified into each class are shown in Figure 23. Note that the second example of a reference page is also a title page.

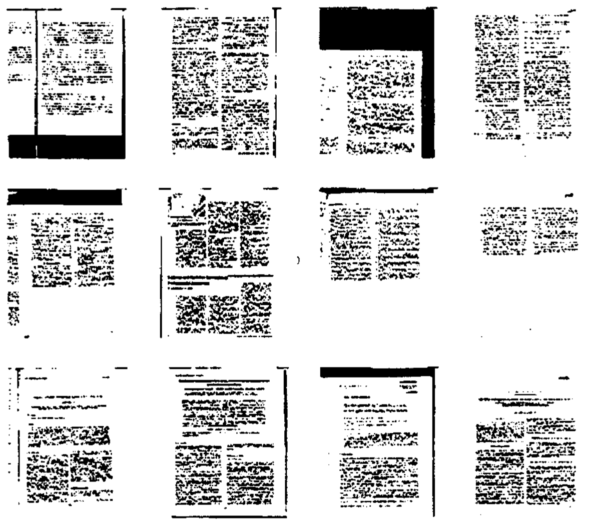
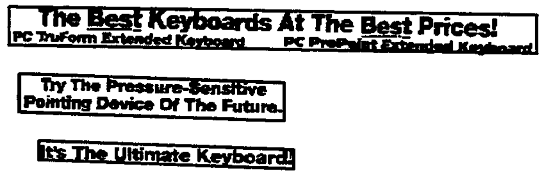


Figure 23: Pages classified as body (top), reference (middle) and title (bottom).



The Best Keyboards At The Best Prices!
PC TruForm Extended Keyboard PC PrePaint Extended Keyboard

Try The Pressure-Sensitive Pointing Device Of The Future.

It's The Ultimate Keyboard!

Figure 24: Enhanced browsing capability

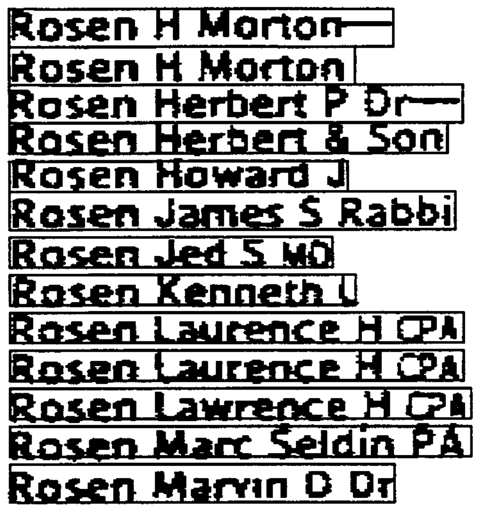
5.4.4 Functional Enhancement

If we can decompose a document into functional components, we can use its functional organization to help decide which portions of it should be presented to the user and which can be ignored or considered as lower priority. The extraction of functional constructs allows this to be done without the need for content-level reasoning. Using these ideas,we can present document images to users in accordance with their goals. If a user wants, for example, to browse collections of documents, we can provide only the upper-level headers, and give the user the option to retrieve full information when needed.

Assume that a user wants to browse through a document. We can present the information in a manner consistent with the traversal mode by giving the title of each information unit (see Figure 24), and allowing the user to ask for the full unit if needed Search documents such as a phone book can be

5.5 Discussion

Document functionality relates to how the document conveys information to its user. In this paper, we have provided a basis for understanding the functional aspects of document design and usage. Authors use layout and emphasis to make it easier to extract information from documents. Traditional



Rosen H Morton
Rosen H Morton
Rosen Herbert P Dr
Rosen Herbert & Son
Rosen Howard J
Rosen James S Rabbi
Rosen Jed S MD
Rosen Kenneth L
Rosen Laurence H CPA
Rosen Laurence H CPA
Rosen Lawrence H CPA
Rosen Marc Seldin PA
Rosen Marvin D Dr

Figure 25: Enhanced search capability

document understanding and conversion techniques have ignored the intended functionality of the document, especially its class-independent functional structure. An important advantage of our approach is that it provides an ability to organize documents without understanding their content.

We plan to extend our work to provide a more complete taxonomy of functional primitives, and to implement a full-scale system for functional typing and document classification.

6 Signature Verification

Progress on the problem of signature verification has advanced more rapidly in on-line applications than off-line applications, in part because information which can easily be recorded in on-line environments, such as pen position and velocity, is lost in static off-line data. In off-line applications, valuable information which can be used to discriminate between genuine and forged signatures is embedded at the stroke level. Examining properties of strokes is difficult, however, due to issues involving the segmentation of often stylish and unconventional strokes. In this work, we present an approach to segmenting strokes into stylistically meaningful segments and establish a local correspondence between a questioned signature and a reference signature to enable the analysis and comparison of stroke features.

From the reference signature, we extract a partially ordered set of feature points which are used to generate a model. Given a questioned signature, we robustly extract stroke features based on an analysis of the edges obtained from grey level images. Edge features tend to be less distorted than skeletons obtained by thinning and allow us to extract a more stable representation to match. Questioned signatures which do not conform to the reference signa-

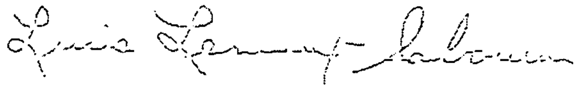


Figure 26: Extracted Model Strokes.

ture are identified as *random* forgeries. Most *simple* forgeries can also be identified, as they do not conform to the reference signature's invariant properties such as connections between letters. Since we have access to both local and global information, our approach also shows promise for extension to the identification of *skilled* forgeries.

6.1 Model Acquisition

6.1.1 Acquisition of model strokes

As mentioned previously, the basic model includes a *tracing* of the signature, either gathered a priori online, extracted automatically [11], or traced manually with a digitizer. Since this process must only be done once for each signature, even a manual process is acceptable[3].

After obtaining the tracing of a reference instance, we segment the tracing into stroke segments at its junction and end points. The segmented model consists of stroke segments, junction points and endpoints. Note, that any strokes which may appear as "retraced strokes" in an image are represented explicitly in the model.

6.1.2 Features Extraction

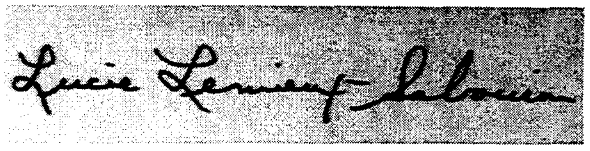
The features that will be used to verify the signature are those which tend to be consistently produced by the writer when producing their signature. Recall that we are only dealing with features that will allow us to establish a correspondence, not features to which prove authorship. The features computed for each stroke include average curvature, the incoming and outgoing angle, and the relative position and the relative size of the segment.

6.1.3 Training the Model

During the training process there are two primary considerations. One is the model and the other is the expected "cost" of matching a genuine signature to the model.

The weight, $w_k(i)$, of feature k and segment i is computed as the normalized standard deviation of the cost of the features and stored as part of the model. In some sense, we bootstrap ourselves into computing the weights. Given a single reference signature, we segment and compute the appropriate features. We then assume that all weights are one and match all of the other training signatures according to the procedure described in Section 6.3.

[3]Note that the tracing does not have be performed in exactly the same way as the original signature was written, since the segmentation and ordering of the strokes in the questioned signature will be model driven.


(a)

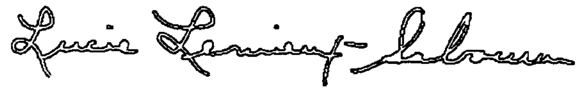

(b)

Figure 27: (a) a signature sample; (b) the edge image of (a).

This establishes an initial correspondence. From this correspondence, we can compute the standard deviations for each feature on each stroke segment in the model.

6.2 Questioned Signature Processing

The first step in processing the questioned signature is to extract segments to match. Since there is only a single static instance of an image to verify, care must be taken to extract a meaningful representation. As we have mentioned, we rely on edge information as a basis for extracting strokes, as opposed to using a thinned representation.

We apply the Canny edge detector to the grey level image [4], which provides a reasonable set of edge hypothesis. Tses edges are analyized to produce a stable midline representation of the signature.

The first process in stroke extraction is to trace the stroke, segment them at junction (crossing) and end points, classify individual segments as simple strokes and collapsed loops. Subsequent processing extracts simple strokes, extracts closed loops and merges fragmented strokes as described in [15].

Once the questioned signature is segmented, a dynamic programming technique is used to try to match the segmented representation with the derived model.

6.3 Signature Matching

Given a questioned signature, we want to find a segment-wise correspondence between it and the model in order to examine additional features of the signature. The feature extracted from the questioned signature are the same as the ones used in the model, namely the relative size and position of the segment, its curvature and angles at its endpoints. Given a model and a questioned signature, we define a cost function that is used to evaluate the quality of the match. Minimizing this cost function results in the best correspondence.

6.3.1 Cost Function

The cost of matching the data and model is given by

$$d = \sum_{i=1}^{M} d(i)$$

d(i) is the measure of dissimilarity between two strokes and M is the total number of strokes in the model. $d(i)$ is simply the weighted sum of the five features extracted. Thus,

$$d(i) = \sum_{k=1}^{5} w_k(i) * d_k(i)$$

Where $d_1(i)$ is the difference in relative position between the model and data on stroke segment i, $d_2(i)$ is the difference in incoming angles, $d_3(i)$ is the difference in outgoing angle, $d_4(i)$ is the difference in average curvature and $d_5(i)$ is the difference in size. Recall, the $w_k' s$ are the per stroke segment weights computed based on the standard deviation.

6.3.2 The Matching Procedure

The algorithm for matching is a combination of dynamic programming and elastic matching [49]. To do this, a tree is constructed where each segment can be matched to either a corresponding neighbor in the model, a previous neighbor or the next neighbor. At each stage, the cost of matching is computed and the minimal cost path is chosen. A given segment is allowed to match to the corresponding stroke, the next stroke or the previous stroke in the model.

6.3.3 Random Forgery Detection

In the process of making the local correspondence, we calculated the final cost as the accumulated cost for the entire signature. As a result, the genuine signatures which corresponds well with a model is reflected in a lower cost. During the training, each of the training signatures is matched to the model, and cutoff score which is computed from the training set for each is used to determine if the match is low enough to be considered on a genuine signature. The random forgeries which generated without knowing the spelling of the signature will not confirm to the structure of the genuine signature and tend to have a much higher cost.

6.4 Experiments

We have carried out experiments on a set of 800 signatures[4]. No normalization or alignment of the signatures was required. The database consists of 800 off-line signatures, with 40 signatures collected from each of the 20 authors, and the location portions of the model was acquired by manually tracing the signature point by point from a rendered version of the image.

[4]The signature database was provided by Prof. Robert Sabourin at *École de Technologie Supérieure*.

Training set size	Type I	Type II
10	2.2%	0.97%
20	1.67%	0.56%

Table 3: Threshold chosen from training data to minimize total error.

Training set size	Type I	Type II
10	1%	2.1%
20	0%	0.75%

Table 4: Threshold chosen from training data to minimize Type I (missed forgery) errors.

For each experiment, we used the remaining (40-N) signatures of each class for testing. We match each test signature against the model, and if the cost exceeds the threshold computed during training, we reject the signature as a forgery, otherwise, we accept it as a genuine signature. The verification threshold is again determined by calculating the cost of matching the training signatures to the model.

We performed two sets of experiments by making the local correspondence between the model and questioned signatures, using 10 and 20 signatures in the training respectively. For each of the training set sizes (10 and 20), we experiments with two different methods of computing the threshold. The first set (shown in Figure 3 attempted to minimize the total error between the genuine training set and 10 additional forgered signatures. The second set (shown in Figure 4 attempted to minimize the the type I error using the genuine training set and 10 additional forgered signatures.

6.5 Discussion

We have proposed a model-based segmentation approach for the verification of static (off-line) signature images and the detection of forgeries. Our segmentation involves identification of junction points and recovery of the strokes consistent with the model. For verification, a questioned signature is segmented based on the edge information and the features of the segments such as width, direction and type (loop, retrace, etc.). A matching process attempts to establish a correspondence between the test image and the model, and the quality of the correspondence is used for detecting forgeries.

A significant feature of our approach is that we obtain not only a reasonably good segmentation but also a segment-wise correspondence between the model and the questioned signatures. This enables us to examine both global and local features of the questioned signature. Thus, it can serve as an initial step in the detection of skilled forgeries. Preliminary work on the processing of simple forgeries suggests that a single cost function on selecting signature features may provide sufficient discrimination power.

7 Other projects

We have a number of other ongoing projects in document and video areas. Limited information about the following projects can be found on our WWW page at http://documents.cfar.umd.edu/LAMP.

Distributed Document Testing - software engineering project in providing document testing and management between developers working in different locations.

Video Indexing and Retrieval - MERIT - Mpeg Encoded Retrieval and Indexing Toolkit - a compressed domain segmentation, indexing and retrieval package.

Video Performance Evaluation - a system begin developed to provide ground truth and evaluation of the performance of video segmentation, classification and labeling schemes.

For other information, please contact us directly.

References

[1] H.S. Baird. The skew angle of printed documents. In *SPSE 40th Annual Conference and Symposium on Hybrid Imaging Systems*, pages 21 - 24, 1987.

[2] H.S. Baird. Anatomy of a versatile page reader. *Proceedings of the IEEE*, 80:1059–1065, 1992.

[3] H.S. Baird, H. Bunke, and K. Yamamoto. *Structured Document Image Analysis*. Springer, Berlin, 1992.

[4] J. F. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8:679–698, 1986.

[5] F. R. Chen, L. D. Wilcox, and D. S. Bloomberg. Detecting and locating partially specified keywords in scanned images using hidden Markov models. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 133 - 138, 1993.

[6] F.R. Chen, D.S. Bloomberg, and L.D. Wilcox. Spotting phrases in lines of imaged text. In *Proceedings of the SPIE - Document Recognition II*, pages 256–269, San Jose, CA, 1995.

[7] J.L. DeCurtins and E.C. Chen. Keyword spotting via word shape recognition. In *Proceedings of the SPIE - Document Recognition II*, pages 270–277, San Jose, CA, 1995.

[8] A. Dengel, R. Bleisinger, F. Fein, R. Hoch, F. Hones, and M. Malburg. OfficeMAID — A system for office mail analysis, interpretation and delivery. In *International Workshop on Document Analysis Systems*, pages 253 - 276, 1994.

[9] D. Doermann, H. Li, and O. Kia. Duplicate document image detection. Technical report, University of Maryland, College Park, 1996.

[10] D. Doermann, E. Rivlin, and A. Rosenfeld. The function of documents. Technical Report CAR-TR-841, University of Maryland, 1996. To appear in IJCV.

[11] D. Doermann and A. Rosenfeld. The interpretation and recognition of interfering strokes. In *Proceedings of the International Workshop on Frontiers in Handwriting Recognition*, pages 41–50, 1993.

[12] D. Doermann and C. Shin. The development of a general framework for intelligent document image retrieval. In *Document Analysis Systems*, pages 605–632, 1996.

[13] L. Van Gool et al. Survey: Texture analysis. *Computer Vision, Graphics and Image Processing*, pages 336–357, 1985.

[14] V.N. Gudivada and V.V. Raghavan. Spatial similarity based retrieval in image databases. In *Symposium on Document Analysis and Information Retrieval*, pages 255 - 270, April 1993.

[15] K. Guo, D. Doermann, and A. Rosenfeld. Local correspondence for detecting random forgeries. In *ICDAR*, 1997. Accepted.

[16] P. Herrmann and G. Schlagetar. Retrieval of document images using layout knowledge. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 537–540, 1993.

[17] J.J. Hull. Document image matching and retrieval with multiple distortion- invariant descriptors. In *International Workshop on Document Analysis Systems*, pages 383 - 400, 1994.

[18] International Standards Organization. *Text and Office Systems—Office Document Architecture (ODA) and Interchange Format*, 1989. International Standard 8613.

[19] T. Kanungo, R.M. Haralick, and I. Phillips. Global and local document degradation models. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 730 - 734. IEEE Computer Society Press, 1993.

[20] H. Kauniskanga and M. Pietikainen. Development support for content-based image retrieval systems, 1996.

[21] O. Kia and D. Doermann. Structural compression for document analysis. In *Proceedings of the International Conference on Pattern Recognition*, volume C, pages 664–668, Vienna, Austria, 1996.

[22] O. Kia and D. Doermann. Structure-preserving image compression and transmission. In *Proceedings of the International Conference on Image Processing*, volume I, pages 193–196, Lausanne, Switzerland, 1996.

[23] O. Kia, D. Doermann, and R. Chellappa. Compressed-domain document retrieval and

analysis. In *Proceedings of the SPIE - Multimedia Storage and Archiving Systems*, volume 2916, pages 176–187, 1996.

[24] Omid E Kia. *Document Image Compression and Analysis.* PhD thesis, University of Maryland, 1997.

[25] K. Koffka. *Principles of Gestalt Psychology.* Harcourt, Brace & World, New York, 1935.

[26] M. Koga, T. Murakami, Y. Shima, and H. Fujisawa. Structure analysis method of graph image for document image retrieval. In *Proceedings of the SPIE - Character Recognition Technologies*, pages 291–295, San Jose, CA, 1993.

[27] M. Krishnamoorthy, G. Nagy, S. Seth, and M. Viswanathan. Syntactic segmentation and labeling of digitized pages from technical journals. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15:737–747, 1993.

[28] C. Lagoze, E. Shaw, J.R. Davis, and D.B. Krafft. Dienst: Implementation reference manual. Technical report, Cornell University, 1995.

[29] J. Liu, C.M. Lee, and R.B. Shu. An efficient method for the skew normalization of a document image. In *Proceedings of the International Conference on Pattern Recognition*, pages 122–125, 1992.

[30] O. Lorenz and G. Monagan. A retrieval system for graphical documents. In *Symposium on Document Analysis and Information Retrieval*, pages 291–300, Las Vegas, NV, 1995.

[31] C. Maa. Identifying the existence of bar codes in compressed images. *Computer Vision, Graphics and Image Processing*, 56(4):352, 1994.

[32] S. Mori, C.Y. Suen, and K. Yamamoto. Historical review of OCR research and development. *Proceedings of the IEEE*, 80:1029–1058, 1992.

[33] G. Nagy, P. Sarkar, D. Lopresti, and J. Zhou. Spatial sampling of printed patterns. Draft, 1996.

[34] University of Washington. Document image databases collection. CDROM.

[35] L. O'Gorman. The document spectrum for page layout analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15:1162–1173, 1993.

[36] T. Ojala, M. Pietikinen, and D. Harwood. A comparative study of texture measures with classification based on feature distributions. *Pattern Recognition*, 29:51–59, 1996.

[37] T. Pavlidis. Problems in recognition of poorly printed text. In *Symposium on Document Analysis and Information Retrieval*, pages 162 – 173, March 1992.

[38] P. Sarkar. Random phase spatial sampling effects in digitized patterns. Master's thesis, Rensselaer Polytechnic Institute, 1994.

[39] J. Sauvola and M.Pietikainen. Page segmentation and classification using fast feature extraction and connectivity analysis. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 1127–1131, Montreal, Canada, 1995.

[40] J. Sauvola and M. Pietikainen. A document management interface utilizing page decomposition and content-based compression. In *Proceedings of the International Conference on Pattern Recognition*, Vienna, Austria, 1996.

[41] A. Soffer. *Retrieval by Content in Symbolic-Image Databases.* PhD thesis, University of Maryland, College Park, MD, 1995.

[42] A.L. Spitz. Skew determination in ccitt group 4 compressed document images. In *First Symposium on Document Analysis and Information Retrieval*, pages 11 – 25, 1992.

[43] A.L. Spitz. Using character shape codes for word spotting in document images. In *Shape, Structure and Pattern Recognition*, pages 382–389. World Scientific, Singapore, 1995.

[44] A.L. Spitz. Logotype detection in compressed images using alignment signatures. In *Proceedings of the Fifth Symposium on Document Analysis and Information Retrieval*, pages 303–310, 1996.

[45] R.K. Srihari. Automatic indexing and content-based retrieval of captioned photographs. In *Proceedings of the International Conference on Document Analysis and Recognition*, pages 1165–1168, Montreal, Canada, 1995.

[46] R.K. Srihari. Automatic indexing and content-based retrieval of captioned photographs. *IEEE Computer*, pages 49–56, 1995.

[47] A. Takasu, S. Satoh, and E. Katsura. A document understanding method for database construction of an electronic library. In *Proceedings of the International Conference on Pattern Recognition*, pages 463–466, 1994.

[48] H. Tanaka and A. Kogawara. High speed string edit methods using hierarchical files and hashing technique. In *Proceedings of the International Conference on Pattern Recognition*, pages 334–336, 1988.

[49] C.C. Tappert and J.M. Kurtzberg. Elastic matching for handwritten symbol recognition. Technical Report 9988, IBM, 1983.

[50] S.L. Taylor. Information-based document analysis systems in a distributed environment. In *International Workshop on Document Analysis Systems*, pages 93–108, 1994.

[51] T. Watanabe, Q. Luo, and N.Sugie. Structure recognition methods for various types of documents. *Machine Vision and Applications*, 6:163–176, 1993.

Scripts and Languages

SIMULTANEOUS IDENTIFICATION OF SCRIPT AND ORIENTATION

Alan S. Ratner
Department of Defense
9800 Savage Road
Fort Meade, MD 20755-6518
alanratner@alum.mit.edu

Abstract

The distribution of low-order moments is a useful tool for classifying image characteristics. On a macroscopic scale, we can identify the script and orientation of a document image. At a finer level, we can identify language, case, alphabetic versus numeric, and possibly topic.

1 Introduction

Analysis of document images typically requires four basic steps: identification of orientation, script, language, and characters.

Orientation identification is the task of determining which way is up. Assuming limited skew, there are four possible orientations.

Script identification is the task of determining the script, or alphabet, on the document. The scripts "in common use" are: Amharic, Arabic, Armenian, Bengali, Burmese, Chinese, Cyrillic, Devanagari, Georgian, Greek, Gujarati, Gurmukhi, Hebrew, Japanese, Kannada, Khmer, Korean, Latin (or Roman), Lao, Malayam, Maldivian, Mongolian, Oriya, Sinhalese, Tamil, Telugu, Thai, Tibetan and Urdu [1].

Language identification is the task of determining the language in use. Over 500 languages use the Latin script including English, French, Hawaiian and transliterated Russian. Over 60 languages use the Cyrillic script including Russian, Ukrainian, Mordvin and Udmurt.

Character (or word) identification is the task of converting a character (or word) image to an encoding scheme such as Unicode or ASCII. In some applications character identification may be performed prior to (or iteratively with) language identification.

This paper addresses a simple, efficient and effective scheme to accomplish the first three steps: orientation identification, script identification and language identification.

In Section 2 we describe the observations made on each character or connected component. In Section 3 we describe how the features used by the classifier are obtained from the observations. Section 4 presents several classification examples.

2 Observations

The information used to characterize a document image consists of a number of parameters called observations. The initial observations are based on the calculation of normalized low-order centered moments of either each character or connected component. (A connected component is a set of horizontally, vertically or diagonally contiguous black pixels.) We take the x direction as running from left to right and the y direction as running from top to bottom on the document image (assuming, at present, that the document is right-side up). For each character or connected component on the image we:

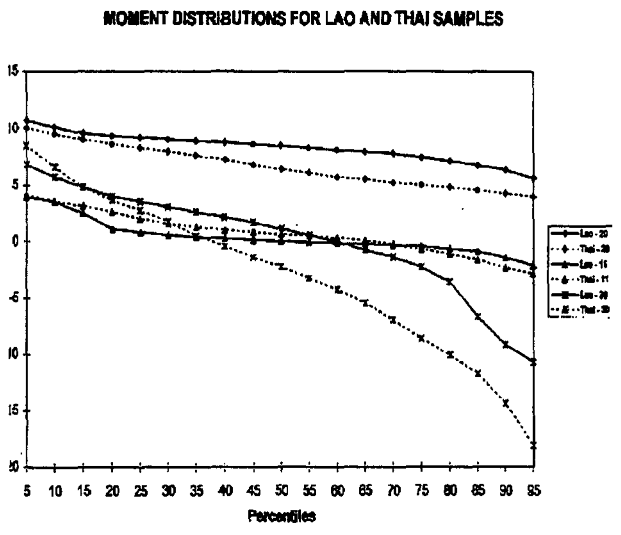
1. Compute the horizontal and vertical mean positions, $\bar{x}$ and $\bar{y}$, for the black pixels.
2. Compute the number of black pixels, N.
3. Compute the horizontal and vertical extent of the black pixels, D_x and D_y. (The number of pixels between and including the left-most and right-most black pixel within the component is D_x.)
4. Compute the normalized centered moments m_{ab}:

$$m_{ab} = \left(\sum_{(x,y) \in blackpixels} (x-\bar{x})^a \times (y-\bar{y})^b\right) / N \times \Delta_x^{\ a} \times \Delta_y^{\ b}$$

where (a,b) take on the values:
(2,0) for the x^2 (horizontal second) moment,
(1,1) for the xy (diagonal second) moment,
(0,2) for the y^2 (vertical second) moment,
(3,0) for the x^3 (horizontal third) moment,
(2,1) for the x^2y (diagonal third) moment,
(1,2) for the xy^2 (diagonal third) moment,
(0,3) for the y^3 (vertical third) moment.

We have now described each component on the document with its three second moments and four third moments. However, our interest is not in the moments describing specific instances of components, but rather in the distribution of these moments over all the components on the document. We would expect that all instances of character 1 (say "a" in the Latin script) would have similar values of m_{11} but distinctly different from those of instances of character 2 (say "b"). Thus the distribution of each moment involves the frequency of occurrence of each character as well as the moments of these characters. We sort the moments and select the inner quintiles: the 20, 40, 60 and 80 percentiles of each moment. We now have 28 observations to describe each document image: the 20, 40, 60 and 80 percentile values of each of the seven normalized moments. Figure 1 shows the sorted (2,0), (1,1) and (3,0) moments for one Lao and one Thai image.

Figure 1: Sorted Moments



3 Features

The 28 observations could be used as features directly. However, it is desirable to reduce the dimensionality of the problem in order to simplify the model, allow visualization and reduce the amount of training data required. If we wish to classify the images as belonging to one of C classes where C is less than 28, we can use the generalization of Fisher's linear discriminant [2] to project the observations into C-1 features using linear combinations of the observations which maximize the projections of between-class scatter to within-class scatter. In many instances, C may be small. For example: a) is a document in the Latin or Cyrillic script? or b) what is the orientation of a Chinese document?

4 Classification

The simplest classifier is Gaussian. The means and standard deviations of the features for each class are computed. At the point defined by the feature vector of an unknown document we compare the relative magnitude of the C Gaussian distributions which are weighted by the *a priori* probabilities.

Figure 2 shows a 3-class system distinguishing between Burmese, Lao and Thai scripts using a 2-dimensional feature vector. The data set consisted of images with relatively low resolution. The boundary regions, 1-sigma ellipses around the class means, and data points are shown. The error rate is in the bottom right corner. This figure was generated by LNKnet [3].

Figure 2: 3-Class Script Identification.

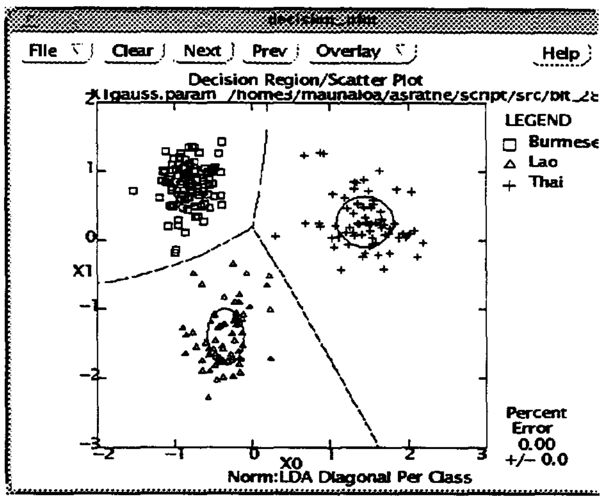
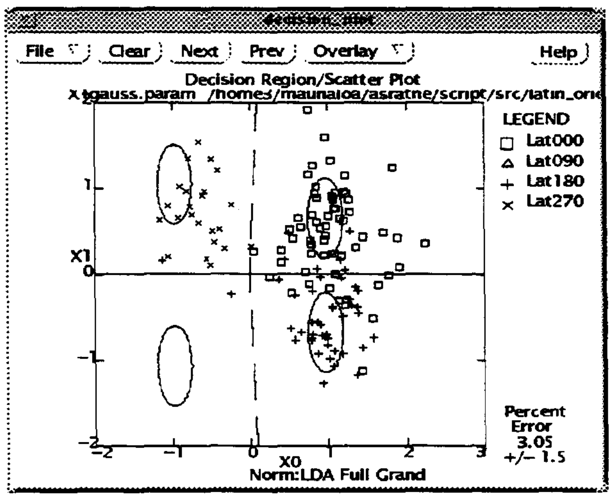


Figure 3 shows a 4-class system distinguishing orientation for low resolution Latin images. In this case, the classifier was trained using upright images with other orientations simulated by interchange and sign changes on the moments. (See Appendix.)

Figure 3: 4-Class Orientation Identification

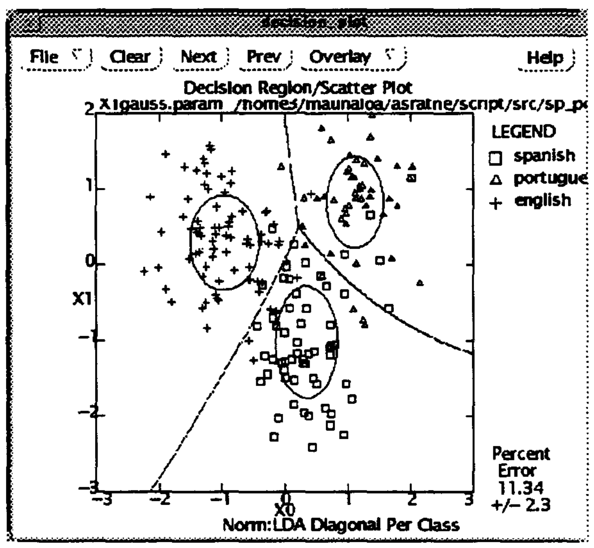


The data set presented in Figure 3 does not contain any images with 270° orientation. With as few as 4 classes we observe visualization problems. Although a relatively large number of data points appear to be in error, this is an artifact of viewing a two-dimensional slice of a three-dimensional space.

We can also simultaneously classify images by script and orientation although this is a four-fold increase in number of classes (if all four orientations are possible) with resultant increase in complexity of the model and accompanying increase in error rate.

Given a set of images with the same script, we can identify the language apparently based on differences in the script characters and in differences in the frequency of occurrence of the characters. Figure 4 shows the classification of low resolution English, Spanish and Portuguese document images.

Figure 4: 3-Class Language Identification



References

[1] A. Nakanishi, *Writing Systems of the World* (Tuttle, Rutland VT, 1980).

[2] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis* (Wiley, New York, 1973) 114-121.

[3] R. P. Lippman, L. Kukolich and E. Singer, *LNKnet: Neural Network, Machine-Learning, and Statistical Software for Pattern Classification*, Lincoln Lab. Journal (1993) 249-267.

Appendix

The following table shows the relationship between the moments as a function of orientation with angle measured in a clockwise direction:

Table 1: Moment Translation with Orientation

	Moments						
Orient	x^2	xy	y^2	x^3	x^2y	xy^2	y^3
0°	m_{20}	m_{11}	m_{02}	m_{30}	m_{21}	m_{12}	m_{03}
90°	m_{02}	$-m_{11}$	m_{20}	$-m_{03}$	m_{12}	$-m_{21}$	m_{03}
180°	m_{20}	m_{11}	m_{02}	$-m_{03}$	$-m_{21}$	$-m_{12}$	$-m_{03}$
270°	m_{02}	$-m_{11}$	m_{20}	m_{03}	$-m_{12}$	m_{21}	$-m_{03}$

Page Segmentation Using Script Identification Vectors: A First Look

Judith Hochberg, Michael Cannon, Patrick Kelly, and James White

Abstract

This paper explores the use of script identification vectors *in the analysis of multilingual document images. A script identification vector is calculated for each connected component in a document. The vector expresses the closest distance between the component and templates developed for each of thirteen scripts, including Arabic, Chinese, Cyrillic, and Roman. We calculate the first three principal components within the resulting thirteen-dimensional space for each image. By mapping these components to red, green, and blue, we can visualize the information contained in the script identification vectors.*

Our visualization of several multilingual images suggests that the script identification vectors can be used to segment images into script-specific regions as large as several paragraphs or as small as a few characters. The visualized vectors also reveal distinctions within scripts, such as font in Roman documents, and kanji *vs.* kana *in Japanese.*

Results are best for documents containing highly dissimilar scripts such as Roman and Japanese. Documents containing similar scripts, such as Roman and Cyrillic, will require further investigation.

1 Introduction

Document images in which different scripts, such as Chinese and Roman, appear on a single page pose a problem for optical character recognition (OCR) systems. Existing OCR capabilities for multi-script recognition are limited to identifying individual foreign characters, such as Roman characters within a page of Chinese. This approach is inappropriate for documents with larger numbers of possible foreign characters, e.g., Roman documents that contain Chinese characters. It also does not allow larger textual regions containing foreign scripts to be processed by more powerful OCR algorithms which incorporate contextual and linguistic information.

One possible solution to this problem is to extend the user-guided page segmentation approach currently used by OCR packages to process multilingual documents printed in a single script. That is, the user could make a separate scan of the document for each script, each time manually selecting the region(s) to be recognized. It would be preferable to develop an automatic means of segmenting multi-script document images. This could be the first step in a document processing stream for multi-script documents that includes language identification as well as content-based steps, such as machine translation, information retrieval, indexing, etc.

We hypothesized that a good starting point for such an algorithm would be the *script identification vectors* produced by our published algorithm for performing script identification on single-script documents [1]. Our current implementation encompasses thirteen scripts: Arabic, Armenian, Burmese, Chinese, Cyrillic, Devanagari, Ethiopic, Greek, Hebrew, Japanese, Korean, Roman, and Thai. It requires minimal preprocessing of the document image, and identifies the script in which a document is printed with a high degree of accuracy.

The algorithm is based on cluster analysis. Connected components from a training set of documents are clustered in order to determine the most frequent character types in each script. A representative template is chosen for each cluster. New documents are classified by comparing a subset of their connected components to the templates for each script, and choosing the script whose templates provide the best match.

As part of this process, a thirteen-element script identification vector is developed for each connected component in the test document. The first element is the Hamming distance between the component and its most similar Arabic template, the second is the distance between the component and its most similar Armenian template, and so on for all thirteen scripts currently in our system.

Each of these 'most similar' templates has an associated reliability statistic that was calculated in a second pass through our training set, as part of the training procedure described in [1]. As shown in Figure 1, reliable templates are true hallmarks of a script; unreliable templates tend to be blobs or other uninformative shapes.

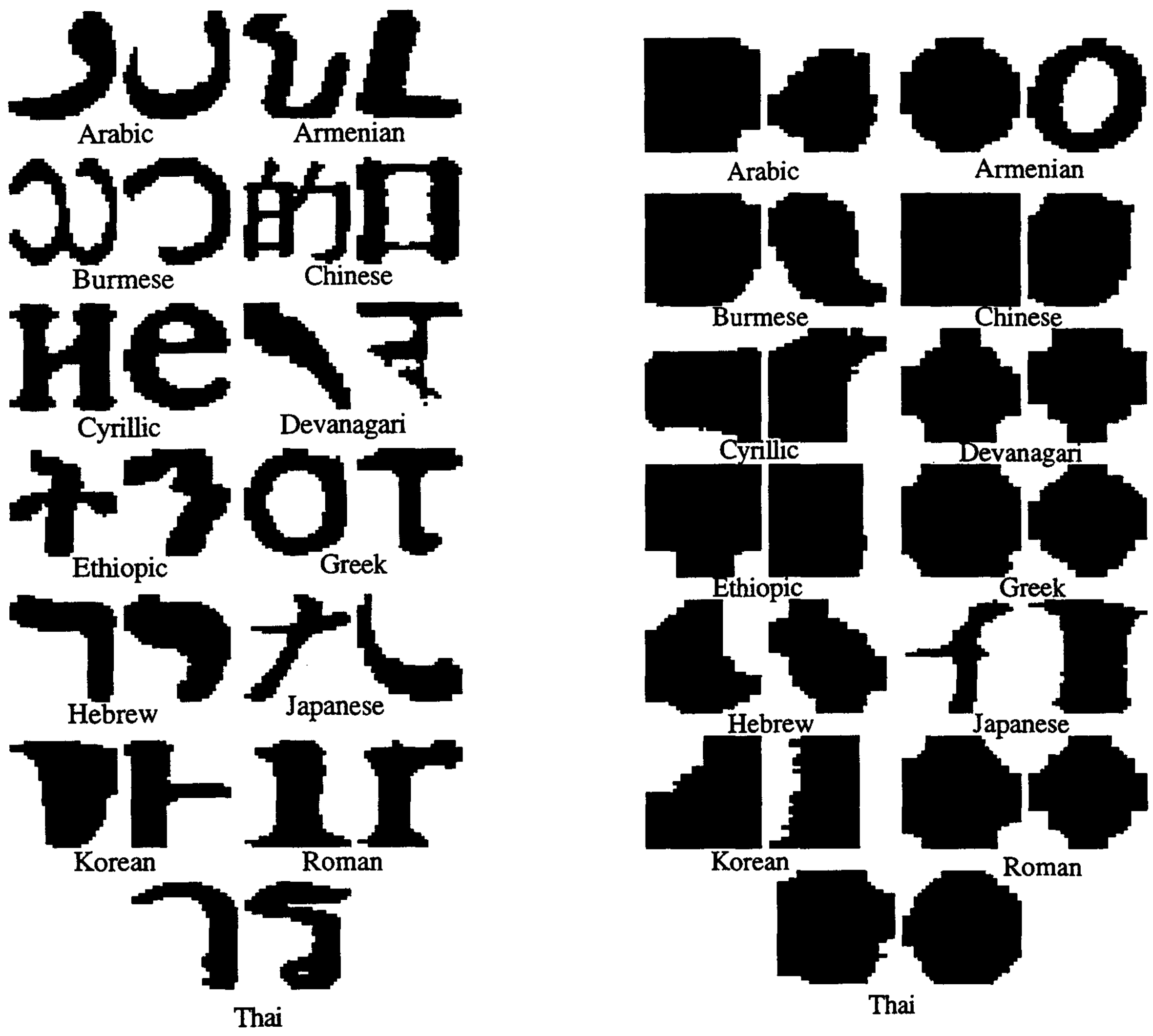


Fig. 1. Some frequent reliable (left) and unreliable (right) templates in thirteen scripts (from [1])

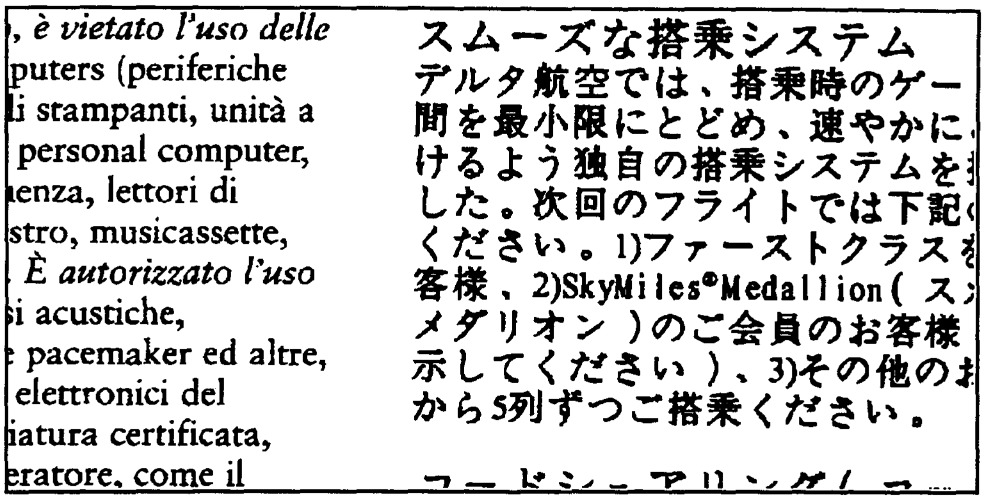
The script identification vectors were not intended to be accurate on an individual component basis. For example, a particular component from an Arabic document might have a higher matching score for Roman than for Arabic. Therefore, when using the vectors to perform script identification on a document as a whole, we considered several connected components simultaneously; our best results were obtained by examining more than 75 components per image. Combining information across components filtered out the noise and led to an accurate classification for the image. The question remained whether the information contained in the vectors was accurate enough to perform reliable script identification for textual units smaller than a page, such as a line or paragraph, or even an individual word.

Before trying to develop an algorithm to perform this task, we decided to use visualization as a tool to unpack the information inside the vectors. If the visualization showed good separation between script regions, this would be our go-ahead to develop script segmentation algorithms based on the vectors.

An eventual algorithm would use all thirteen elements from the script identification vectors. For the first look described in this paper, we reduced the vectors to three elements, using principal components analysis, so that we could easily visualize the information they contained. This is a lossy approach, yet, as it turns out, sufficient for most script combinations we analyzed.

2 Method

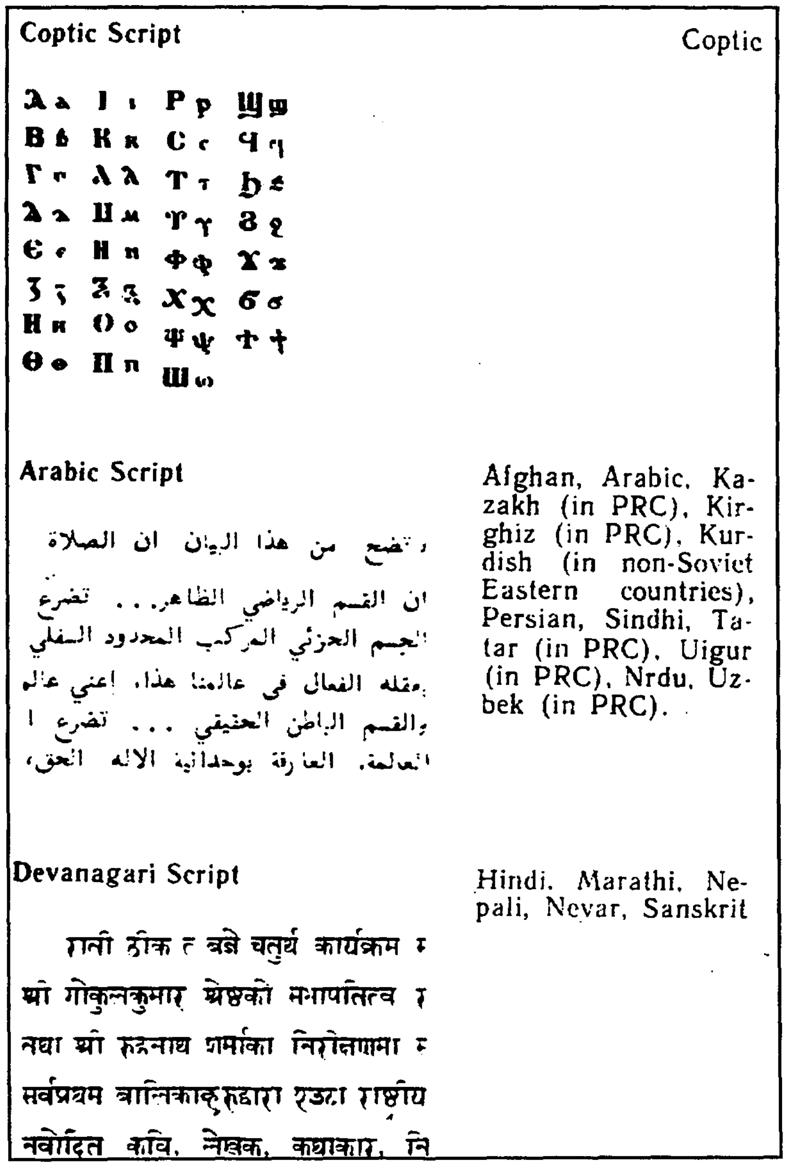
We collected an initial corpus of seven multi-script documents. Image sources were an airline magazine (Fig. 2), a book about manual script identification (Fig. 3, [2]), and bilingual dictionaries (e.g., Fig. 4). Script regions in these images varied in size from individual characters to entire paragraphs. All images were scanned as line art (black and white), with a resolution of 200 dpi, using an Agfa scanner equipped with StudioScan II software.



, è *vietato l'uso delle*
puters (periferiche
i stampanti, unità a
personal computer,
enza, lettori di
stro, musicassette,
È autorizzato l'uso
i acustiche,
pacemaker ed altre,
elettronici del
iatura certificata,
eratore, come il

スムーズな搭乗システム
デルタ航空では、搭乗時のゲー
間を最小限にとどめ、速やかに
けるよう独自の搭乗システムを
した。次回のフライトでは下記
ください。1)ファーストクラス
客様、2)SkyMiles®Medallion(ス
メダリオン)のご会員のお客様
示してください)、3)その他のお
から5列ずつご搭乗ください。

Fig. 2. Fragment of a multilingual image from an airline magazine



Coptic Script

Coptic

Ⲁⲁ Ⲓⲓ Ⲣⲣ Ϣϣ
Ⲃⲃ Ⲕⲕ Ⲥⲥ Ϥϥ
Ⲅⲅ Ⲗⲗ Ⲧⲧ Ϧϧ
Ⲇⲇ Ⲙⲙ Ⲩⲩ Ϩϩ
Ⲉⲉ Ⲛⲛ Ⲫⲫ Ϫϫ
Ⲍⲍ Ⲝⲝ Ⲭⲭ Ϭϭ
Ⲏⲏ Ⲟⲟ Ⲯⲯ Ϯϯ
Ⲑⲑ Ⲡⲡ Ⲱⲱ

Arabic Script

يتضح من هذا البيان ان الصلاة
ان القسم الرياضي الظاهر... تضرع
الجسم الجزئي المركب المحدود السفلي
ومقله الفعال في عالمنا هذا، اعني عالم
والقسم الباطن الحقيقي ... تضرع ا
العاقلة. العارفة بوحدانية الاله الحق،

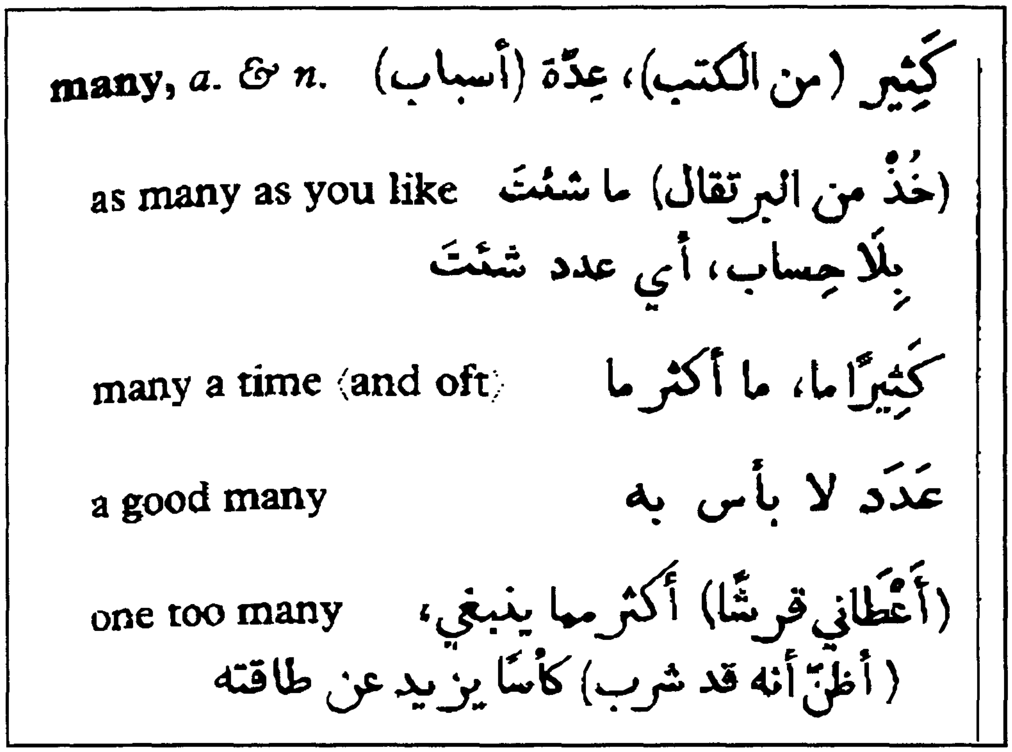
Afghan, Arabic, Kazakh (in PRC), Kirghiz (in PRC), Kurdish (in non-Soviet Eastern countries), Persian, Sindhi, Tatar (in PRC), Uigur (in PRC), Nrdu, Uzbek (in PRC).

Devanagari Script

रानी ठीक द बजे चतुर्थ कार्यक्रम र
श्री गोकुलकुमार श्रेष्ठको सभापतित्व र
तथा श्री रुद्रनाथ शर्माका निर्देशनमा र
सर्वप्रथम क्रान्तिकाव्यद्वारा एउटा राष्ट्रिय
नवोदित कवि, लेखक, कथाकार, नि

Hindi, Marathi, Nepali, Nevar, Sanskrit

Fig. 3. Fragment of a multilingual image from a script identification manual [2]


many, *a. & n.* كَثِير (من الكتب)، عِدَّة (أسباب)

as many as you like (خُذْ من البرتقال) ما شئتَ
بِلَا حِساب، أي عدد شئتَ

many a time (and oft) كَثِيرًا ما، ما أكثر ما

a good many عَدَد لا بأس به

one too many (أَعْطاني قرشًا) أكثر مما ينبغي،
(أظنّ أنه قد شرب) كأسًا يزيد عن طاقته

Fig. 4. Fragment of a multilingual image from an English-Arabic dictionary

The scripts represented in each image are presented in Table 1. Most of the scripts in these documents were among the thirteen included in the script identification templates. The exceptions were all found in images 4-5, from the script identification book [2].

Table 1: Sources and scripts in test images

Image #	Source	Scripts
1	dictionary	Arabic, Roman
2	airline magazine	Japanese, Roman
3	dictionary	Korean, Roman
4	script identification book	Roman, Coptic, Arabic, Devanagari, Armenian, Ethiopic, Bengali
5	script identification book	Roman, Chinese, Nasi pictorial script, Hebrew, Javanese, Avestan, Mayan hieroglyphics
6	dictionary	Greek, Roman
7	dictionary	Cyrillic, Roman

The first step in processing each image was to develop a script identification vector, using the approach described in [1]. This involved the following steps:

- identifying all connected components in the image that contained more than ten pixels and were less than 80 pixels in height;
- rescaling components to 30 x 30 pixels;
- comparing each component to the script identification templates for the thirteen scripts in order to find the template within each script that gave the closest match (judged by Hamming distance). This step created a script identification vector for each connected component consisting of thirteen Hamming distances.
- Finally, looking up the associated reliability statistic for each chosen template.

For each image, after creating the script ID vectors as described above, we performed a principal components analysis. This identified the main axes in the thirteen-dimensional space defined by the image's vectors. Reliability statistics were not taken into account in this analysis, although we intend to use them in the future.

We discarded all but the first three principal components, and normalized the values on each of the three to a range of 0 to 255. We then mapped the normalized values onto the colors red, green, and blue. Thus each connected component in the image was assigned a color that symbolized its location in a reduced, three-dimensional script identification space.

This process is demonstrated in Figures 5a-c, for the fragment of image 1 seen in Figure 4. The three figures illustrate the first three principal components in grayscale, with intensity indicating the value of the component. The second principal component (Fig. 5b) showed the sharpest separation between Arabic and Roman, with Arabic characters generally darker than Roman ones. The first and third components also showed some separation. For the first principal component (Fig. 5a), Arabic characters tended to be lighter than Roman ones. For the third principal component (Fig. 5c), most Roman characters had medium values, with Arabic characters darker or lighter.

The color image produced by mapping Figs. 5a-c to red, green, and blue, respectively, had Roman characters in green, shading to green-blue, and Arabic characters in red, purples, blues, and blacks. The touching characters *man* in the word *many* in the second row, which were unusually dark in the second component and unusually light in the third component, were bluer than the rest of the Roman in the color image.

This image, and the other six images described in this paper, can be viewed in color on the Internet at [3].

3 Script segmentation

We evaluated each image according to the degree of observed color separation between scripts. Images 1-3 had the sharpest separation. These images each contained Roman, plus a second script that was visually dissimilar to Roman: Arabic, Japanese, or Korean. In these images, each script was mapped to a distinct color or range of colors, indicating that the script identification vectors held the information required for script segmentation. Image 1 was described in the previous section. In image 2, Roman characters were in shades of red, purple, and red-blue, while Japanese characters were in greens and blues. In image 3, Roman characters were in greens and greenish browns, while Korean characters were in purple and purplish browns (recall that color images are available at [3]).

Within this set, images 1 and 2 had the sharpest script separation. In these images, individual words and phrases in one script stood out against a background from a contrasting script. Good examples of this were "SkyMiles® Medallion", near the top of the second column of image 2 (Fig. 2), and "dead march", near the bottom of the second column of image 1.

In images 1 and 2, the visualization also revealed differences within scripts. This happened most dramatically in image 2. In the Roman areas in this image, italicized characters were bluer, and bold characters pinker, than the other Roman characters. In the Japanese areas, *kanji* (Chinese root characters) were blue or blue-green, while *kana* (phonetic characters) were green or blue-green. In image 1, italicized Roman tended to be bluer than non-italicized Roman.

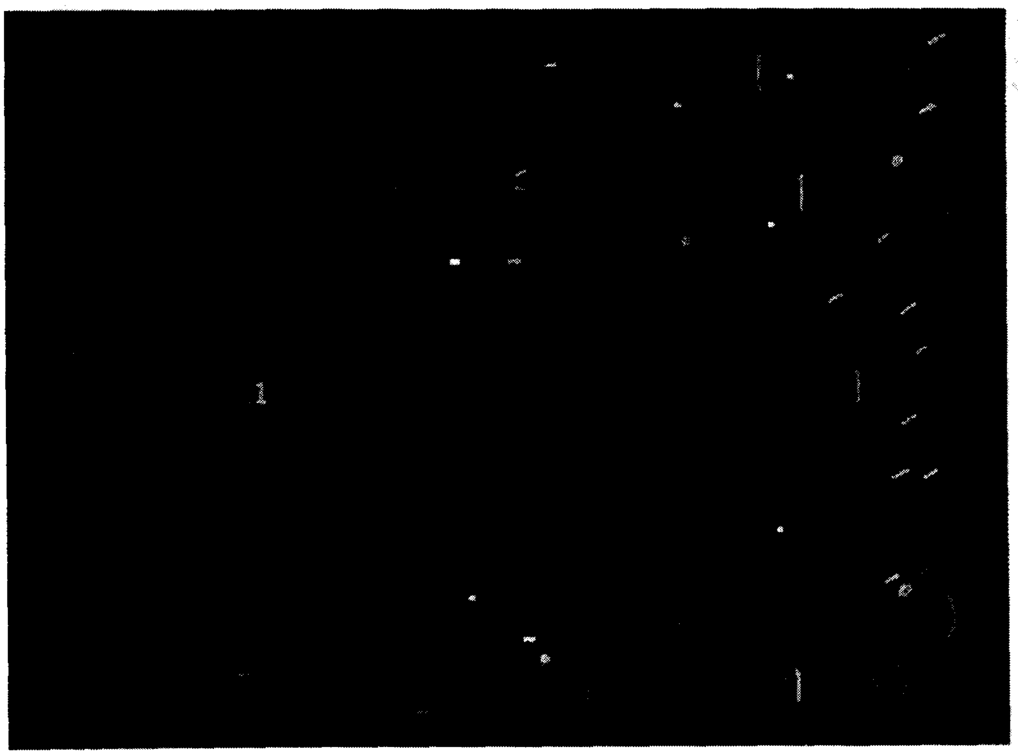


Figure 5a. First principal component for a fragment of image 1

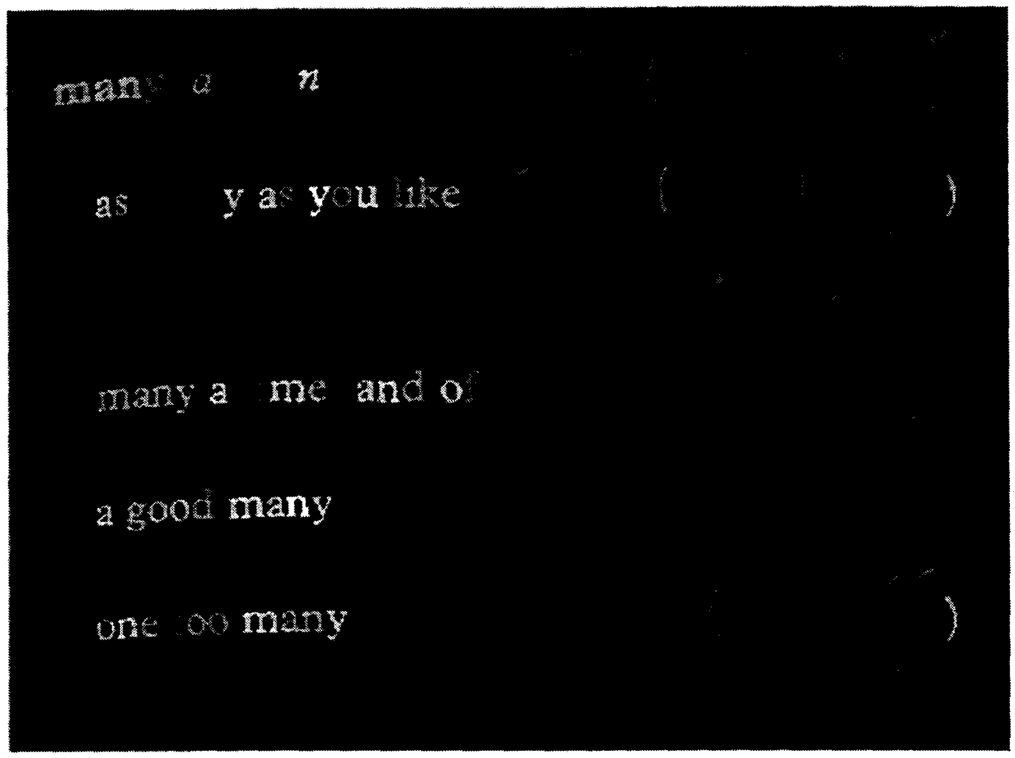


Figure 5b. Second principal component for a fragment of image 1

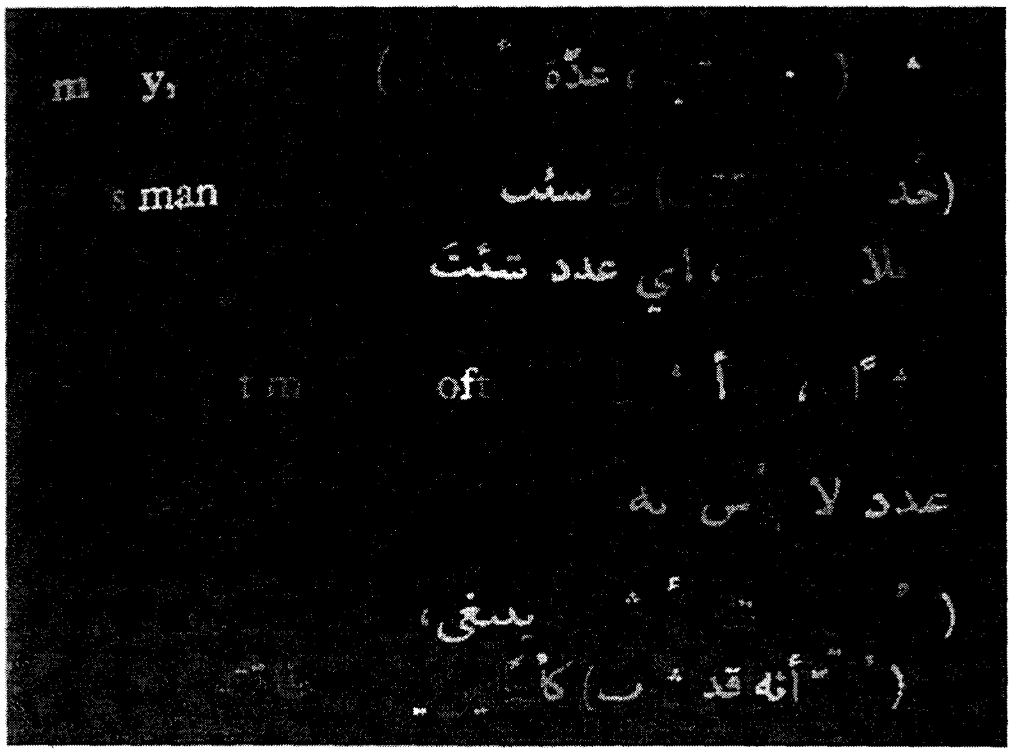


Figure 5c. Third principal component for a fragment of image 1

Images 4-5, which contained several scripts each, also showed clean script separation. The tendency in these images was for familiar scripts to be mapped to a single color, or narrow range of colors, while unfamiliar scripts were mapped to a mélange of colors. Thus in image 4, Roman was mapped to green, Arabic to blue/red, Armenian to brown, and Devanagari to dark blue. In image 5, Roman was mapped to dark reds and purples, Hebrew to green, and Chinese to blue, shading to brownish green. Coptic, Nasi, Javanese, Avestan, and Mayan, all unfamiliar scripts, were mapped to a mélange of colors.

Two scripts in images 4 and 5 differed from this general pattern. In image 4, Bengali was mapped to the same dark blue as Devanagari. This was a pleasing result given the visual similarity of the two scripts. Also in image 4, Ethiopic was mapped to a mélange of colors. This was surprising because Ethiopic was among our set of known scripts.

Images 6 and 7, which combined Roman with Greek or Cyrillic (both of which share several letters with Roman), showed the least script separation. In image 7, bright greens were generally Roman, and bright blues generally Greek; no other systematic differences were observable, and the overall effect was a mélange. In image 8, the only systematic separation was that bright greens were generally Roman. Most observable patterning pertained to individual characters: Cyrillic 'T' was always pale blue-green, and Cyrillic and Roman 'o' always red. Again, the overall effect was a mélange.

4 Conclusion

Our visualization efforts were encouraging. They suggested that, for all but closely related scripts, the script identification vectors, even as reduced by principal components analysis, contained the information needed to distinguish script regions on an individual page. For our best pairings, English/Arabic and English/Japanese, the vectors apparently also contained distinguishing font information.

Given this result, we expect that automatic segmentation algorithms based on the vectors will be fruitful in most cases. For more difficult script combinations, such as Roman/Greek and Roman/Cyrillic, we have two possibilities in mind for improving our current results. First, we plan to incorporate the reliability scores described in sections 1 and 2. These scores were crucial in making fine distinctions in our earlier work, and should help in the segmentation problem as well. The reliability scores could be used in visualization, by reducing the visual intensity of components whose best match overall was to a template with low reliability. Alternatively, each Hamming distance could be weighted by the reliability of the relevant template prior to any vector analysis. Second, in moving from visualization to an actual page segmentation algorithm, we plan to make use of the original script identification vector instead of the principal components. Perhaps the information lost in the principal components analysis is necessary for full-scale segmentation.

Acknowledgments

This work was performed under the auspices of the United States Department of Energy, contract W-7405-ENG-36.

References

[1] J. Hochberg, P. Kelly, T. Thomas, and L. Kerns, Automatic script identification from document images using cluster-based templates, *IEEE Trans. on Pattern Anal. and Mach. Intell.* **19** (1997) 176-181.

[2] R.S. Gilyarevsky and V.S. Grivnin, *Languages Identification Guide* (Nauka, Moscow, 1970).

[3] http://www.c3.lanl.gov/~judithh/LIFI/main.shtml

Proceedings of the 4th Conference on Applied Natural Language Processing, Stuttgart, Germany, pp 15-21, 1994

Language Determination: Natural Language Processing from Scanned Document Images

Penelope Sibun & A. Lawrence Spitz
Fuji Xerox Palo Alto Laboratory
3400 Hillview Avenue
Palo Alto, CA 94304 USA
sibun@pal.xerox.com

Abstract

Many documents are available to a computer only as images from paper. However, most natural language processing systems expect their input as character-coded text, which may be difficult or expensive to extract accurately from the page. We describe a method for converting a document image into *character shape codes* and *word shape tokens*. We believe that this representation, which is both cheap and robust, is sufficient for many NLP tasks. In this paper, we show that the representation is sufficient for determining which of 23 languages the document is written in, using only a small number of features, with greater than 90% accuracy overall.

1 Introduction

Computational linguists work with texts. Computational linguistic applications range from natural language understanding to information retrieval to machine translation. Such systems usually assume the language of the text that is being processed. However, as corpora become larger and more diverse this assumption becomes less warranted. Attention is now turning to the issue of determining the language or languages of a text before further processing is done. Several sources of information for language determination have been tried: short words (Kulikowski 1991, Ingle 1976); n-grams of words (Batchelder 1992); n-grams of characters (Cavner & Trenkle 1994); diacritics and special characters (Beesley 1988, Newman 1987); syllable characteristics (Mustonen 1965); morphology and syntax (Ziegler 1991). Each of these approaches is promising although none is completely accurate. More fundamentally, many rely on relatively large amounts of text data and all rely on data in the form of character codes (e.g., ASCII).

In today's world of text-based information, however, not all sources of text will be character coded. Many documents such as incoming faxes, patent applications, and office memos are only accessible on paper. Processes such as Optical Character Recognition (OCR) have been developed for mapping paper documents into character-coded text.

However, for applications like OCR, it is desirable to know the language a document is in before trying to decode its characters. There appears to be a fundamental Catch-22: natural language processing systems want to be able to work automatically with arbitrary documents, many of which may be available only on paper, and in the process, they minimally need to know which language or languages are present. The algorithms cited above can determine a document's language, but they require a character-coded representation of the text. OCR can produce such a representation, but OCR does not work well unless the language(s) of the document are known. So how can the language of a paper document be determined?

We have developed a method which reliably determines the language or languages of a document image. In this paper, we discuss Roman-alphabet languages such as English, Polish, and Swahili; see Spitz (1994) for a discussion of the determination of Asian-script languages. Our method finesses the problems inherent in mapping from an image to a character-coded representation: we map instead from the image to a *shape-based representation*. The basal representation is the *character shape code* of which there are a small number. These shape codes are aggregated into *word shape tokens* which are delimited by white space. From examining these word shape tokens we can determine the language of the document. An example of the transformation from character codes to character shape codes is shown in figure 1.

Character codes

```
Confidence in the international
monetary system was shaky enough be-
fore last week's action.
```

Character shape codes

AxxAAxxxx ix AAx ixAxxxxAixxxA xxxxAxxg xgxAxx xxx xAxAg xxxxgA Ax-Axxx AxxA xxxA'x xxAixx.

Figure 1: Character code representation and character shape code representation.

The shape-based representation of a document is proving to be a remarkably rich source of information. While our initial goal has been to use it for language identification, in support of downstream OCR processes, we are finding that this representation may itself be sufficient for natural language applications such as document indexing and content characterization (see Nakayama (this volume), Sibun & Farrar 1994). We find these indications exciting because OCR is an expensive, slow, and often inaccurate process, especially in the presence of printing and scanning artifacts such as broken or touching characters or skew or curvature of text lines. Thus, if our technique allows natural language processing systems to apply OCR selectively or to side-step OCR entirely, such systems will become faster, less expensive, and more robust.

In this paper, we first explain the background of our system that constructs character shape codes and word shape tokens from a document image. We next describe our method for language determination from this shape-based representation, and demonstrate our approach using only the three languages English, French, and German. We then describe an automated version of this process that allows us to apply our techniques to an arbitrary set of languages and show its performance on 23 Roman-alphabet languages.

2 Character shape codes and word shape tokens

Our determinations about document characteristics are made neither on the raw image[1] nor on the character codes by which the document can be represented. The determinations are made on a shape-based representation built of a novel component, the character shape code (Spitz 1993).

Four horizontal lines define the boundaries of three significant zones on each text line (see figure 2). The area between the bottom and the baseline is the descender zone; the area between the baseline and the top of such characters as x is the x zone; and the area between the x-height level and the top is the ascender zone.

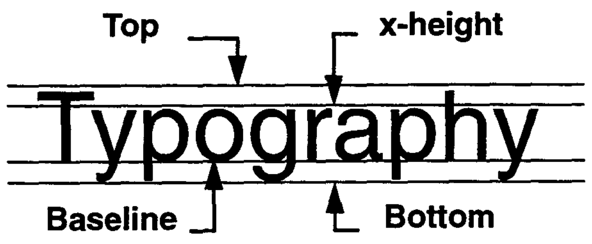


Figure 2: A text image showing the text line parameter positions: Top, x-height, Baseline and Bottom.

Characterizations of the number of connected components in a character cell and, in some instances, their aspect ratios, contribute to the coding. Thus most characters can be readily mapped from their positions relative to the baseline and x-height to a small number of distinct codes (see figure 3).[2]

Character shape code	Character
A	A-Zbdfhkltß0-9#$&/@|
x	acemnorsuvwxz
i	iáàâéèêîôûùñ
g	gpqyç
j	j
U	äëïöüÖÜ

Figure 3: Character shape codes.

2.1 Typesetting effects

Typesetters use different conventions. For example, in German text ü may be set as ue and ß may be set ss. Therefore, there may be several-to-one mappings of typeset information to character shape codes, since ü maps to **U** and ue to **xx**.

If this shape mapping can be done from document images, it can more trivially be accomplished from character-coded documents (e.g., ASCII, ISO-Latin-1, JIS, Unicode), providing, of course, that the method of encoding is known.

2.2 Computational complexity

Our approach takes on a much less difficult problem than does OCR. There is no need to investigate the fine structure of character images, the number of classes is small, and measurements are largely independent of font or typeface. As a result, the process of classifying text into character shape codes and aggregating those codes into word shape tokens is two to three orders of magnitude faster than current OCR technology.

3 Language determination

We have found that we can readily distinguish the language of a document for 23 Roman-alphabet (mostly European) languages from a relatively small text. This technique exploits the high frequency of short words in such languages and the diversity of their word shape token representations.

In this section, we describe our method for determining a document's language from the shape-based representation derived from the image (some of this

1. Document images may be obtained by scanning of paper documents, by retrieval from a document image database, or by digital rendering of a high level representation of the document.

2. This paper adopts the following conventions: monospaced to represent input characters, **boldface** to represent the character shape codes (**A**, **x**, **i**, **g**, **j**, **U**), and sans-serif to represent typographic conventions.

work has been reported in Nakayama & Spitz 1993). Our system learns how to discriminate a set of languages; then, for any input document, the system determines to which language it belongs. Our method uses the statistical technique of Linear Discriminate Analysis (LDA). First, we demonstrate the method using a hand-selected set of distinguishing features for a small set of languages. In section 4, we describe our process for automating the selection of distinguishing features across an arbitrary number of languages, and show the results on a corpus that includes documents from 23 languages.

Our initial set of discriminable languages comprised English, French, and German. To ascertain the set of discriminating features, we built a training corpus of approximately 15 scanned images of one-page documents for each language. We tokenized these images following the procedure described in section 2. This resulted in 7621 tokens from English, 6826 tokens from French, and 5472 tokens from German. We then ranked the frequency of word shape tokens across each corpus and noted the ten most frequent tokens. By comparing these top ten word shape tokens for each of the languages, we were able to select one per language that was both frequent in that language and less frequent in the other languages. Intuitively, each of these tokens is characteristic of its language; therefore, we call these *characteristic* tokens (see figure 4). The characteristic token for English is **AAx**; **AAx** constitutes 7% of the tokens in the

	English		French		German	
Token	**Rank**	Word	**Rank**	Word	**Rank**	Word
AAx	**1**	the				
xA	**2**	of	**7**			
Ax	**3**	to	**1**	la	**6**	
ix	**4**	is			**10**	
xxA	**5**	and			**3**	auf
xx	**6**		**2**	en	**2**	an
Axx	**9**		**3**	les	**1**	der
xxx	**8**		**4**	aux	**5**	wer
gxx			**5**	pas		
Aix					**4**	die

Figure 4: Most frequent word shape tokens in English, French and German: the top five for each language are shown; rankings of these are shown for the other languages when they fall in the top ten; shading indicates the characteristic token for each language; and common words that map to the top five tokens for each language are shown.

English corpus and is quite rare in the others. In the German corpus, **Aix** is not the most frequent token: **xx**, **xxA**, **Aix**, and **xxx** each make up about 3% of the corpus while **Axx** constitutes 6%. However, of the five, only **Aix** is rare in the other languages. While **Ax** is frequent in all three corpora, it is overwhelmingly frequent in French, where it makes up 11% of the tokens (vs. 4% for English and 2% for German). These differences in the distribution of the characteristic tokens in the three corpora are sufficient for LDA to correctly identify each language almost every time (see figure 5).[3] The documents are from the training corpus: by a process called cross-validation, each was removed from the training corpus one at a time and classified based on the discriminating results from training on the rest of the corpus.

		Language assigned		
		English	**French**	**German**
Language of document	**English**	13		
	French		17	1
	German	1		14

Figure 5: Number of documents from each corpus assigned to each language.

It may be noted that each of the top five word shape tokens in each of the English, French, and German corpora is a mapping of closed class words such as determiners, conjunctions, and pronouns. This is not surprising, since closed class words are frequent in European languages. Of course, other words map to these word shape tokens too. For example, in English, the word `flu` maps to **AAx**. However the overwhelming proportion of **AAx** tokens in the English corpus are mappings of `the`. Since `the` is such a common word in English, we can expect **AAx** to be characteristic of any shape-level representation of an English document. Similar situations obtain in the other languages.

While it may seem fortuitous that in English **AAx** is virtually always a mapping of `the`, unique word shape tokens are more common in Roman-alphabet languages than one might suppose. We mapped an English lexicon of surface forms into word shape tokens and discovered that 20% of the resulting word shape tokens were unique; examples include the surface forms `apple` and `apples`.

4 Automated language determination

In the previous section, we discussed the selection of discriminating word shape tokens by hand. We now describe our method for automating this process. We have been able to use this technique to discover a discriminating set of tokens for a large fraction of the languages written in the Roman alphabet. We initially tested this automated technique by recapitulating our

3. In the case of the German document that was misclassified, examination of the image reveals that, due to printing and scanning artifacts, many characters are artifactually touching each other.

work done by hand in discriminating English, French, and German. We then applied the technique to a 755-document corpus comprising 23 languages.

4.1 The automated method

While it is easy to hand-select a single discriminating token for each of a few languages, the task becomes more complex as the number of languages grows. Further, a single feature per language may no longer be sufficient; a profile, or vector of features, for each language would be more robust.

For the automated method, a corpus for each of the languages is scanned and tokenized, and the tokens are sorted by frequency. The *n* most frequent tokens for each corpus are selected. We apply stepwise discriminant analysis, a variant of LDA, to this token set: variables are selected one by one according to their ability to discriminate between languages. The optimal value of *n* has not yet been determined. We need to gather enough discriminating tokens to characterize the languages as completely as possible. However, if we use too many, the accuracy of the classification may actually be degraded; further, relatively uncommon tokens may improve performance on test data but may not work well in general. As we discuss below, $n = 5$ suffices for three languages, but may not be optimal for 23.

There are several considerations for ensuring that this process is robust. The size of the corpus for each language must be sufficiently large in terms of both the number of documents and the total number of word shape tokens. The number of documents must be large enough to enable the LDA testing procedure to systematically eliminate some of them for cross validation without skewing the overall characteristics of the corpus. The number of word shape tokens must be large enough to be reflective of the language in which the documents are written to allow for accurate comparison between languages. A further consideration is that the number of discriminating tokens used by the LDA system should be considerably smaller than the number of documents.

For our initial test we selected the five most frequent word shape tokens from each of English, French, and German; this formed a set of ten tokens (because of overlap between corpora). Using stepwise discriminant analysis, the system found the best way to use the tokens by selecting the single token that was most discriminating and then for each of the remaining tokens adding the next most discriminating tokens given the ones that had already been selected. This resulted in a ranking of nine discriminating tokens (**Ax**, **xA**, **ix**, **Aix**, **Axx**, **xx**, **AAx**, **xxA**, **xxx**). The tenth was not found to improve the reliability of the discrimination; in fact accuracy peaked at four tokens.

We compared the performance of the automated system with that using the hand-selected tokens. When the top three automatically-selected tokens were used, performance was comparable to that of the three hand-selected tokens. Interestingly, there is no overlap in the misclassification of documents. Using four automatically-selected tokens, the system classified all but one document correctly (see figure 6).

		Language Assigned		
		English	**French**	**German**
Language of document	**English**	13		
	French		18	
	German	1		14

Figure 6: Assignment of documents to language using four automatically-selected discriminating tokens.

4.2 Automated determination for many languages

We have constructed a database of 755 one-page documents in 23 languages including virtually every European language written in the Roman alphabet. There are 18 Indo-European languages: Afrikaans, Croatian, Czech/Slovak[4], Danish, Dutch, English, French, Gaelic, German, Icelandic, Italian, Norwegian, Polish, Portuguese, Rumanian, Spanish, Swedish, and Welsh. There are two Uralic languages: Finnish and Hungarian. Finally, we include three languages from disparate families: Turkish, Swahili, and Vietnamese.

To construct a set of discriminating features, we selected the five most frequent word shape tokens from each language. Because of overlap, this resulted in 23 tokens. Some of these discriminating tokens have a high frequency across languages; in fact, **xx** appears in the top five of 22 of the languages we examined. However, even when we consider 23 languages, there are eight tokens appearing in the top five of one language which do not appear in the top five of any others. (This does not mean of course, that these tokens do not appear in other languages at all, but simply that they are relatively much less frequent.) The 23 tokens comprise the set (**x**, **xx**, **xxx**, **xxxx**, **i**, **ix**, **xi**, **xix**, **A**, **AAx**, **Ax**, **AxA**, **AxAx**, **Axx**, **Axxx**, **xA**, **xxA**, **Ai**, **Aix**, **g**, **gx**, **xg**, **xxg**, **jx**).

As before, we used LDA to build a statistical model of the language categorizations, and by cross validation tested the accuracy of the model (see figure 7). Our overall accuracy is better than 90%, while the accuracy for individual languages varies between 100% and 75%, with an outlier of 44% for Czech/Slovak. Examination of misclassifications proves somewhat instructive, as can be seen in the confusion matrix in figure 8. For example, Dutch and Afrikaans are closely related languages, and the only error in either language is the categorization of one Afrikaans document as Dutch. Among the five

4. We initially considered Czech and Slovak as separate languages, but this yielded worse results than combining them. We feel our decision was legitimate because "Slovak is similar enough to Czech to be considered by some as merely a dialect" despite "the existence of slightly different alphabets, as well as distinct literatures" (Katzner 1986, p 91).

Romance languages – French, Italian, Spanish, Portuguese, Rumanian – nine of the ten classification errors are within that language family. For the Scandinavian language family – Danish, Norwegian, Swedish, and Icelandic – the pattern is less clear. Two Norwegian documents are classified as Icelandic, but the three other errors in that family are classifications outside of the family.

Language	abbr	Acc (%)	Language	abbr	Acc (%)
Afrikaans	af	97	Italian	it	95
Croatian	cr	100	Norwegian	no	95
Czech/Slovak	cs	44	Polish	po	100
Danish	da	96	Portuguese	pt	96
Dutch	du	100	Rumanian	ru	93
English	en	95	Spanish	sp	95
Finnish	fi	75	Swahili	sa	97
French	fr	92	Swedish	sw	98
Gaelic	ga	86	Turkish	tu	93
German	ge	97	Vietnamese	vi	100
Hungarian	hu	94	Welsh	we	97
Icelandic	ic	96			

Figure 7: Language detection accuracy. The abbreviations shown are used as indices in figure 8.

Croatian, Czech/Slovak, and Polish are all Slavic languages; Hungarian and Finnish are related to each other but not to any other European languages. However, there is a large cluster of errors within the set of these five languages. Most of these errors are for Czech/Slovak documents; in fact, Czech/Slovak was recognized far less accurately than any other language and it is unclear why. It may be the case that many of these documents are of poor quality. Seventeen of the 69 errors seem to be random; while we are working to reduce such errors, it is unlikely that we can eliminate them entirely. It is possible that 23 discriminating tokens is not sufficient; since the accuracy has been improved by the addition of each new token, adding several more may continue the improvement.

4.3 Discussion of methodology

While LDA has proved adequate, there are some drawbacks to this technique. We are somewhat disappointed by the system's accuracy. Examination of token frequencies suggests that the profiles for each language are distinct enough that 90% should be a lower bound on classification accuracy. However, for several languages the accuracy was much lower, and for many more it was not much better than 90%. A more troubling problem is the instability of the model. When we add or delete languages, overall accuracy fluctuates between 80% and 93%. This suggests that removing a language affects the typical distribution across all languages, which should not be the case. It is difficult to identify the underlying causes of both of these observations. Finally, the results of LDA are difficult to interpret. All these considerations suggest that LDA may not be the best technique to use. Therefore, we are exploring alternative statistical models, such as classification trees, to find an approach that is more robust for our task.

5 Comparison with other methods

It is difficult for us to compare our approach to other methods of language determination. Most sources we have found are simply guides for librarians or translators. For example, Ingle (1976) found that the presence or absence of specific one- or two-character words suffices to distinguish among 17 Roman-alphabet languages. There are several implemented systems, some of which report on their accuracy, but none is addressing exactly the same problem as ours: all work from character-coded text. However, it is useful to get a ballpark estimate of the accuracy to be expected of character-based systems.

Batchelder (1992) trained neural networks to recognize 3-6 character words from 10 languages. While her networks had high accuracy in recognizing words from the training set, their best-case performance on untrained words was 53%, thus making accurate determination of a document's language highly unlikely.

Cavner and Trenkle (1994) used n-grams of characters for n = 1 to 5. Their task was not language determination *per se*, but determining to which country's newsgroup (in the netnews soc.culture hierarchy) a document belonged. In each newsgroup, the documents were written in either English or other language(s). For documents longer than 300 characters, the system determined the correct newsgroup with 97% accuracy when using the 100 most frequent n-grams. These results are good, but the technique should be tested on a set of documents for which the languages are known and the topics are varied.

Kulikowski (1991) used a semi-automatic method to determine a profile of frequent 2-3 character words for nine languages. He claims at least 95% accuracy for determining that a single-language document is in one of the nine languages or in none of them. Unfortunately he does not expand on this claim. Henrich (1989) used criteria such as language-specific word-boundary character sequences and common short words to determine the language of sentences in English, French, or German.

Mustonen (1965) used discriminant analysis to distinguish English, Swedish, and Finnish words. His system, which used 43 discriminating features, such as particular letters and syllable types, performed with 76% accuracy. This relatively poor performance is probably due to the data being isolated words rather than documents, though it may also be due to overfitting of the test data by too many features (see section 4.1).

We would like to emphasize that our statistics on word shape token distribution across the various lan-

	Detected Language																						
	en	ge	du	af	fr	it	sp	pt	ru	da	no	se	ic	ga	we	cr	cs	pl	hu	fi	tu	sa	vi	Error Total
en	36	1			1																			2
ge		29												1										1
du			28																					0
af			1	29																				1
fr					23			1						1										2
it						35		1	1															2
sp							39	2																2
pt							1	25																1
ru						3			38															3
da					1					25														1
no											39		2											2
se												40								1				1
ic													23							1				1
ga						2			2				1	31										5
we															30							1		1
cr																33								0
cs									1							6	24	16	3	5				31
pl																		28						0
hu																		2	29					2
fi																	6		2	24				8
tu							1													1	28			2
sa																					1	37		1
vi																							13	0
	0	1	1	0	2	5	2	4	4	0	0	0	3	2	0	6	6	18	3	9	1	1	0	69

Figure 8: Confusion matrix showing detection accuracy between languages. Numbers on the major diagonal indicate the number of correct classifications for each language. Numbers off the diagonal indicate classification errors.

guages are generated entirely from scanned images of text. We feel this is important because the text whose language we are trying to identify should not be systematically different in any way from the texts from which the discriminate analysis was generated. For example, typographic conventions such as a ligature between a vowel and an acute accent (as in characters like à) cause the character shape code recognizer to classify these characters as **A**. However, if we were working from encoded on-line corpora we would "know" that such a character should be classified as **i**.

6 Conclusion

We have described our method for generating word shape tokens from images and have shown how this shape-level representation of the text can be used for important tasks such as determining the language or languages of a document. We have shown that the method can discriminate among 23 languages with high accuracy.

Since our approach is statistical, the more text our system sees in a document image, the more reliably it can determine the document's language. So far, we have not tried to determine the language of a document shorter than 27 words, and most of the documents we work with are a few hundred words long (2000-3000 characters). We are investigating the lower bound on the length of texts whose language we can reliably determine. In the ideal case we would be able to detect the presence of a very few words of a secondary language interpolated into a document predominated by another language.

In other work, we are using the shape-level representation as input to higher-level natural language processing systems for rudimentary content analysis. However, many sorts of information, particularly style characteristics, can be derived from the shape-level representation directly. For instance, since the number of character shape codes extracted form a document is comparable to the number of characters, characterizations about word length in a shape-level representation apply as well to the character-coded version of the document. This word length characterization is not perfect: ligatures introduce some uncertainty. Additionally, braces, brackets, and parentheses which are typically set contiguous with words, are currently mapped to A, this will affect word length counts. We are refining the mapping to account for these delimiting characters.

Acknowledgments

We thank David Hull for his expertise in developing statistical methods and help in explaining them, Arlene Holloway for patiently scanning and processing most of our document image database and helping to analyze the results, Marti Hearst and Michael Berch for comments on drafts of this paper, and Jussi Karlgren for a last-minute reference.

Bibliography

Batchelder, Eleanor Olds, *A Learning Experience: Training an Artificial Neural Network to Discriminate Languages,* Unpublished Technical Report, 1992.

Beesley, Kenneth R., *Language Identifier: A Computer Program for Automatic Natural-Language Identification of On-line Text,* Language at Crossroads: Proceedings of the 29th Annual Conference of the American Translators Association, 12-16 Oct 1988, pp 47-54.

Cavner, William B. & Trenkle, John M., *N-Gram Based Text Categorization,* Proceedings of the Third Annual Symposium on Document Analysis and Information Retrieval, 11-13 April 1994, pp 161-169.

Henrich, Peter, *Language Identification for the Automatic Grapheme-to-Phoneme Conversion of Foreign Words in a German Text-to-Speech System,* Proceedings of Eurospeech 1989, European Speech Communication and Technology, Paris, Sept. 1989, pp 220-223.

Ingle, Norman C. *A Language Identification Table,* The Incorporated Linguist vol. 15 no. 4 pp 98-101, 1976.

Katzner, Kenneth, *The Languages of the World,* London: Routledge, 1986.

Kulikowski, Stan, *Using Short Words: A Language Identification Algorithm,* Unpublished Technical Report, 1991.

Mustonen, Seppo, *Multiple Discriminant Analysis in Linguistic Problems,* Statistical Methods in Linguistics, No. 4, Skriptor Fack, Stockholm, 1965, pp 37-44.

Nakayama, Takehiro & Spitz, A. Lawrence, *European Language Determination from Image,* Proceedings of the International Conference on Document Analysis and Recognition, 20-22 Oct 1993, pp 159-162.

Nakayama, Takehiro, *Modeling Content Identification from Document Images,* this volume.

Newman, Patricia, *Foreign Language Identification: First Step in the Translation Process,* Proceedings of the 28th Annual Conference of the American Translators Association, 8-11 October 1987, pp 509-516.

Sibun, Penelope & David S. Farrar, *Content Characterization Using Word Shape Tokens,* Proceedings of the 15th International Conference on Computational Linguistics, Kyoto, Japan, 1994, pp 686-690.

Spitz, A. Lawrence, *Generalized Line, Word and Character Finding,* Progress in Image Analysis and processing III, Impedovo, Ed., World Scientific, 1993, pp 377-383.

Spitz, A. Lawrence, *Script and Language Determination from Document Images,* Proceedings of the Third Annual Symposium on Document Analysis and Information Retrieval, 11-13 April 1994, pp 229-235.

Ziegler, Douglas-Val, *The Automatic Identification of Languages Using Linguistic Recognition Signals.* Dissertation, State University of New York at Buffalo, 1991.

Proceedings of the 3rd Annual Symposium on Document Analysis and Information Retrieval, Las Vegas, Nevada, April 1994, pp 229-235

Script and Language Determination from Document Images

A. Lawrence Spitz

Fuji Xerox Palo Alto Laboratory
3400 Hillview Avenue
Palo Alto, CA 94304 USA
spitz@pal.xerox.com

Abstract

We have developed techniques for distinguishing which language is represented in an image of text. This work is restricted to a small but important subset of the world's languages, using techniques that should be applicable across much more comprehensive samples. The method first classifies the script into two broad classes: European and Asian. This classification is based on the spatial relationships of fiducial points related to the upward concavities in character structures. Language identification within the Asian script class, (Japanese, Chinese, Korean) is performed by analysis of the optical density distribution of the text images. Within the European script class, language identification is described in separate papers.

1 Introduction

Worldwide there are many different languages in common use and many different scripts in which these languages are typeset. In a document processing system that includes automated document recognition capabilities, early detection of the language or languages present in a document has implications both in the selection of the proper character recognition service and in the resolution of errors produced by character recognition.

We have made some progress toward enabling document recognition systems to handle a mixture of documents in different languages, or individual documents incorporating more than one language. Our system is currently limited to the processing of seven languages: English, French, German, Russian, Japanese, Chinese and Korean.

We have developed alternate methods of performing the Asian vs. European script determination. In one method this determination can be performed without any knowledge of the layout of the page, text flow direction or any restriction on the mixture of character sizes in the image.

In the second method, after connected components analysis, it is necessary to understand the text line layout and text line orientation and the character sizes are assumed to be homogeneous within a single text line.[1] Knowledge of these parameters is also required for European language classification.

Once the script classification is complete, Asian language classification can proceed independent of the determination of text line characteristics.

European language classification relies on text line characteristics and the relationships of the connected components to the character cells. This classification is fully described in Nakayama & Spitz.[2]

The flow of information in the process of script and language classification is shown in Figure 1.

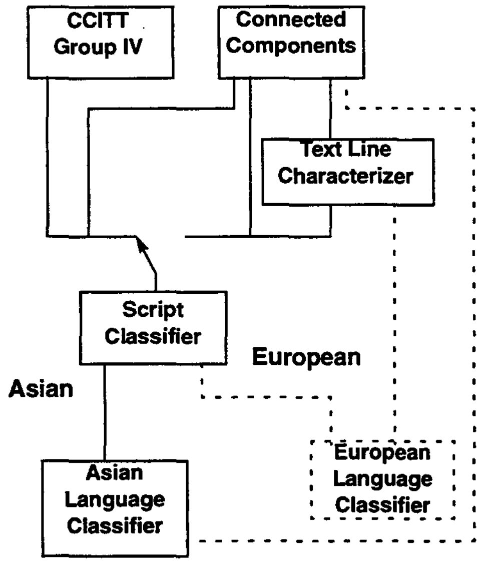


Figure 1: This schematic diagram shows the alternate information flow for gross script classification followed by language classification. The elements shown in dashed lines are not described in this paper.

2 Image processing

All non-text information is removed from the document image and the image is skew corrected, if necessary.

All analysis is based on abstractions of the image rather than on the binary bitmap itself. Two abstractions are used: lists of connected components[3] and the CCITT Group IV compressed representation [4][5]. Both of these representations are loss-less; that is, the original bitmap can be reproduced without error from each.

2.1 Connected component generation

The set of 8-connected components in the document image is calculated. The representation of each connected component includes the coordinates and dimensions of the bounding box, and a list of the individual runs of black pixels that make up the component as well as the 'center of mass' or centroid of each connected component.

2.2 Upward concavity

It is trivial to examine sets of runs within the connected component to determine the presence and location of upward concavity. Where two runs of black pixels appear on a single scan line of the raster image, if there is a run on the line below which spans the distance between these two runs, an upward concavity is formed on the line.

The reference coordinate system used for defining the spatial distribution of concavities is the character cell baseline.

The alternate method of processing the image to find the upward concavities takes advantage of the implicit encoding of concavity position inherent in the CCITT Group IV compression algorithm commonly used for image data storage as well as for facsimile transmission.

The reference coordinate system used for defining the spatial distribution of CCITT pass codes is the connected component centroid position.

2.3 Pass code generation and labeling

As described in Spitz [6], it is possible to locate the position of CCITT pass codes in the image and to label each pass to indicate the color (white or black) of pixels being processed when the pass code is encountered. Using the CCITT black pass positions takes advantage of a pre-calculated transform of the image. Figure 2 shows the word "Laboratory" and the positions of the black passes.

For example, the "a"s show two black passes each at roughly equal distances below the centroid, while the "b" shows one black pass at a greater distance below the centroid and one a short distance above the centroid. Note that the positions of the black passes are directly related to the bottoms of upward concave black structures.

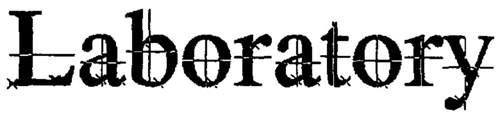


Figure 2: This document image segment shows the positions of connected component centroids, marked by the cross-hairs, and black pass codes, marked by small x's

For Asian script the occurrence of upward concavities is significantly different from that for European script. Because of the more complex characters, incorporating more instances of enclosed white space, there are many more concavity occurrences per character. Figure 3 shows an example.

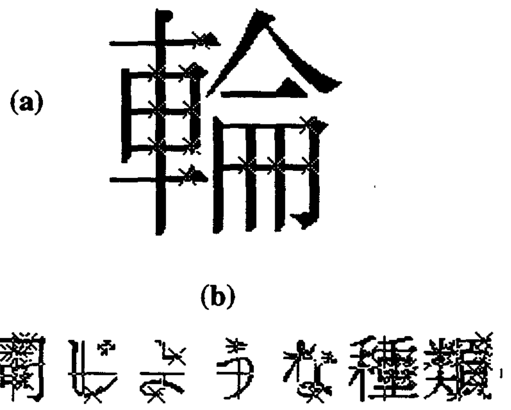


Figure 3: Locations of upward concavities in a single, relatively complex Kanji character (a), and of centroids and upward concavities in portion of an Asian document are shown.(b)

3 Gross script classification

In earlier work, Spitz [7] described a method of classifying individual text lines as being either English or Japanese in documents restricted to contain only those two languages; hence the technique was really a script classifier rather than a language classifier.

The technique described in [7] relies on the distribution of an index of optical density and relies on the fact that Japanese comprises Kanji which tend to be complex, and therefore optically dense, and Kana which tend to be simple and therefore optically light, while the Roman script that is used to set English has a more consistent optical density, character by character.

In this paper we approach the problem somewhat differently; we divide the scripts into two gross classes: European, comprising Roman and Cyrillic scripts, and Asian, comprising Kanji, Kana and Hangul. This gross classification is accomplished on the basis of the vertical distribution of upward concavities. The position of these concavities may in turn be derived from examination of the individual runs that comprise the connected component or from the position of the black pass codes referenced to connected component centroids.

For each connected component the positions of all associated black passes are calculated. Note that fully or partially overlapping connected component bounding boxes will result in some passes being associated with more than one connected component.

Next we relate the positions of upward concavities to the position of a stable fiducial point. In the instance where we do not have page layout information or where we have a mixture of a small number of isolated characters, perhaps of varying size, we use the centroid of the associated connected component. Where text line information is available and where we do not already have the CCITT representation pre-calculated, we use the baseline position for the relevant text line as our fiducial level. It is sufficient to look at the vertical distribution of upward concavity position with regard to the fiducial level only.

Figure 4 shows vertical profiles of upward concavity population for typical European and Asian documents. The units above and below the centroid are normalized to vertical extent of the aligned connected components in the instance of pass-based classification and to character cell height in the instance of text line-based classification. The distributions are normalized to the mean. In this figure a single document sample is shown for each script.

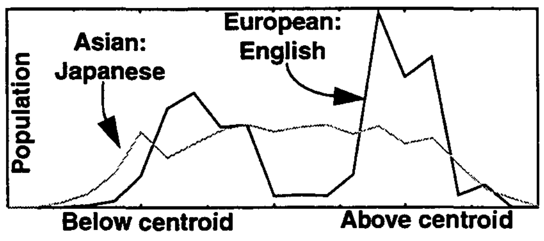


Figure 4: Spatial distributions of black passes with respect to connected component centroid vertical position for European and Asian documents are characteristically different

A display of the distributions by language is shown in Figure 5. The left column shows the European languages (Roman and Cyrillic scripts) and the right column shows the Asian languages (Kanji, Kana and Hangul scripts).

The distribution of upward concavities relative to character cell baselines shows an identical shape but with a possible offset.

Discriminating between the two classes of distribution is relatively easy: we use a simple measure of variance, with the European distribution showing the greater variance, reflecting the clustering of vertical position of upward concavities. This variance measure is insensitive to offset between the distributions.

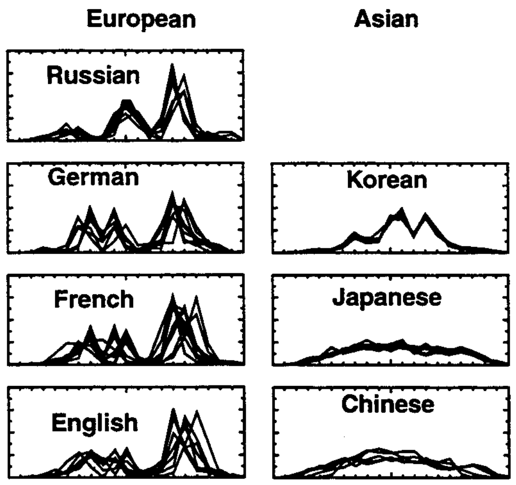


Figure 5: Distributions of black pass code vertical distance with respect to connected component centroid position from multiple documents in each of seven languages are shown.

4 Language classification

We deal with the problems of classifying Asian and European languages in completely separate ways.

4.1 Asian

Examples of Japanese, Chinese and Korean text are shown in Figure 6. Note that the Japanese characters are a mixture of relatively light characters (typically Kana) and relatively dense characters (typically Kanji) The Chinese text is composed of predominantly relatively dense Kanji characters. The Korean text is made up of both locally dense and locally light characters.

4.1.1 Optical density function

A function is derived that reflects the perceived optical density of the character cells as a function of reading order, that is, across each character cell within each text line under consideration. Since this technique is

原子核の質量は，原子の質量の大部分を占めるので，

中國古時候，有一個人姓馬，名字叫書田。

새계에서 가장 정조관념이 굳다는 것이 중국의 여성들이다.

Figure 6: Text fragments from Japanese, Chinese and Korean documents are shown.

only applied to Asian characters, where by typographic convention the inter-character spacing is large enough that unintentional ligatures are unlikely, there is no need for a character splitting process to be applied.

Within each character cell, the number of "on" pixels is counted by summing the lengths of the runs that comprise all of the connected components within the cell. This sum represents the mass of the character and its value is represented across the entire horizontal span of the character cell. The inter-component and inter-character spaces and the inter-word spaces found in Hangul are skipped. The resulting function reflects the reading order distribution of optical density.

Again looking at the Japanese text sample, digitized at 300 pixels per inch, we derive the optical density function for a fragment of that image as shown in Figure 7.

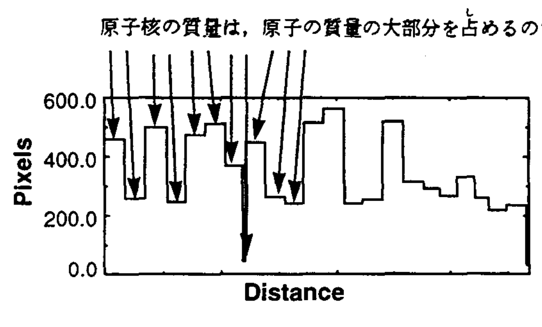


Figure 7: The optical density function for part of a line of Japanese text is shown

The vertical axis represents the number of "on" pixels in the character cell, while the horizontal axis reflects the distance (not counting inter-character spaces) along the assemblage of text lines in units of pixels.

4.1.2 Optical density distribution

We collected statistics about the frequency of occurrence of the various levels of optical density found in the optical density function. Histograms of the density functions when applied to Japanese, Chinese and Korean documents show characteristically different distributions. Note that the optical density function for Korean documents shows a distinct bi-modal nature, with the low density mode smaller than the high density mode. The distributions for the Japanese documents might also be characterized as bi-modal, but in this instance the relative heights of the modes is reversed: the low density mode is greater than the high density. In the Chinese document there is only one significant mode.

4.1.3 Classification

We identified three prototypical node positions by examination, roughly corresponding to the lower peak in the Korean and Japanese, the null in the Korean, and the peak in the Chinese and Japanese. We integrate the areas under the curves in these three areas.

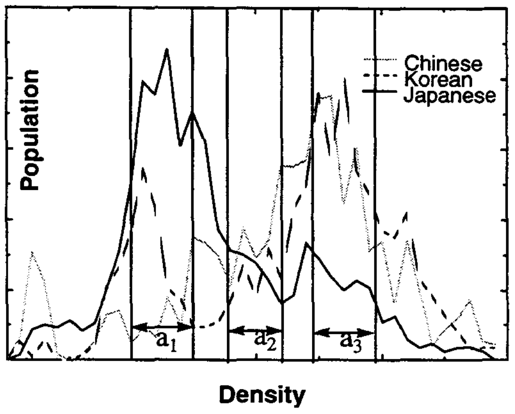


Figure 8: Specific areas of the histograms of the distribution of optical density are characteristic of different Asian languages.

We apply linear discriminant analysis (LDA) to the multivariate area data to resolve into multiple language classes. Since we are attempting to select one of three classes, two LDA (V_1,V_2) variables are needed. See Figure 9.

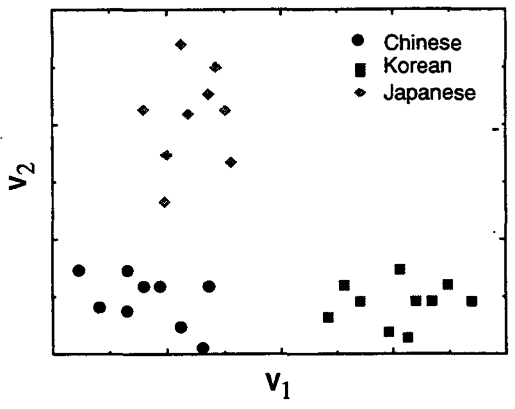


Figure 9: Clusters are visible in the scatter plot of locations of the test documents in LDA space

Classification is performed by measuring the distance in LDA space from the centroids of the clusters formed by the points derived from documents of the various languages.

4.2 European language classification

Nakayama & Spitz [2] have developed a technique to accurately detect the language of the text based on gross spatial characterizations about the rendering of the character coded data. This technique is based on the frequency of occurrence of particular character shape tokens that are aggregations of individual character shape codes into word-sized quanta. Sibun and Spitz [8] have extended this work to include more languages and to provide a mechanism for the automated selection of the optimal discriminating set of tokens.

5 Granularity

At the present time we are able to classify the script of a document as being either Asian or European on the basis of as little as two lines of text and we are able to classify the language of a document image containing Asian script with perfect accuracy on text samples as small as six lines. Table 1 shows the increase in error rate as the number of lines decreases.

Lines	Samples	Error rate (%)
10	72	0.0
8	95	0.0
6	140	0.0
4	217	0.5
2	447	0.2
1	910	15.8

Table 1: The error rate of language classification relates to the number of lines of text processed.

We expect to extend this work by developing tools for script classification of single line, and ultimately single word, text images, but since both Japanese and Chinese are set without word spaces, it is likely that text line granularity will be the limiting case.

Within Asian script, separation of Hangul from Chinese and Japanese using the technique described in this paper becomes problematic at small granularity, as the optical density distributions incorporate data from smaller and smaller samples.

Discriminating between Chinese and Japanese will rely upon supplemental recognition of Hiragana and Katakana as separate scripts, with the presence of these Kana used as an indicator that the text is Japanese. Moreover, since Japanese and Chinese both incorporate Kanji script, it will be very difficult to discriminate Chinese from Japanese if the fragment contains only Kanji.

6 Conclusion

We have assembled a system that can process documents in seven languages. Using the techniques described here, we are able to accurately determine the script class, and if the script is Asian, to classify the language of the document. Work reported elsewhere includes the basic image processing functions needed to support these determinations and also the process of classifying European languages.

We hope to extend these techniques to handle more scripts and more languages. Specifically we expect to add Hebrew and Greek scripts as well as Devanagari and Arabic.

7 Acknowledgment

I would like to thank David Hull of Stanford University for his assistance in automating the process of linear discriminate analysis.

8 References

[1] A. Lawrence Spitz, "Generalized Line, Word and Character Finding", *Proceedings of the 7th International Conference on Image Analysis and Processing*. Bari, Italy, 1993.

[2] T. Nakayama & A. Lawrence Spitz, "European Language Determination from Image", *Proceedings of the International Conference on Document Analysis and Recognition*, October 1993

[3] C. Ronse & P. A. Devijver., "Connected Components in Binary Images: The Detection Problem" *Research Studies Press*, 1984

[4] R. Hunter, A. H. Robinson, International Digital Facsimile Coding Standards. *Proceedings of the IEEE*, Vol. 68, No. 7, July 1980, 854-867

[5] CCITT Recommendation T.6 Facsimile Coding Schemes and Coding Control Functions for Group 4 Facsimile Apparatus, Terminal Equipment and Protocols for the Telematic Services, Vol. VII, Fascicle VII.3, Geneva 1985

[6] A. Lawrence Spitz, "Skew Angle Determination in Group 4 Compressed Document Images", *Proceedings of the Symposium on Document Analysis and Information Retrieval*, April 1992

[7] A. Lawrence Spitz. "Multilingual Document Recognition." in *Electronic Publishing, Document Manipulation & Typography*, R. Furuta (ed.), Cambridge University Press, 1990, pp 193-206

[8] P. Sibun and A. Lawrence Spitz, "Language Determination: Natural Language Processing from Scanned Document Images", *Proceedings of the Association for Computational Linguistics*, July 1994 (submitted)

Fly-Away Portable Intelligent Understanding System

Peter A. Roberts
Kathpal Technologies, Inc.
2230 Gallows Road, Suite 380
Dunnn Loring, VA 22027
kti@kathpal.com

Abstract

The acquisition of documents that contain important information but which are written in languages other than English has created requirements for Intelligent Document Understanding (IDU) systems that can translate documents into English without the aid of linguists. A portable IDU system may prove to be critical for field operations where documents cannot be physically removed from the facility or the operations require an immediate translation of the documents. The Federal Intelligent Document Understanding Laboratory (FIDUL) contracted Kathpal Technologies Incorporated to design and integrate a portable IDU system that can handle Russian, Spanish, and Chinese documents. The system now provides FIDUL with a valuable means to demonstrate the state-of-the-art to intelligence analysts.

The Portable IDU system is an integrated hardware and software solution that takes advantage of available state-of-the-art hardware and software. The Russian IDU system was designed and built using only Commercial Off The Shelf (COTS) hardware and software. The hardware fits into an airline carry on case that weights less than 50 pounds. The system operates on an Intel base computer, scans up to 8.5" x 11" single-sided documents with separable pages and can handle different power configurations, plug types and communication adapters.

The Portable IDU system was designed to be flexible in meeting a variety of user translation requirements. The system was also designed to provide the best translation based on available COTS software and to reduce the number of steps to translate a document. The system is composed of three different groupings of software. The first group of software handles the capture and conversion of the document image (i.e., Optical Character Recognition and Machine Translators). The second and third group of software handles the document management and telecommunications operations, respectively.

The Russian IDU system was completed at the end of 1996 and work began on the Spanish System subsequently. The Spanish system is scheduled to be completed at the end of February of 1997 and the Chinese system is scheduled to be completed in May of 1997. Some lessons learned from the building and development of the Russian system were:

- A variety of software is available but, it varies in terms of its performance;
- Software often lags behind hardware developments; and
- Technical support is marginal at best.

MEDICAL DATABASE INPUT USING INTEGRATED OCR AND DOCUMENT ANALYSIS AND LABELLING TECHNOLOGY

G.R.THOMA, D.X.LE
National Library of Medicine
Bethesda, Maryland 20894, USA

The Communications Engineering Branch of the National Library of Medicine is developing an automated system, code named MARS for *Medical Article Record System,* to handle the input of 500 bibliographic records a day. The system is being designed to replace (with minimal human intervention) a completely manual and labor-intensive keyboarded entry of records (a procedure followed for many years) with a system that processes images taken from a journal, segments and labels the image contents, extracts and reformats the citation information from the article, transmits the appropriate portions into the indexing system, and collates all the citation and indexing information into the appropriate citation database (MEDLINE, its successors, as well as specialized databases).

MARS consists of multiple workstations of two types: *scanner, and proofing and editing.* The scanner workstation operator scans the article into the system and the proofing and editing workstation operator reviews the entire citation after both the page segmentation and labelling process and the OCR process are completed. In addition, the system requires four servers: a network file server, a page segmentation and labelling server, a voting OCR server, and a spell check and special character recognition server. All workstations and servers are networked via a LAN.

Briefly, the MARS system works as follows. The scanning operator scans the first pages of articles and the bitmapped TIFF files are sent to the network server. The page segmentation and labelling server zones the bitmapped text lines and labels the text lines as belonging to "author", "affiliation", "article title", "abstract", etc. The voting OCR server performs text conversion, and produces a text file of the labelled zones. The spell check and special character recognition server performs spell check on the text and identifies any special characters such as Greek letters, symbols, and characters with diacritical marks. At this point, the OCR output text and the corresponding bitmapped TIFF file are available for validation and proofing. Following this step, the text file are FTP'ed to the NLM mainframe computer, and those completed records may be accessed later by indexers who add the appropriate descriptive information such as Medical Subject Headings.

The page segmentation and labelling technology is based on an adaptive smearing bottom-up approach for block segmentation, a classification of blocks as text blocks or non-text blocks using neural network models, and a set of rules for block labelling that is derived from an analysis of the page layout for each journal (journal profile).

Three types of errors were considered for OCR software package selection. They include the detected (highlighted) errors, the highlighted correct words (false alarms) and the undetected errors. The selection of the OCR package for the MARS system was based on minimizing the undetected error rate since this is the factor that gives a high level of confidence that the proofers need not compare the converted text against the original printed material, a highly labor-intensive step.

Performance Evaluation

Title: Results of a Technology Transfer Assessment of a Russian Language System

Authors: Mike Lutjen
Federal Intelligent Document Understanding Laboratory
Vienna, VA 22182
(703) 827-2220

Tamra Hall
Federal Intelligent Document Understanding Laboratory
Vienna, VA 22182
(703) 827-2220

Related Application Area: Document Processing, Organization, and Management

Sponsor: Federal Intelligent Document Understanding Laboratory

Description:

This abstract contains the results of studying a manual business process and making a transferability assessment of technology available to automate that process. The manual process under investigation is the daily review by Russian linguists of hard copy Russian documents and the selection of articles from those documents that, by nature of their content, should be translated into English. The technology assessed is a prototype Russian Language System that consists of a scanner, scanning and optical character recognition software (Cuneiform 2.95), and a software package that allows storage, browsing and information retrieval (Euphrates 1.0).

Given the conditions where post-editing of recognized document pages is not required, the transfer of this technology is recommended.

Under the conditions where post-editing of all recognized document pages is required, the transfer of this technology is not recommended.

The conclusions are drawn based on modeling a hypothetical task that involves reviewing 300 Russian documents to identify 50 relevant articles. Using return on investment measures based on data captured during testing, a favorable technology transfer assessment of the prototype Russian Language System is made, with caveats added for consideration by those organizations contemplating the insertion of the technology.

The decision to insert the technology into an organization depends on several key factors of its operational environment: the physical characteristics of the hard copy (quality, size, orientation) to be processed, the need for performing ad hoc retrospective information searches, and personnel costs.

Status: Ongoing research involving current technologies.

Operational Use: Measures will provide managers with the type of information they need to assess whether a technology purported to improve intelligence analysis will actually result in a reasonable return on the investment required to insert that technology.

Metadata Text Retrieval Conference (METTREC): Evaluating Document Conversion Technology for Retrieval

Michael D. Garris
National Institute of Standards and Technology
Gaithersburg, MD 20899
mgarris@nist.gov

Abstract

The Information Technology Laboratory at the National Institute of Standards and Technology (NIST) conducted the Census Optical Character Recognition (OCR) System Conferences to evaluate off-line handwriting recognition from forms, and it continues to run the Text Retrieval Conferences (TREC) now in its seventh year to evaluate information retrieval (IR) systems. Keeping with the tradition of these conferences, NIST under joint sponsorship with Department of Defense, is setting up a new series of evaluations called METTREC (Metadata Text Retrieval Conference) with a focus on evaluating document conversion and information retrieval (IR) technologies within the context of integrated tasks. Fundamentally, evaluations will be designed to investigate the impact machine recognition errors have on information retrieval. As a research focus, METTREC will investigate the interfacing of OCR/IR technologies. Specifically, how can metadata (additional non-text information) be utilized to enhance the performance and utility of these systems? A technical planning committee has been formed to develop a working definition of metadata, propose relevant OCR/IR tasks, and define methods of performance assessment. Data collection and processing has begun in preparation for an evaluation scheduled for next year. An open call for participation will be made as the project progresses.

NIST

Technology Evaluation Conferences

Purpose - promote competitiveness and commercialization of emerging technologies through technology evaluation.

Off-Line Handwriting Recognition:

- First Census OCR Systems Conference (1992)
  Isolated handprinted character recognition
- Second Census OCR Systems Conference (1994)
  Handprinted responses on forms

Information Retrieval:

- Text Retrieval Conferences (TREC)
  Text retrieval systems
  In its 7th year

METTREC

Metadata Text Retrieval Conference

Purpose - evaluate document conversion and information retrieval technologies within integrated applications.

- No application ... no test

Investigate:

- How should large digital collections of documents be constructed?
- What impact does machine recognition errors have on information retrieval?
- Are certain retrieval methods more tolerant of these errors than others?
- What impact does image quality have on these applications?
- What should the interface between the OCR and IR subsystems look like?
- What types of information should be passed?

  What types of metadata do users require?

  What types of metadata can be detected by OCR?

  What types of metadata can IR systems retrieve?

Metadata

- Non-text elements (physical or logical) of a document

  Fonts, page layout, equations, figures, tables, document type, language, title, author, dates, ...

- Metadata may be used to construct queries and/or it may be part of the information retrieved.

- Working definition of metadata will be developed by the planning committee.

Planning Committee

DFKI

- Andreas Dengel

DOD

- Steve Dennis, Glenn VanDoren

ETH Zurich

- Peter Schäuble

FIDUL

- Mike Lutjen

Lockheed Martin

- Suzanne Liebowitz Taylor

NIST

- Charles Wilson, Donna Harman

Ricoh

- Jonathan Hull

RPI

- George Nagy

Sabir Research

- Chris Buckley

UMD

- David Doermann

UNLV

- Kazem Taghva

Xerox

- Francine Chen

Testing Measures

OCR

- character and word error rates

Metadata

- detection and recognition error rates

IR

- retrieval rates on known items

New Measures

- relative, on-the-fly, statistical measures???

Testing Data

(Ground truth is a costly challenge)

Images

- Plethora of paper legacy documents that can be imaged
- Service bureau will scan ¢10 / page

Text (transcription and alignment)

- Electronic source files
  UW tools for Tex source documents
  NIST tool for GPO typesetting files (Federal Register)
- Keyed by service bureau (doubly)
  NIST tool to align text to word bounding boxes in the image
- Fusion of results from OCR API(s) with human verification

Metadata

- May be parsed or inferred from electronic source files
  Logical structures inferred from physical markup
- Marked by a manual human process
- Machine-assisted via customized detectors on page images

No application ... No test ... No labeling

Federal Register

- 1994 calendar year
- Published each business day of the year
- Typically one bound book per day
- Recycled paper, smudgy newsprint
- Books from 249 of the 250 days published
- The entire collection has been scanned at 400dpi
- Over 68000 imaged pages
- GPO typesetting files for each day published
- Parse physical markup and extract the printed text
- Infer logical elements from physical markup
- Align text to word bounding boxes in the page images
- Hierarchical / sectional organization to each publication
- Table of contents, indices, and cross references
- Multiple font styles and sizes
- Planning committee will develop testing scenarios and define relevant metadata

Federal Register Database

- 400dpi scanned page images
- SGML files containing text and relevant metadata
- Ideal synthesized images generated from a latex representation of the SGML files
- Published on CD-ROM

Contents

Federal Register
Vol. 59, No. 1
Monday, January 3, 1994

Rules and Regulations

Federal Register
Vol. 59, No. 1
Monday, January 3, 1994

DEPARTMENT OF TRANSPORTATION

Federal Aviation Administration

14 CFR Part 39

[Docket No. 92–ANE–08; Amendment 39–8781; AD 93–25–17]

Airworthiness Directives; General Electric CT7 Series Turboprop and Turboshaft Engines

AGENCY: Federal Aviation Administration, DOT.

ACTION: Final rule.

SUMMARY: This amendment adopts a new airworthiness directive (AD), applicable to General Electric (GE) CT7 series turboprop and turboshaft engines. This proposal would require the removal from service of certain gas generator turbine (GGT) rotor assembly parts that were plasma-sprayed during manufacture. This amendment is prompted by material testing that indicates that the plasma-spray process reduces the low cycle fatigue (LCF) capability of the material. The actions specified by this AD are intended to prevent fatigue cracks that can result in uncontained engine failure.

DATES: Effective February 2, 1994.

The incorporation by reference of certain publications listed in the regulations is approved by the Director of the Federal Register, as of February 2, 1994.

ADDRESSES: The service information referenced in this AD may be obtained from General Electric Aircraft Engines, 1000 Western Avenue, Lynn, Massachusetts 01910. This information may be examined at the FAA, New England Region, Office of the Assistant Chief Counsel, Attn: Rules Docket No. 92–ANE–08, 12 New England Executive Park, Burlington, Massachusetts 01803–5299; or at the Office of the Federal Register, 800 North Capitol Street, NW., suite 700, Washington, DC 20001.

FOR FURTHER INFORMATION CONTACT: Barbara G. Caufield, Aerospace Engineer, Engine Certification Branch, ANE–141, Engine Certification Office, FAA, New England Region, Engine and Propeller Directorate, Aircraft Certification Service, 12 New England Executive Park, Burlington, Massachusetts 01803–5299; telephone (617) 238–7146; fax (617) 238–7199.

SUPPLEMENTARY INFORMATION: A proposal to amend part 39 of the Federal Aviation Regulations to include an airworthiness directive (AD) that is applicable to General Electric CT7 series turboprop and turboshaft engines was published in the Federal Register on November 13, 1992 (57 FR 53862). That action proposed to require the removal from service of certain gas generator turbine (GGT) rotor assembly parts that were plasma-sprayed during manufacture, and replacement with serviceable parts. The actions would be required to be accomplished in accordance with GE CT7 Turboprop Service Bulletin (SB) A72–252, dated August 31, 1990; and GE CT7 Turboshaft SB A72–17 and SB A72–18, both dated September 10, 1990.

Interested persons have been afforded an opportunity to participate in the making of this amendment. Due consideration has been given to the comments received.

One commenter supports the rule as proposed.

One commenter observes that the economic analysis in the proposed AD is based on 23 engines being affected by the AD, when, in fact, there are only 11 engines containing a combination of 23 parts. The FAA concurs. The wording of the economic analysis for the final rule has been changed accordingly.

After careful review of the available data, including the comments noted above, the FAA has determined that air safety and the public interest require the adoption of the rule as proposed.

The FAA estimates that 11 engines (containing 23 affected parts) installed on aircraft of U.S. registry will be affected by this AD, that it will take approximately 7 work hours per engine to accomplish the required actions, and that the average labor rate is $55 per work hour. Required parts will cost approximately $8,255 per engine. Based on these figures, the total cost impact of the AD on U.S. operators is estimated to be $95,040. The manufacturer has advised the FAA that a pro-rata credit allowance will be granted for labor and parts required to accomplish these removals and replacements at a GE authorized service or overhaul facility. Based on this information, the FAA has determined that the total cost impact of the AD on U.S. operators would be approximately $5,940 ($540 per engine).

The regulations adopted herein will not have substantial direct effects on the States, on the relationship between the national government and the States, or on the distribution of power and responsibilities among the various levels of government. Therefore, in accordance with Executive Order 12612, it is determined that this final rule does not have sufficient federalism implications to warrant the preparation of a Federalism Assessment.

For the reasons discussed above, I certify that this action (1) is not a "significant regulatory action" under Executive Order 12866; (2) is not a "significant rule" under DOT Regulatory Policies and Procedures (44 FR 11034, February 26, 1979); and (3) will not have a significant economic impact, positive or negative, on a substantial number of small entities under the criteria of the Regulatory Flexibility Act. A final evaluation has been prepared for this action and it is contained in the Rules Docket. A copy of it may be obtained from the Rules Docket at the location provided under the caption **ADDRESSES**.

List of Subjects in 14 CFR Part 39

Air transportation, Aircraft, Aviation safety, Incorporation by reference, Safety.

Adoption of the Amendment

Accordingly, pursuant to the authority delegated to me by the Administrator, the Federal Aviation Administration amends 14 CFR part 39 of the Federal Aviation Regulations as follows:

PART 39—AIRWORTHINESS DIRECTIVES

1. The authority citation for part 39 continues to read as follows:

Authority: 49 U.S.C. App. 1354(a), 1421 and 1423; 49 U.S.C. 106(g); and 14 CFR 11.89.

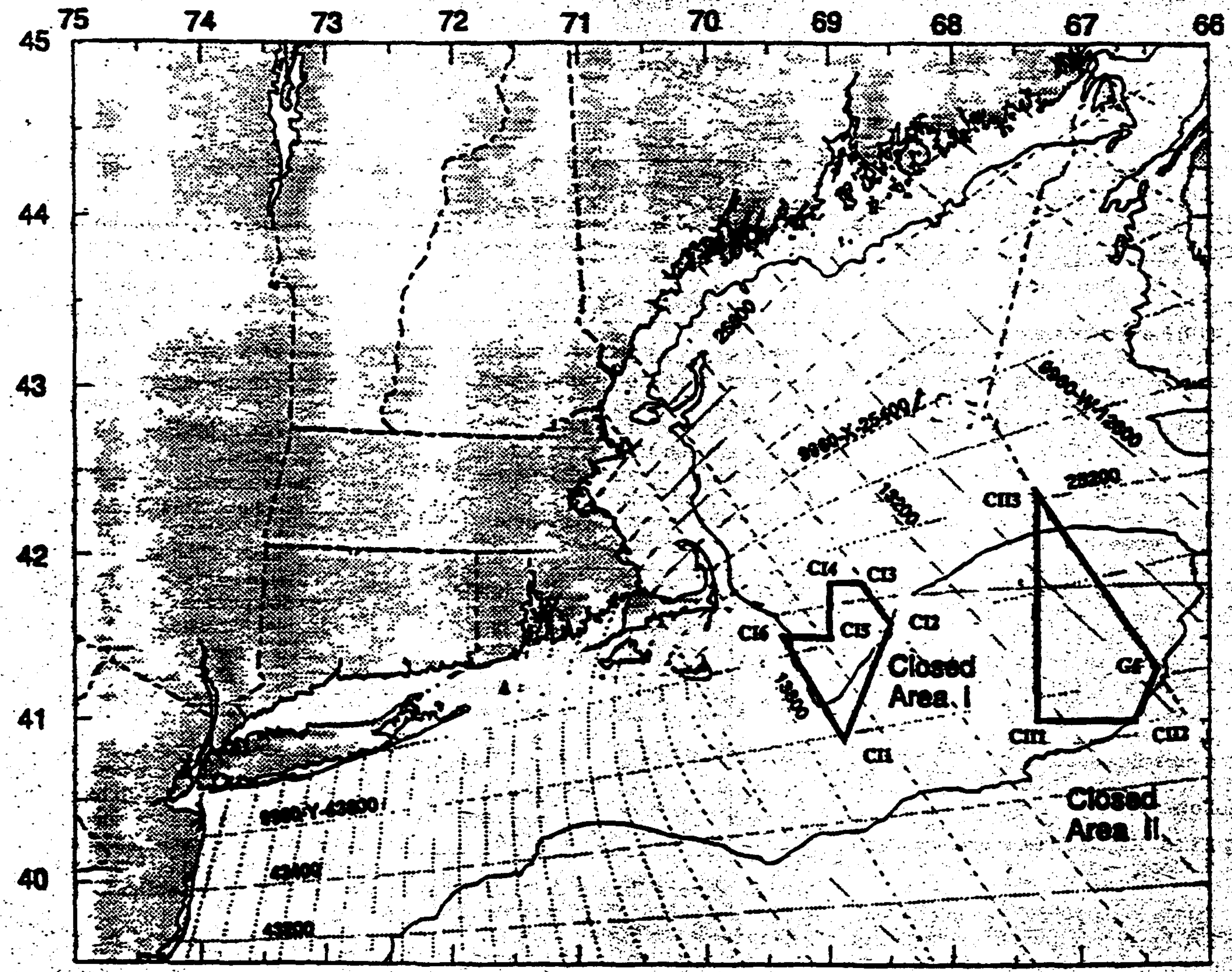


Figure 6: Closed Areas

[FR Doc. 93-32010 Filed 12-28-93; 3:22 pm]

BILLING CODE 3510-22-C

APPLICABLE BEGINNING JANUARY 1, 1999

Biological marker	Monitoring result categories		
	A	B	C
Cadmium in urine (CdU) (µg/g creatinine)	≤3	>3 and ≤7	>7
β_2-microglobulin (β_2–M) (µg/g creatinine)	≤300	>300 and ≤750	>750*
Cadmium in blood (CdB) (µg/liter whole blood)	≤5	>5 and ≤10	>10

* If an employee's β_2–M levels are above 750 µg/g creatinine, in order for mandatory medical removal to be required (See Appendix A Table B.), either the employee's CdU level must also be >3 µg/g creatinine or CdB level must also be >5 µg/liter whole blood.

Appendix A Table B—Actions Determined by Biological Monitoring

This table presents the actions required based on the monitoring result in Appendix A Table A. Each item is a separate requirement in citing non-compliance. For example, a medical examination within 90 days for an employee in category B is separate from the requirement to administer a periodic medical examination for category B employees on an annual basis.

Required actions	Monitoring result category		
	A[1]	B[1]	C[1]
(1) Biological monitoring:			
(a) Annual	X		
(b) Semiannual		X	
(c) Quarterly			X
(2) Medical examination:			
(a) Biennial	X		
(b) Annual		X	
(c) Semiannual			X
(d) Within 90 days		X	X
(3) Assess within two weeks:			
(a) Excess cadmium exposure		X	X
(b) Work practices		X	X
(c) Personal hygiene		X	X
(d) Respirator usage		X	X
(e) Smoking history		X	X
(f) Hygiene facilities		X	X
(g) Engineering controls		X	X
(h) Correct within 30 days		X	X
(i) Periodically assess exposures			X
(4) Discretionary medical removal		X	X
(5) Mandatory medical removal			X[2]

[1] For all employees covered by medical surveillance exclusively because of exposures prior to the effective date of this standard, if they are in Category A, the employer shall follow the requirements of paragraphs (l)(3)(i)(B) and (l)(4)(v)(A). If they are in Category B or C, the employer shall follow the requirements of paragraphs (l)(4)(v)(B)–(C).

[2] See footnote Appendix A Table A.

Appendix A—Attachment–2: List of Medications

A list of the more common medications that a physician, and the employee, may wish to review is likely to include some of the following: (1) Anticonvulsants: Paramethadione, phenytoin, trimethadone; (2) antihypertensive drugs: Captopril, methyldopa; (3) antimicrobials: Aminoglycosides, amphotericin B, cephalosporins, ethambutol; (4) antineoplastic agents: Cisplatin, methotrexate, mitomycin-C, nitrosoureas, radiation; (4) sulfonamide diuretics: Acetazolamide, chlorthalidone, furosemide, thiazides; (5) halogenated alkanes, hydrocarbons, and solvents that may occur in some settings: Carbon tetrachloride, ethylene glycol, toluene; iodinated radiographic contrast media; nonsteroidal anti-inflammatory drugs; and, (7) other miscellaneous compounds: Acetominophen, allopurinol, amphetamines, azathioprine, cimetidine, cyclosporine, lithium, methoxyflurane, methysergide, D-penicillamine, phenacetin, phenendione. A list of drugs associated with acute interstitial nephritis includes: (1) Antimicrobial drugs: Cephalosporins, chloramphenicol, colistin, erythromycin, ethambutol, isoniazid, para-aminosalicylic acid, penicillins, polymyxin B, rifampin, sulfonamides, tetracyclines, and vancomycin; (2) other miscellaneous drugs: Allopurinol, antipyrene, azathioprine, captopril, cimetidine, clofibrate, methyldopa, phenindione, phenylpropanolamine, phenytoin, probenecid, sulfinpyrazone, sulfonamid diuretics, triamterene; and, (3) metals: Bismuth, gold.

This list have been derived from commonly available medical textbooks (e.g., Ex. 14–18). The list has been included merely to facilitate the physician's, employer's, and employee's understanding. The list does not represent an official OSHA opinion or policy regarding the use of these medications for particular employees. The use of such medications should be under physician discretion.

Attachment 3—Biological Monitoring and Medical Examination Results

Employee ______

Testing Date ______

Cadmium in Urine _____ µg/g Cr—Normal Levels: ≤3 µg/g Cr.

Cadmium in Blood _____ µg/lwb—Normal Levels: ≤5 µg/lwb.

Beta-2-microglobulin in Urine _____ µg/g Cr—Normal Levels: ≤300 µg/g Cr.

Physical Examination Results: N/A _____ Satisfactory _____ Unsatisfactory _____ (see physician again).

Physician's Review of Pulmonary Function Test: N/A _____ Normal _____ Abnormal _____

Next biological monitoring or medical examination scheduled for ______

The biological monitoring program has been designed for three main purposes: 1) to identify employees at risk of adverse health effects from excess, chronic exposure to cadmium; 2) to prevent cadmium-induced disease(s); and 3) to detect and minimize existing cadmium-induced disease(s).

relating to the most recent calibration and QC sample analyses;

2. For these instruments, a tabulated record for each analyte of those determinations found to be within and outside of control limits over the past 2 years;

3. Results for the previous 2 years of the QC sample analyses conducted under the internal QA/QC program (this information should be: Provided for each analyte for which determinations are made and for each analytic instrument used for this purpose, sufficient to demonstrate that internal QA/QC programs are being executed properly, and consistent with data sent to responsible physicians.

4. Duplicate copies of monitoring results for each analyte sent to clients during the previous 5 years, as well as associated information; supporting material such as chain-of-custody forms also should be retained; and,

5. Proficiency test results and related materials received while participating in the CTQ interlaboratory program over the past 2 years; results also should be tabulated to provide a serial record of relative error (derived per Section 3.3.3 below).

3.3.3 Reporting Procedures

Participating laboratories should maintain these documents: QA/QC program plans; QA/QC status reports; CTQ proficiency program reports; and, analytical data reports. The information that should be included in these reports is summarized in Table 2; a copy of each report should be sent to the responsible physician.

TABLE 2.—REPORTING PROCEDURES FOR LABORATORIES PARTICIPATING IN THE CADMIUM MEDICAL MONITORING PROGRAM

Report	Frequency (time frame)	Contents
1 QA/QC Program Plan	Once (initially)	A detailed description of the QA/QC protocol to be established by the laboratory to maintain control of analyte determinations.
2 QA/QC Status Report	Every 2 months	Results of the QC samples incorporated into regular runs for each instrument (over the period since the last report).
3 Proficiency Report	Attached to every data report	Results from the last full year of proficiency samples submitted to the CTQ program and Results of the 100 most recent QC samples incorporated into regular runs for each instrument.
4 Analytical Data Report	For all reports of data results	Date the sample was received; Date the sample was analyzed; Appropriate chain-of-custody information; Types of analyses performed; Results of the requested analyses and Copy of the most current proficiency report.

As noted in Section 3.3.1, a QA/QC program plan should be developed that documents internal QA/QC procedures (defined under Section 3.3.1) to be implemented by the participating laboratory for each analyte; this plan should provide a list identifying each instrument used in making analyte determinations.

A QA/QC status report should be written bimonthly for each analyte. In this report, the results of the QC program during the reporting period should be reported for each analyte in the following manner: The number (N) of QC samples analyzed during the period; a table of the target levels defined for each sample and the corresponding measured values; the mean of F/T value (as defined below) for the set of QC samples run during the period; and, use of $\bar{X}\pm2\hat{\sigma}$ (as defined below) for the set of QC samples run during the period as a measure of precision.

As noted in Section 2, an F/T value for a QC sample is the ratio of the measured concentration of analyte to the established (i.e., reference) concentration of analyte for that QC sample. The equation below describes the derivation of the mean for F/T values, X (with N being analyzed the total number of samples analyzed):

$$X = \frac{\Sigma(F/T)}{N}$$

The standard deviation, $\hat{\sigma}$, for these measurements is derived using the following equation (note that $2\hat{\sigma}$ is twice this value):

$$\hat{\sigma} = \left[\frac{\Sigma(F/T-\bar{X})^2}{N-1}\right]^{\frac{1}{2}}$$

The nonmandatory QA/QC protocol (see Attachment 1) indicates that QC samples should be divided into several discrete pools, and a separate estimate of precision for each pools then should be derived. Several precision estimates should be provided for concentrations which differ in average value. These precision measures may be used to document improvements in performance with regard to the combined pool.

Participating laboratories should use the CTQ proficiency program for each analyte. Results of the this program will be sent by CTQ directly to physicians designated by the participating laboratories. Proficiency results from the CTQ program are used to establish the accuracy of results from each participating laboratory, and should be provided to responsible physicians for use in trend analysis. A proficiency report consisting of these proficiency results should accompany data reports as an attachment.

For each analyte, the proficiency report should include the results from the 6 previous proficiency rounds in the following format:

1. Number (N) of samples analyzed;
2. Mean of the target levels, $(1/N)\Sigma T_i$, with T_i being a consensus mean for the sample;
3. Mean of the measurements, $(1/N)\Sigma M_i$, with M_i being a sample measurement;
4. A measure of error defined by: $(1/N)\Sigma(T_i-M_i)^2$

Analytical data reports should be submitted to responsible physicians directly. For each sample, report the following information: The date the sample was received; the date the sample was analyzed; appropriate chain-of-custody information; the type(s) of analyses performed; and, the results of the analyses. This information should be reported on a form similar to the form provided an appropriate form. The most recent proficiency program report should accompany the analytical data reports (as an attachment).

Confidence intervals for the analytical results should be reported as $X\pm2\hat{\sigma}$, with X being the measured value and $2\hat{\sigma}$ the standard deviation calculated as described above.

For CDU or B2MU results, which are combined with CRTU measurements for proper reporting, the 95% confidence limits are derived from the limits for CDU or B2MU, (p), and the limits for CRTU, (q), as follows:

$$\frac{X}{Y} \pm \left(\frac{1}{Y^2}\right)(Y^2 \times p^2 + X^2 \times q^2)^{\frac{1}{2}}$$

For these calculations, X±p is the measurement and confidence limits for CDU or B2MU, and Y±q is the measurement and confidence limit for CRTU.

Participating laboratories should notify responsible physicians as soon as they receive information indicating a change in their accreditation status with the CTQ or the CAP. These physicians should not be expected to wait until formal notice of a status change has been received from the CTQ or the CAP.

3.4 Instructions to Physicians

Physicians responsible for the medical monitoring of cadmium-exposed workers must collect the biological samples from workers; they then should select laboratories

Target Concentration: 1.1 g/L (this amount is representative of creatinine concentrations found in urine).
Procedure: A 1.0 mL aliquot of urine is passed through a C18 SEP-PAK® (Waters Associates). Approximately 30 mL of HPLC (high performance liquid chromatography) grade water is then run through the SEP-PAK. The resulting solution is diluted to volume in a 100-mL volumetric flask and analyzed by HPLC using an ultraviolet (UV) detector.
Special Requirements: After collection, samples should be appropriately stabilized for cadmium (Cd) analysis by using 10% high purity (with low Cd background levels) nitric acid (exactly 1.0 mL of 10% nitric acid per 10 mL of urine) or stabilized for *Beta-2-Microglobulin* (B2M) by taking to pH 7 with dilute NaOH (exactly 1.0 mL of 0.11 N NaOH per 10 mL of urine). If not immediately analyzed, the samples should be frozen and shipped by overnight mail in an insulated container
Date: January 1992.

Chemists: David B. Armitage,
Duane Lee,
Organic Service Branch II, OSHA Technical Center, Salt Lake City, Utah.

1. General Discussion
1.1. Background
1.1.1. History of procedure
Creatinine has been analyzed by several methods in the past. The earliest methods were of the wet chemical type. As an example, creatinine reacts with sodium picrate in basic solution to form a red complex, which is then analyzed colorimetrically (Refs. 5.1. and 5.2.).
Since industrial hygiene laboratories will be analyzing for Cd and B2M in urine, they will be normalizing those concentrations to the concentration of creatinine in urine. A literature search revealed several HPLC methods (Refs. 5.3., 5.4., 5.5. and 5.6.) for creatinine in urine and because many industrial hygiene laboratories have HPLC equipment, it was desirable to develop an industrial hygiene HPLC method for creatinine in urine. The method of Hausen, Fuchs, and Wachter was chosen as the starting point for method development. SEP-PAKs were used for sample clarification and cleanup in this method to protect the analytical column. The urine aliquot which has been passed through the SEP-PAK is then analyzed by reverse-phase HPLC using ion-pair techniques.
This method is very similar to that of Ogata and Taguchi (Ref. 5.6.), except they used centrifugation for sample clean-up. It is also of note that they did a comparison of their HPLC results to those of the Jaffe method (a picric acid method commonly used in the health care industry) and found a linear relationship of close to 1:1. This indicates that either HPLC or colorimetric methods may be used to measure creatinine concentrations in urine.
1.1.2. Physical properties (Ref. 5.7.)
Molecular weight: 113.12
Molecular formula: C_4-H_7-N_3-O
Chemical name: 2-amino-1,5-dihydro-1-methyl-4H-imidazol-4-one
CAS#: 60-27-5
Melting point: 300 °C (decomposes)
Appearance: white powder
Solubility: soluble in water; slightly soluble in alcohol; practically insoluble in acetone, ether, and chloroform
Synonyms: 1-methylglycocyamidine, 1-methylhydantoin-2-imide
Structure: see Figure #1

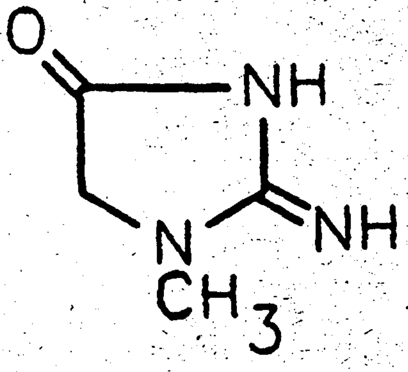


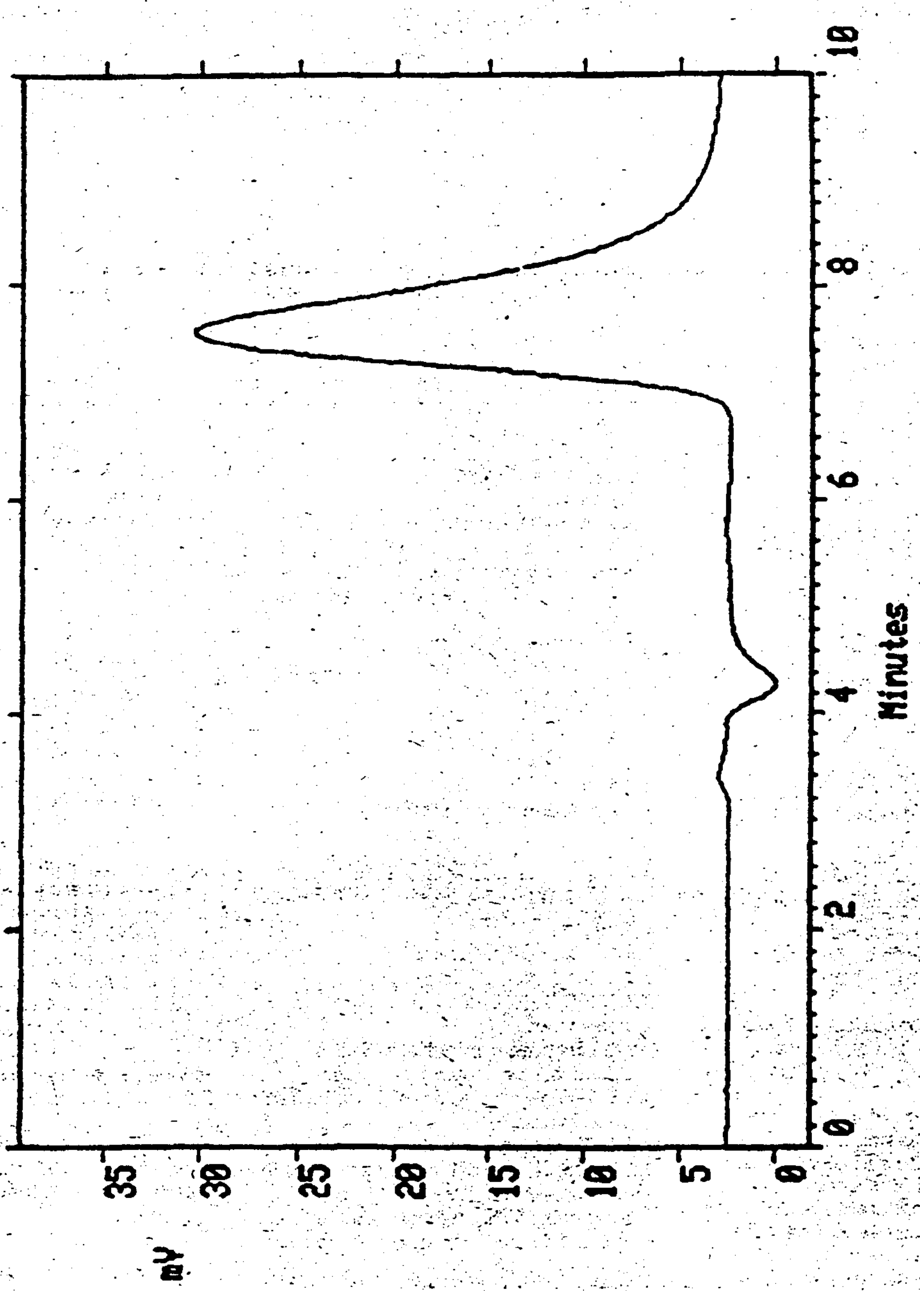
Figure #1

1.2. Advantages
1.2.1. This method offers a simple, straightforward, and specific alternative method to the Jaffe method.
1.2.2. HPLC instrumentation is commonly found in many industrial hygiene laboratories.
2. Sample Stabilization Procedure
2.1. Apparatus
Metal-free plastic container for urine sample.
2.2. Reagents
2.2.1. Stabilizing Solution—1) Nitric acid (10%, high purity with low Cd background levels) for stabilizing urine for Cd analysis or 2) NaOH, 0.11 N, for stabilizing urine for B2M analysis
2.2.2. HPLC grade water
2.3. Technique
2.3.1. Stabilizing solution is added to the urine sample (see section 2.2.1.). The stabilizing solution should be such that for each 10 mL of urine, add exactly 1.0 mL of stabilizer solution. (Never add water or urine to acid or base. Always add acid or base to water or urine.) Exactly 1.0 mL of 0.11 N NaOH added to 10 mL of urine should result in a pH of 7. Or add 1.0 mL of 10% nitric acid to 10 mL of urine.
2.3.2. After sample collection seal the plastic bottle securely and wrap it with an appropriate seal. Urine samples should be frozen and then shipped by overnight mail (if shipping is necessary) in an insulated container. (Do not fill plastic bottle too full. This will allow for expansion of contents during the freezing process.)
2.4. The Effect of Preparation and Stabilization Techniques on Creatinine Concentrations
Three urine samples were prepared by making one sample acidic, not treating a second sample, and adjusting a third sample to pH 7 The samples were analyzed in duplicate by two different procedures. For the first procedure a 1.0 mL aliquot of urine was put in a 100-mL volumetric flask, diluted to volume with HPLC grade water, and then analyzed directly on an HPLC. The other procedure used SEP-PAKs. The SEP-PAK was rinsed with approximately 5 mL of methanol followed by approximately 10 mL of HPLC grade water and both rinses were discarded. Then, 1.0 mL of the urine sample was put through the SEP-PAK, followed by 30 mL of HPLC grade water. The urine and water were transferred to a 100-mL volumetric flask, diluted to volume with HPLC grade water, and analyzed by HPLC. These three urine samples were analyzed on the day they were obtained and then frozen. The results show that whether the urine is acidic, untreated or adjusted to pH 7, the resulting answer for creatinine is essentially unchanged. The purpose of stabilizing the urine by making it acidic or neutral is for the analysis of Cd or B2M respectively.

COMPARISON OF PREPARATION AND STABILIZATION TECHNIQUES

Sample	w/o SEP-PAK (g/L creatinine)	with SEP-PAK (g/L creatinine)
Acid	1.10	1.10
Acid	1.11	1.10
Untreated	1.12	1.11
Untreated	1.11	1.12
pH7	1.08	1.02
pH7	1.11	1.08

2.5. Storage
After 4 days and 54 days of storage in a freezer, the samples were thawed, brought to room temperature and analyzed using the same procedures as in section 2.4. The results of several days of storage show that the resulting answer for creatinine is essentially unchanged.



Chromatogram of a creatinine standard
Figure #2

calendar year 1995 under Medicare's hospital insurance program (Part A) for the uninsured aged and for certain disabled individuals who have exhausted other entitlement. The monthly Medicare Part A premium for the 12 months beginning January 1, 1995 for these individuals is $261. The reduced premium for certain other individuals as described in this notice is $183. Section 1818(d) of the Social Security Act specifies the method to be used to determine these amounts.

EFFECTIVE DATE: This notice is effective on January 1, 1995.

FOR FURTHER INFORMATION CONTACT: John Wandishin, (410) 966-6389.

SUPPLEMENTARY INFORMATION:

I. Background

Section 1818 of the Social Security Act (the Act) provides for voluntary enrollment in the Medicare hospital insurance program (Medicare Part A), subject to payment of a monthly premium, of certain persons who are age 65 and older, uninsured for social security or railroad retirement benefits and do not otherwise meet the requirements for entitlement to Medicare Part A. (Persons insured under the Social Security or Railroad Retirement Acts need not pay premiums for hospital insurance.)

Section 1818(d) of the Act requires the Secretary to estimate, on an average per capita basis, the amount to be paid from the Federal Hospital Insurance Trust Fund for services performed and for related administrative costs incurred in the following year with respect to individuals age 65 and over who will be entitled to benefits under Medicare Part A. The Secretary must then, during September of each year, determine the monthly actuarial rate (the per capita amount estimated above divided by 12) and publish the dollar amount to be applicable for the monthly premium in the succeeding year. If the premium is not a multiple of $1, the premium is rounded to the nearest multiple of $1 (or, if it is a multiple of 50 cents but not of $1, it is rounded to the next highest $1). The 1994 premium under this method was $245 and was effective January 1994. (See 58 FR 58555; November 2, 1993.)

Section 1818(d)(2) of the Act requires the Secretary to determine and publish, during September of each calendar year, the amount of the monthly premium for the following calendar year for persons who voluntarily enroll in Medicare Part A.

Section 1818A of the Act provides for voluntary enrollment in Medicare Part A, subject to payment of a monthly premium, of certain disabled individuals who have exhausted other entitlement. These individuals are those not now entitled but who have been entitled under section 226(b) of the Act, continue to have the disabling impairment upon which their entitlement was based, and whose entitlement ended solely because they had earnings that exceeded the substantial gainful activity amount (as defined in section 223(d)(4) of the Act).

Section 1818A(d)(2) of the Act specifies that the premium determined under section 1818(d)(2) of the Act for the aged will also apply to certain disabled individuals as described above.

Section 13508 of the Omnibus Budget Reconciliation Act of 1993 (Pub. L. 103-66, enacted on August 10, 1993) amended section 1818(d) of the Act to provide for a reduction in the monthly premium amount for certain voluntary enrollees. The reduction applies for individuals who are not eligible for social security or railroad retirement benefits but who:

- Had at least 30 quarters of coverage under title II of the Act;
- Were married and had been married for the previous 1-year period to an individual who had at least 30 quarters of coverage;
- Had been married to an individual for at least 1 year at the time of the individual's death and the individual had at least 30 quarters of coverage; or
- Are divorced from an individual who at the time of divorce had at least 30 quarters of coverage and the marriage lasted at least 10 years.

For calendar year 1995, section 1818(d)(4)(A), as added by section 13508 of Public Law 103-66 specifies that the monthly premium that these individuals will pay for calendar year 1995 will be equal to the monthly premium for aged voluntary enrollees reduced by 30 percent.

II. Premium Amount for 1995

Under the authority of sections 1818(d)(2) and 1818A(d)(2) of the Act, the Secretary has determined that the monthly Medicare Part A hospital insurance premium for the uninsured aged and for certain disabled individuals who have exhausted other entitlement for the 12 months beginning January 1, 1995 is $261.

The monthly premium for those individuals entitled to a 30 percent reduction in the monthly premium for the 12-month period beginning January 1, 1995 is $183.

III. Statement of Actuarial Assumptions and Bases Employed in Determining the Monthly Premium Rate

As discussed in section I of this notice, the monthly Medicare Part A premium for 1995 is equal to the estimated monthly actuarial rate for 1995 rounded to the nearest multiple of $1. The monthly actuarial rate is defined to be one-twelfth of the average per capita amount that the Secretary estimates will be paid from the Federal Hospital Insurance Trust Fund for services performed and related administrative costs incurred in 1995 for individuals age 65 and over who will be entitled to benefits under the hospital insurance program. Thus, the number of individuals age 65 and over who will be entitled to hospital insurance benefits and the costs incurred on behalf of these beneficiaries must be projected to determine the premium rate.

The principal steps involved in projecting the future costs of the hospital insurance program are (a) establishing the present cost of services furnished to beneficiaries, by type of service, to serve as a projection base; (b) projecting increases in payment amounts for each of the various service types; and (c) projecting increases in administrative costs. Establishing historical Medicare Part A enrollment and projecting future enrollment, by type of beneficiary, is part of this process.

We have completed all of the above steps, basing our projections for 1995 on (a) current historical data and (b) projection assumptions under current law from the Midsession Review of the President's Fiscal Year 1995 Budget, incorporating the provisions of Public Law 103-66. It is estimated that in calendar year 1995, 32.548 million people age 65 and over will be entitled to Medicare Part A benefits (without premium payment), and that these individuals will, in 1995, incur $102.113 billion of benefits for services performed and related administrative costs. Thus, the estimated monthly average per capita amount is $261.44 and the monthly premium is $261. The monthly premium for those individuals eligible to pay this premium reduced by 30 percent is $183.

IV. Costs to Beneficiaries

The 1995 Medicare Part A premium is about 7 percent higher than the $245 monthly premium amount for the 12-month period beginning January 1, 1994.

We estimate that there will be, in calendar year 1995, approximately 277,000 enrollees who will voluntarily

Arabic and Persian OCR Training and Test Data Sets

Robert B. Davidson **Richard L. Hopley**
Science Applications International Corporation
McLean, Virginia 22102-3779
Robert.B.Davidson@cpmx.saic.com

Abstract

Large data sets consisting of bit-mapped images of real-world printed secular Arabic-alphabet text (in Arabic and Persian), accompanied by corresponding high-fidelity coded transcriptions (text ground truth), have been systematically chosen, prepared, and documented. Each data set is divided into a training set, which is made available to developers, and a carefully matched equal-sized set of closely analogous samples, which is reserved for testing of the developers' products. The samples were systematically chosen to represent current vocabulary, usage, typography, and publication practices in major newspapers and news magazines, and in recent books and journals dealing with politics, economics, and commercial and military matters. Lexicons and character-frequency tables have been compiled for each data set and for the Arabic and Persian collections as a whole. Availability of these data sets is intended to catalyze development by others of Arabic and Persian optical character recognition (OCR) and natural language understanding systems. The tools developed to support this effort are readily applicable in other non-Roman alphabets.

1 Introduction

Elements of the international business and foreign affairs communities that must communicate with and follow developments in parts of the world that communicate internally using non-Roman alphabets are severely handicapped by their practical inability to use modern computer-based analytical tools. In particular, they find it difficult to build and use efficiently databases of current and archival information, because much of that information is available only in printed form in the local languages. Potential local compilers and users of collections of coded electronic information in these parts of the world operate under similar handicaps. Effective optical character recognition systems for these alphabets could remove or lower the most significant barrier to use of electronic information systems in these alphabets: the high cost (in resources and time) of converting printed information into coded electronic form. Because many of the needs of these communities are analytic (rather than contractual or legal), even imperfect OCR systems could contribute significantly.

Arabic is the official language of all of the countries of North Africa and most of the countries of the Middle East (including the economically important nations of the Arabian peninsula, and strategic Iraq, Libya, and Syria). Arabic is the sixth most commonly used language in the world (after Mandarin Chinese, English, Hindi, Spanish, and Russian). In addition, the people of Iran, the Kurdish regions (in Iraq, Iran, Turkey, Syria, and the former Soviet Union), Afghanistan, and the Muslim portion of the Indian subcontinent (Pakistan and some parts of India) write their languages (Persian, Kurdish, Pashto, and Urdu,[1] respectively) in modified Arabic alphabets.

In the last few years, an increasing fraction of printed material in some of these countries has begun to be produced electronically. However, only a small fraction of this electronically produced material is being published and disseminated in electronic form, and almost no archival material is available in electronic form. And in many of these countries, electronic production of printed/typed material lags significantly, because the tools for electronic production in their alphabets (including capable desktop computers with native-language operating systems) are less well developed and less widely disseminated than the corresponding Roman-alphabet tools.

Efforts to date in Arabic OCR have identified and begun to address the problems involved in development of successful systems, but have not yet produced practical, widely usable systems. In part, this reflects inherent difficulties arising from particular characteristics of Arabic-alphabet languages (*e. g.*, cursive connection of printed letters, and routine intrusion of ascenders and descenders into the "space" of adjoining letters; distinct characters that differ from one another only by the placement of small auxiliary elements; extensive shape changes of letters and combinations of letters depending on their environment within a word, etc.). But it also probably reflects limitations of the development efforts to date. In particular, no reported study has used a training data set

[1]Urdu is essentially the same language (Hindustani) as Hindi, the official language of India. The major distinction is that Urdu is written in a modified Arabic alphabet and Hindi is written in the Devanagari alphabet.

of adequate size and diversity, or tested its recognition scheme against a large, real-world sample of printed Arabic. Recognition rates of known systems for material outside their training data sets are unacceptably low. The availability of large, carefully chosen, widely accepted standard training and test data sets can support more rigorous (and ultimately more successful) development activities. The sponsors of this effort[2] intend to catalyze such activities by providing the necessary training and test data (removing a significant obstacle from the development path).

To support rigorous OCR system development efforts, test and training data sets must themselves meet certain standards of rigor. At a minimum, they must be large enough (and carefully enough selected) to provide a population of words and characters that fairly represents the universe of material of most interest to a broad range of potential users. Real world samples (scanned with real scanners from real newspapers, magazines, and books) must predominate. The correspondence between characters in the images and characters in the text truth files must be very nearly perfect (perfect, if possible). All of these factors require diligent attention during development of the data sets, as do careful configuration management and quality assurance. The rest of this paper describes our efforts to achieve these high standards in constructing three data sets: two consisting of Arabic text and one consisting of Persian text. They approach our standards much more closely than any comparable non-Roman alphabet data set that has been described in the literature.

2 Selecting the Sample Sources

For each language, highly qualified native scholars of the language first prepared analyses of printing and publication practices. They identified the most important font families, and patterns of use of those font families within different categories of publication and different "schools" of typography (*e. g.*, the "Cairo" style of Arabic typography commonly followed in North Africa and the "Beirut" style of Arabic typography commonly followed in the Middle East). They documented standard usages within the character set of their language with respect to use of vowels, diacritical marks, numerals, special symbols, etc., which vary somewhat from country to country. Other experts (responsible for the African and Middle East collections of the Library of Congress) identified the most widely read and most influential regional and national newspapers, and also general interest and news magazines of wide circulation and some influence. Our language scholars made sure that the resulting selections broadly represented vocabulary, usage, typography, and publication practices. Among magazines/journals and books, they also selected titles to reflect bias in subject matter in favor of materials that dealt with current events, politics, and history; with military affairs; with science and technology; and with society and religion. (Literary works, low-level popular culture, and religious writings *per se* were purposely excluded.) Similar subject matter preference was applied later (see below) when choosing samples from the selected newspapers.

[2]The Defense Advanced Research Projects Agency (DARPA) and the Office of Naval Research (ONR).

3 Sample Collection, Preparation, and Configuration Control

Most of our samples came from the extensive Arabic-alphabet holdings of the George Washington University Library (books and magazines/journals of the first Arabic data set) and the Library of Congress (almost all newspaper samples; books and magazines/journals for the second Arabic data set and for the Persian data set). The rest (mostly technical and other books, military affairs journals, and a few newspapers) came from material borrowed from private collections.

For both development and evaluation reasons, it is desirable to separate the pure Arabic/Persian character recognition problem from issues associated with page analysis and recognition in any language (the commercial OCR marketplace is addressing the latter issues aggressively). To support this separation, each of our samples is a single column of body text (with minor headings, but without headlines in display fonts, graphics, or tables) or a one-column book page. In order to avoid copyright infringement, we limited our sampling to at most four such insubstantial excerpts from any given issue of a newspaper, magazine, or journal, or from any given book. Native speakers made the final sample selections, consistent with the guidance of our language/typography experts, our source category targets, and our subject matter preferences. (The target distribution among source categories was 50% newspapers, 30% magazines, and 20% books.) We took the excerpts in closely matched pairs, to help produce strictly comparable training and test data sets. (Within a training data set, it is also generally possible to select closely comparable pairs of samples, so that the developer can divide the data set into matching halves for training and in-house testing.) The average sample has about 300-350 words and about 1800-1900 characters. (See below for detailed statistics of the samples.)

Binary images were scanned at the collection sites, using a 600 dot per inch (dpi) flatbed scanner attached to a laptop computer. Flat settings were used on the brightness and contrast controls whenever the quality of the paper allowed (which was almost all the time for the sources selected). Scanned image originals were collected on floppy disks in Tag Image File Format (TIFF) (using lossless LZW compression); the original write-protected disks are retained as one of our primary archives. Serial numbers were assigned to the images as they were collected; a concurrent field log associated a full bibliographic citation (including English translation and Roman-character transliteration of the title) with the serial number. The field log entries were transcribed into our working catalog, which assigns each sample a unique identification number that incorporates much of the bibliographic information; the catalog allows each sample to be tracked through each of the subsequent steps of its processing. At the same time that the initial catalog entries were made, the images were transcribed from their floppy disk originals onto our working archival storage disks.

Each sample image was inspected. All extraneous graphic elements (*e. g.*, unintentionally scanned remnants of headlines) impinging on the image area were cropped out or erased, as were rules, boxes, or areas of anomalously degraded text (*e. g.*, smeared ink areas at the fold of a newspaper). One of us, who has had extensive professional experience in the printing industry, assigned each sample a print quality rating from 1 ("excellent") to 5 ("very poor"). Grade 5 samples, which were bad enough to give human readers difficulty, were excluded from further processing and from the final data sets. During the inspection, any unusual features of the samples, such as the presence of bullets, Roman or other non-standard characters, or bold or oblique type, were noted in their catalog entries.

4 Generation of Text Ground Truth

Printed versions of the images were assigned to two native-speaking typists. (Each typist received both same-size and twice-normal-size prints of each sample, the latter presumably being more readable.) Both contractual and practical measures were taken to ensure that the transcriptions would be truly independent. Each typist was carefully instructed to follow uniform conventions intended to produce identical typescripts from identical samples, with any typographical or other errors in the original images faithfully reproduced. Typists were instructed to substitute a place holder character for any non-Arabic-alphabet character that might occur in a sample.

After preliminary inspection (and, if necessary, correction) to bring the typescripts into closer conformance to our conventions, the two independent typescripts for each sample were reconciled by a native-speaking editor (assisted by custom text comparison software). Another copy of the printed images was used by the editor as the definitive version in these comparisons. At the conclusion of each editing session, the reconciliation software stored a corrected text truth file and recorded the errors made by each typist. (This same software is used in formal accuracy evaluations. See below.) A final visual inspection of the reconciled files brought the files into close visual conformance to conventions and detected (and corrected) some damage to the truth text files that occurred during the reconciliation process. Finally, a software-based inspection corrected violations of our conventions (Roman left-to-right space instead of Arabic right-to-left space, extra carriage returns, non-Arabic characters retained instead of replaced by the conventional place holder, etc.) that might be missed in a visual inspection.

5 Synthetic Files

In our first Arabic collection (there are now two) we included a small number of "synthetic" files. These were generated from Arabic text acquired in electronic form from a US Arabic-language newspaper. After applying the same subject-matter criteria used in selecting real-world samples, and editing the text to make it correspond to our typing conventions, single full pages of this text were printed on a 300 dpi Postscript printer. Two fonts corresponding to those used most often in printed material were used, in the relatively large type size our consultants told us was common in office correspondence. The printed sheets were then scanned and cataloged in the same way as other samples. These synthetic samples represent modern office correspondence within our corpus, provide a few very-high-quality "easy" recognition targets that developers have told us are useful at the early stages of OCR system development, and have guaranteed perfect correspondence between image and text truth files.

6 Statistical Analysis and Assignment of Samples

The finished text truth files were parsed by software that counts the number of words and characters in each sample and in the data set as a whole. Tables 1 and 2 summarize this statistical information for our (two) Arabic and (one) Persian data sets. In addition, for each alphabetical character, the number of times it occurred in the data set in each position (initial, medial, final, and free-standing) was also counted. When first encountered, each hitherto unseen word was recorded in a lexicon (as was the identity of the sample in which it first occurred[3]); when a word recurred, its count in the lexicon was incremented. (The software makes no attempt to group grammatical variants of the same word or root.[4]) The most recent version of the Arabic lexicon contains 69,905 words; the most recent version of the Persian lexicon contains 11,819 words.

The samples within each data set were divided into two closely matched subsets, a training subset and a test subset. Each subset has very nearly the same distribution of samples by source publication; *i. e.*, from a given book the same number of pages (one or two) are assigned to each subset; from a given newspaper or magazine/journal a very similar number of samples, yielding very nearly the same number of characters, are assigned to each subset. Care was also taken to ensure that each subset has a similar number of samples with each condition code. Within these constraints specific assignments were made in a way

[3]So that suspect entries can be examined readily in the original images.

[4]Thus, the lexicons (along with their frequency-of-occurrence counts) are useful as they are for spelling checking, and are suitable input for more sophisticated grammatical analysis and word/root counting.

Table 1. Statistics for the combined Arabic data sets, by sample type.

	Distribution of Samples	Num. of Samples	Total Chars.	Chars/ Sample	Total Words	Words/ Sample
Newspaper	48%	513	1,098,101	2,140	181,064	352
Magazine	29%	313	616,325	1,969	100,426	320
Book	23%	244	380,996	1,561	61,251	251
Natural Samples	100%	1,070	2,095,422	1,958	342,741	320
Synthetic Samples	–	68	75,943	1,116	12,700	186
Total Samples	–	1,138	2,171,365	1,908	355,441	312

Table 2. Statistics for the Persian data set, by sample type.

	Distribution of Samples	Num. of Samples	Total Chars.	Chars/ Sample	Total Words	Words/ Sample
Newspaper	51%	348	647,111	1,859	121,794	349
Magazine	31%	212	374,693	1,767	71,321	336
Book	18%	126	220,399	1,749	41,913	332
Total Samples	100%	686	1,242,203	1,810	235,028	342

Table 3. Division of samples between training and test subsets for the combined Arabic data sets.

	Training			Test		
	Samples	Chars	Words	Samples	Chars	Words
Newspaper	259	549,819	90,165	254	548,282	90,899
Magazine	158	306,234	49,869	155	310,091	50,557
Book	123	189,238	30,449	121	191,758	30,802
Natural Samples	540	1,045,291	170,483	530	1,050,131	172,258
Synthetic Samples	68	75,943	12,700	68	75,943	12,700
Total Samples	608	1,121,234	183,183	598	1,126,074	184,958

Table 4. Division of samples between training and test subsets in the Persian data sets.

	Training			Test		
	Samples	Chars	Words	Samples	Chars	Words
Newspaper	174	323,604	121,655	174	323,507	122,460
Magazine	106	187,386	71,005	106	187,307	71,110
Book	63	110,187	41,870	63	110,212	41,956
Total Samples	343	621,177	117,265	343	621,026	117,763

that balances the total number of characters in each set. Tables 3 and 4 summarize the results of this assignment process.

7 Available Training Data Sets

The training subsets of these data sets are available to developers, subject to approval by our sponsors. Generally, approval is forthcoming whenever the developer is willing to describe the nature of the project in which the data set is to be used, and to agree not to redistribute the data set.

Along with the image and text truth files, the developer receives a catalog of the included samples. Each is identified by source type (newspaper, magazine/journal, book, or synthetic), source code (a unique identification number that allows the developer to know when samples came from issues of the same newspaper or magazine/journal), condition/print quality code (1=excellent, 2=good, 3=fair, 4=poor but readable), character count, and word count. The developer also receives a lexicon (which contains all words found in either the training or the test subsets and a word frequency table) and a character frequency file (which records how often each character occurs, overall and in each position—initial, medial, or final—within words).

8 Formal Accuracy Testing

We have developed and (in one case so far) executed a protocol for formally testing OCR system accuracy. Using the testing data subsets and the reconciliation tools described above, we can help a developer (or a third party who is interested and has secured permission—through purchase of a commercial license or otherwise—to use a developer's system) to find out how well an Arabic or Persian recognition engine performs against a real-world data set, and what its specific strengths and weaknesses are.

First, we agree with the developer/user about what mutually convenient data transfer medium and protocols will be used for the test images and text files, and rehearse a data exchange. Then, at a mutually convenient time, we visit the developer's (or user's) facility to bring the test data set images and to witness the commencement of their processing. When the first ten images have been processed, the developer provides a copy of the resulting text files to support an initial on-site comparison of those files with the corresponding text truth files. Discussion of these initial results with the developer allows resolution of unforeseen issues at the early stages of (rather than after) the main effort of the evaluation.

When all of the test images have been processed, the developer delivers the rest of the resulting text files to us (in the same format and via the same medium as were used in the evaluation rehearsal). When practical (*i. e.*, when the processing can be completed within a day or two) we remain in the vicinity and return to the developer/user's site to pick up the files and observe the agreed clean-up procedure. In the latter, once we have received (and verified that we can read) all of the developer's text files, the developer/user removes all files associated with the testing from the developer/user's computing system. (The developer/user agrees not to re-create them from system backups.)

All of the developer's text files (or, if this is easier, all of our text truth files) are pre-processed to adjust for know differences in transcription conventions. Then the developer's text files are compared with our text truth files for the corresponding samples. Errors are counted by character, by type (insertion, omission, or replacement), and by position of the character within the word. As the results of these comparisons become available, we discuss them with the developer. Disagreements that the developer attributes to errors in our text truth files are investigated by humans, by reference to the original text images. For samples with reasonable numbers of errors (less than a few hundred[5]), our report details each of the developer's system's errors, summarizing them by type (character insertion, deletion, or replacement), by character, and by character position. We calculate neither error rates nor other "scores," but provide information that allows the reader to readily calculate error rates or other metrics if desired.

Acknowledgments

We are grateful to the Office of Naval Research (Dr. Thomas M. McKenna) and to the Defense Advanced Research Projects Agency (Drs. Sonny Maynard and Barbara L. Yoon) for support of this work under contracts N00014-92-C-0080 and N00014-94-C-0182. Kamal J. Boullata and Dr. Shadin C. Shafa provided invaluable language and typography guidance. Dr. Paritchehr Coates and Samir Abbedrabo were extraordinarily conscientious and skillful editors. We are also grateful to the Library of Congress and some of its outstanding African and Middle East collection staff members (Beverly Gray, Dr. George Atieh, and Ibrahim Pourhadi), and to the George Washington University library, for allowing and helping us to use their collections.

[5]When the OCR system's output for a particular sample is found to bear little or no relation to the text in the image, reconciliation is discontinued.

Additional Submissions

FAX Message Conversion & Searching

Henry S. Baird
Bell Laboratories
Lucent Technologies, Inc.
700 Mountain Ave, Room 2C-322
Murray Hill, NJ 07974
hsb@bell-labs.com

Abstract

FAX message streams pose serious challenges to the current generation of document image analysis systems. I review recent work at Bell Laboratories on automatic identification of the orientation (upside-down, landscape, etc) and language of FAXed pages of text: interestingly, there are advantages in considering these problems simultaneously. Also, I report on advances in classifier design that may permit FAXes to 'cook' — improve in recognition accuracy *indefinitely* — as they wait in multi-media mailboxes. I'll touch on some early engineering results in word–, phrase–, and address–spotting in highly degraded FAX images. If time permits, I may comment on some long–range implications for research in document image analysis of the rapid growth in the multi–media messaging industry. (Joint work with Dar–Shyang Lee, Craig Nohl, and Tin Kam Ho.)

Document Understanding Products and Research at Mitek Systems, Inc.

Gerald I. Farmer
Mitek Systems, Inc.
10070 Carroll Canyon Rd.
San Diego, CA 92131
farmer@miteksys.com
http://www.miteksys.com

Abstract

Mitek Systems, Inc. is an OCR/ICR software development company, concentrating in hand-printed character recognition and financial document processing. Mitek is the industry leader in the application of neural networks for hand-printed character recognition. In addition to several character recognition and related products we have ongoing research in areas of document image understanding.

Mitek is best known for its high-accuracy hand-printed character recognition engine, QuickStrokes. Two of our products, Premier Forms Processor and NiF use QuickStrokes for character recognition. PFP is a Windows based end-user system for forms processing. NiF automatically routes incoming faxes by recognizing the name of the recipient on the cover sheet. Mitek also offers QuickFrame, a neural network based page segmentation system that automatically separates document pages into regions by information type (i.e. machine-printed text, hand-printed text, photographs, drawings, etc.).

Mitek has several ongoing research projects. We continue to enhance the functionality of our QuickStrokes system. In addition we have an Arabic OCR research project that has led to the development of a commercial OCR system for recognizing Arabic machine printed text. We are nearing completion of a map image understanding project to develop a method for extracting text, symbols, roadways/rivers and gridlines from general map images. Currently in development is a language-independent OCR system called LITRE. To date, LITRE has been configured to recognize English, Thai, Lao, Burmese and Vietnamese languages. In another research project we are developing a system to automatically verify handwritten signatures.

Synthetic Training Data for Character Recognition Neural Networks

Michal P. Prussak, Laurence E. Bernstein, Ronald A. Linyard, J. Shane McRoberts, Sheena W. Rice, Brian C. Sparks, Bart Rothwell
Mitek Systems, Inc.
10070 Carroll Canyon Rd.
San Diego, CA 92131
http://www.miteksys.com
e-mail: mpp@miteksys.com

Abstract

Training of neural networks requires a large set of training data to obtain good generalization. Real data yields the best results, but often requires a significant effort to obtain. Synthetic training data can offer a good substitute for training. We show how to apply Baird's image defect model to generate synthetic training data for machine printed characters. We will show how synthetic data can be used to construct character classifiers that use some of the font metrics to aid in recognition. Finally we will show that in the LITRE text recognition system, synthetically trained recognition classifiers perform better than a classifier trained with real data.

1 Introduction

One of the methods used for character recognition are neural networks [1], [2], [3], [4], [5]. Neural networks can be trained to recognize a set of characters by repeated presentation of example characters during the training phase. Accuracy of neural network classification depends highly on the data presented during training. The set of training data for the neural network should be as large as possible, and it should be representative of the data to be classified by the network. Other types of classifiers can also benefit from a similar set of training data.

One way to collect training data for character recognition is to take the documents from which characters will be classified, extract images of characters and assign to each image the identity (*tag*) of the character it contains. This type of data is referred to as the *real data*. The real data is desirable because it is representative of the data to be classified, but it requires a significant manual effort to collect. Moreover, rarely occurring characters are difficult to collect in sufficiently large quantities. Lastly, as this is largely a manual process, it is prone to errors in tagging, sometimes resulting in character images with incorrect tags that can decrease the recognition accuracy. The value and cost of real character training sets is illustrated by the availability and prices of character databases from CEDAR [6], NIST [7], University of Washington [8], [9] and University of Seoul [10].

Another way to collect training data is to generate a page containing the desired characters, print it out and then scan it and extract the training characters from the scanned image. The page that is printed out can also be photocopied multiple times to simulate image defects, as found in real documents. The scanned image can then be automatically or semi-automatically processed to extract and tag character images, as the positions and identities of the characters are known. This type of data is referred to as the *artificial data*. Artificial data is easier to collect than real data, but may not be as representative as real data. Distortions found in artificial data will frequently not match the distortions found in real data.

Finally, training data can by synthetically generated. This can be done by rendering a perfect image of a character using a chosen font. This character can then be synthetically distorted using several possible distortion techniques. The distorted character can then be used for training a network. Data generated this way is referred to as the *synthetic data*. Synthetic data is generated completely automatically, so is easier to collect than both real and artificial data. Synthetic data might not be as representative as real data, as the distortions applied probably do not cover all possible real distortions.

Baird [11] has applied his defect model to create an English character classifier and reports a successful Tibetan classifier created with synthetic data. Bokser [12] reports an unsuccessful attempt to create a Cyrillic classifier with synthetic data - in this case, however, the data had not been synthetically distorted and ideal character images were used. Several attempts were made to measure the quality of this defect model [13], [14] leaning to the conclusion that it does not model the real defects adequately.

We will show that this defect model is adequate for the purpose of construction of character classifiers. We will show that in an experiment a synthetically constructed classifier outperformed a comparable classifier constructed with real data. In this experiment, the real data classifier was trained with about 250000 characters while the synthetic data classifier was trained with 940000 characters. This imbalance in the amount of training data favors the synthetic data classifier. However, the real data for training was expensive to obtain and all available data was used. The synthetic data was easy to generate and a manageable amount of that data was used for training. The ease of creation of synthetic data is its inherent advantage over real data. Therefore to test the applicability of synthetic data for training character classifiers we have generated as many character images as we could reasonably handle.

Section 2 describes an application of Baird's distortion model [11], [15] to synthetically degrade character images for training a neural network. Section 3 describes how synthetically generated data can be used to implement a more advanced character classifier, by adding several font metrics during training and multiple character glyphs. Section 4 shows that the networks trained with synthetic data result in higher overall accuracy in an OCR system, LITRE [2], currently under development by Mitek Systems.

2 Synthetic Character Image Distortions

We have chosen Baird's [11] image defect model as the basis for creation of synthetic data. [11] provides an excellent overview of this model and only a brief summary of is provided here. We also describe our modifications and parameters we have chosen for the distortions.

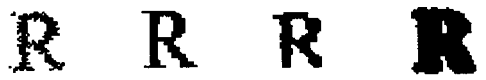


Figure 1. Examples of distorted character images.

We have applied the following distortions: *boldness, skew, width, height, resolution, jitter, blur.* For each of the distortions a range of distortion amount was selected, as well as the percentage frequency it should be applied. During the process of distorting the images, a set of distortions was first randomly chosen for each image, according to the desired frequency of each distortion. Next, for each distortion, a random distortion amount was selected linearly from the given range and the distortions were applied to the image. The distortions were applied cumulatively, in the order given above. The distorted images were generated from TrueType fonts at 300 PPI.

2.1 Boldness Distortion

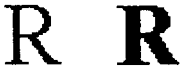


Figure 2. Boldness distortion examples.

The boldness distortion is not among the distortions described by Baird [11]. This distortion was applied by selecting the appropriate weight for the font in the Microsoft Windows SDK *CreateFontIndirect* function. This distortion was used to increase variety of characters used for training. Boldness amount was applied randomly from 100 to 900, where 400 represents the normal weight, 100 is the thinnest weight and 900 is the heaviest weight. Many fonts only had two boldness values. For these fonts only two types of characters - regular and bold - could be generated. Boldness was applied to all characters.

2.2 Skew Distortion

Figure 3. Skew distortion examples.

The skew distortion rotates the character image by a selected skew angle amount, as described in [11]. The skew distortion was applied by selecting an appropriate *escapement* parameter in the Microsoft Windows SDK *CreateFontIndirect* function. As a result, characters were rendered with the desired skew. A randomly selected skew between -2 and 2 degrees was applied to all images.

2.3 Width Distortion

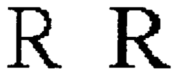


Figure 4. Width distortion examples.

The width distortion stretches the character horizontally by a selected amount, as described in [11]. This distortion was applied to all images with the distortion amount randomly selected between 0.9 and 1.1. A distortion amount of 1.0 represents a non-distorted image.

2.4 Height Distortion

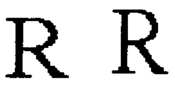


Figure 5. Height distortion examples.

The height distortion stretches the character vertically by a selected amount, as described in [11]. This distortion was applied to all images with the distortion amount randomly selected between 0.9 and 1.1. A distortion amount of 1.0 represents a non-distorted image.

2.5 Resolution Distortion

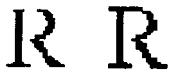


Figure 6. Resolution distortion examples.

The resolution distortion introduces spatial quantization effects to the character image, as described in [11]. This distortion was applied by scaling the images down by a selected amount and then scaling them up to the original size. 25% of the images had resolution distortions applied to them, with the randomly selected distortion amount between 0.5 and 1.0. The amount of 1.0 represents no distortion, while distortion of 0.5 represents scaling the image down by a factor of two and then scaling it back up by a factor of two. As described in [11], selecting small font sizes also introduced resolution distortion effect. During generation of training data, a range of point sizes from 6 to 14 points was selected for each font.

2.6 Jitter Distortion

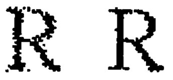


Figure 7. Jitter distortion examples.

The jitter distortion attempts to simulate the effect of slightly misaligned photo-receptors in the scanner, as described in [11]. In jitter distortion, a random amount of jitter distortion is chosen for each pixel in the selected range, separately for horizontal and vertical amount. The new position of the pixel is then calculated. As the new pixel coordinates are not integers, the pixel is mapped into a gray-scale bitmap by computing a proportion of four pixels occupied at that bitmap. Next, the value of each gray-scale pixel is incremented by this proportion. For example, if the new pixel coordinates are (3.5, 5.5), this pixel is split equally between four gray-scale pixels: (3, 5), (3, 6), (4, 5), (4, 6) - each of these pixels will have its value incremented by 0.25. After jitter has been applied to all pixels, pixels in the gray-scale bitmap with a value less than 0.5 are set to white and remaining pixels are set to black. Jitter was applied to 50% of the images with a randomly selected distortion amount in the range of 0 to 1. A jitter of 0 represents no distortion, while with jitter of 1, each pixel is allowed to move up to 1 pixel away.

2.7 Blur Distortion

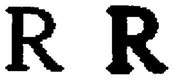


Figure 8. Blur distortion examples.

The blur distortion models the point-spread (or, impulse response) function of the combined printing and imaging process by applying a circularly symmetric Gaussian filter with a standard error of *blur*, as described in [11]. Blur distortion was applied to 25% of the images with the distortion amount randomly selected between 1.0 and 3.0. A blur amount of 1.0 represents no distortion.

3 Advanced Features of Synthetic Data

The defect model allows an easy creation of a training set whose distortions approach real defects. This way the utility of real training data can be approached. However, the process of synthesis permits the creation of a more advanced training set than could be created with real data. During the rendering of the characters their font metrics are known and can be stored with each character image. We have stored the baseline and upper and lower case height for each character image. Then, if the classifier can use these font metrics to aid in classification, higher recognition accuracy can be achieved. In addition images of glyphs other than single characters can be synthesized - either multiple characters or portions of characters. Synthesis of such glyphs allows the creation of a classifier to recognize them. When such glyphs are incorrectly generated by the segmentation procedure, they can still be correctly recognized by the classifier.

3.1 Font Metric Training Parameters

During the synthesis of character images, the font metrics of the characters are known. These font metrics can be included with each synthesized character image to be used for training. Such font metrics include the height of upper and lower case characters, and the baseline of the character. These parameters can also be computed by the segmentation routines during recognition and provided to the classifier to achieve higher recognition accuracy. For example, the baseline parameter can help distinguish commas from quote marks, which could be indistinguishable in many fonts. Similarly, the uppercase and lowercase height parameter can help distinguish characters which have a similar shape in upper and lower case, like 's' or 'o'. The neural network system used in LITRE [2] can use the baseline and upper and lower case height parameters computed by its segmentation module.

3.2 Non-Character Glyph Synthesis

The process of synthesizing distorted character images also allows the synthesis of glyphs other than single characters. Such glyphs can be used to recognize missegmented characters. For example, character pairs such as 'fi' or 'ff' are frequently typeset as a single ligature and are difficult to segment into separate characters. There may be other pairs of narrow characters which, if touching, can be difficult for the segmentation system to separate, for example 'el', 'll'. Being able to recognize such glyphs will decrease the amount of errors caused by incorrect segmentation, rather than incorrect recognition. This is important in current OCR systems, as segmentation errors can account for the majority of the errors [16].

The ability to construct multi-character glyphs is even more important in some non-Latin scripts, where characters can be arranged in a way difficult to segment by the segmentation system. For example, Burmese has a character that can contain one or more characters inside it [2]. Rather than attempt to segment such characters, it may be easier to synthesize all possible combinations of such characters and train a classifier to recognize them.

Multiple character glyphs introduce a new parameter in the synthesis system, called *squashing*. Squashing determines how closely should two characters be placed to each other. This parameter is used to vary the distance between the neighboring characters. During testing we have found that rendering of pairs of characters typically leaves a gap between them. However, character pair recognition is important when the characters are touching. Therefore we have selected 0.0 squashing in our experiments to ensure that in pairs of characters the characters are touching. A squashing of 1.0 would indicate the normal distance between characters.

The synthesis of partial character glyphs would also be useful in certain situations. For example, in a language with a variety of possible diacritical marks over a variety of characters, it may be desirable to segment the main character apart from its diacritical mark and recognize them separately. To achieve this, diacritical marks would have to be synthesized separately, as well as dots over i's and the bottoms of i's. Partial character glyphs could also be useful in a system that over-segments touching characters or to recognize Arabic text by over-segmentation.

4 Experimental Results

We have applied the synthetic distortion technique to train 14 neural classifiers: three English omnifont classifiers, OCR-A font ASCII classifier, OCR-B digit classifier, and two omnifont classifiers for each of these languages: Thai, Lao, Burmese and Vietnamese. All of the networks were trained successfully as measured on the synthetic testing set and by visually inspecting their performance. To compare the performance of a network trained with synthetic data to the performance that could be obtained with real data we have selected two of the classifiers for comparison with an existing classifier trained on real data.

The real data classifier was trained to recognize a set of 72 characters: upper and lower case letters, digits and the following special characters: $, %, &, *. This classifier was trained with approximately 250000 characters, with 3000 to 4000 examples of each character. This data set represents all of the real data available for training at the time. The synthetic data classifier was trained to recognize all 94 ASCII characters and it was trained with 940000 characters - 10000 examples of each character. For each character, samples were generated from 20 fonts at point sizes 6, 8, 10, 12 and 14. The names of the fonts used can be provided upon request.

We believe that it is reasonable to compare these classifiers, even though the synthetic data classifier was presented with about 3 times more data, because this reflects the ease of obtaining the synthetic data and thus is an inherent advantage of synthetic data. We have also constructed and tested a classifier to recognize all 94 ASCII characters as well as a set of 44 character pairs that we found frequently segmented as a single glyph. The synthetic data classifier was tested twice, once with the calculation of upper case character height turned on and once with this calculation turned off. With the uppercase character height calculation turned off, the LITRE system [2] calculated only the lowercase height and set the uppercase height equal to the lowercase height. This test was done to test the synthetic data classifier with the same information that was used by the real data classifier. In all, four tests were performed:

1. Real data classifier.
2. Synthetic data classifier, uppercase set to lowercase.
3. Synthetic data classifier, uppercase calculated.
4. Synthetic data classifier with character pairs.

The test was performed by running the OCR system LITRE [2], currently being developed by Mitek, on a set of 362 images from UW-II CD-ROM database [9]. The results are broken up into three parts: *Journal 1* lists results for 165 images from journal articles that have been photocopied once. *Journal 2* lists results for the same 165 images as *Journal 1*, except that they were photocopied twice. *Memo* lists results for 32 images of office memorandums. The recognition results were compared to the truth data supplied, using the OCR evaluation program from UW-I CD-ROM [8]. The percentage correct was calculated with the formula:

$$\% \text{ correct} = \frac{\#\text{ truth characters} - \#\text{ errors}}{\#\text{ truth characters}}$$

where errors include substitutions, insertions and deletions.

As the real data and synthetic data classifiers were trained to recognize different sets of characters, the results were adjusted for the characters missing from the real data classifier. The LITRE system [2] was running with iterative segmentation turned on, with bi-gram and error modeling postprocessing but without lexicon postprocessing. The system was providing the upper and lower case heights to the classifiers, with the exception of test 2. The real data classifier in test 1 was not using the lowercase height information. Table 1 shows the results of the experiments. It should be noted that the performance reported does not represent the number of characters correctly recognized by the classifiers. Most of the errors are due to incorrect segmentation. However, measuring the performance of classifiers on individual characters required a significant amount of effort, and we have compared the classifiers by testing them in a full OCR system. Moreover, the accuracy of the character pair classifier must be measured in a full OCR system, as this classifier is designed to compensate for OCR segmentation errors.

Table 1: OCR accuracy of LITRE with selected classifiers.

	Real	Synthetic		Synthetic
		Upper = lower	Upper calc.	Synthetic Pair
Journal 1	86.8	95.1	93.1	92.5
Journal 2	87.3	93.5	93.2	92.6
Memo	84.6	86.2	86.8	86.1
Total	**87.0**	**94.1**	**92.9**	**92.3**

The synthetic data classifiers outperformed the real data classifier, even if the comparison is limited to the character set of the real data classifier. Two observations in the results are noted.

First, the synthetic data classifier performed better in the test with uppercase calculation turned off. We have examined the results and noticed that by turning off the uppercase height calculation, better segmentation results were obtained, resulting in more well segmented characters and hence fewer recognition errors. The recognition itself was actually better with uppercase height available, but was overshadowed by the segmentation errors.

The second observation is that the character pair classifier did not outperform the ordinary synthetic data classifier, even though it was superior in smaller tests. Upon closer examination of the results we saw that the character pair classifier actually matched a higher number of characters correctly, but it also made more insertion errors. The insertion errors were made mostly in text areas severely mishandled by the segmentation module, and unrecognizable by either of the two classifiers. However, the character pair classifier tended to produce more output characters in these situations, thus losing its advantage in better quality of recognition. This suggests that appropriate tuning of LITRE's parameters can make the character pair classifier the more accurate of the two.

5 Summary

We have described an adaptation of Baird's image defect model [11] for the purpose of synthesis of distorted character images for training classifiers. We have produced a number of classifiers using this technique and tested two of them against an existing classifier, trained with real data. We have shown that the classifiers trained with synthetic data can outperform classifiers trained with real data, and hence that the image defect model is a sufficiently good approximation of image defects for the purposes of constructing character classifiers.

These results show that the defect model is applicable to generation of synthetic images of machine print characters to train a character classifier. As used currently, our system is limited to synthesis of character images from TrueType fonts. It cannot be easily used to generate images of printer fonts not available as TrueType fonts, such as dot-matrix or the chain or drum printer fonts (fonts such as OCRA, E7B are available as TrueType fonts). However, this system could be extended to degrade not just perfect character images rendered from a TrueType font, but any scanned character images, including hand-printed characters. This would allow the creation of a hybrid data set, containing real characters with additional synthetic distortions to further increase the size of the training set. Such an increase of the training set would result in a classifier more resistant to distortions and less susceptible to overtraining. However, in the case of hand-printed character sets, a small set will not be helped by the defect model, as the variety of shapes of letters cannot be increased synthetically. Still, the accuracy of classifiers trained on hybrid data could be increased this way.

The document image defect model could also be coupled more tightly with the training process, by synthesizing character images on the fly during training. This way classifiers could be trained effectively on infinite training sets.

References

[1] Y. Le Cun, *et al*, Handwritten digit recognition with a back-propagation network, in *Neural Information Processing Systems,* Dave Touretzky, ed. (Morgan Kaufman, Denver 1989), volume 2.

[2] M. Prussak, *et al*, LITRE: Language Independent Text Recognition Engine, in *Proc. of the 1995 Symposium on Document Image Understanding Technology,* Bowie, MD, Oct 1995, 213-221.

[3] S. Kahan, T. Pavlidis, H. S. Baird, On the recognition of printed characters of any font and size, in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **9** (1987) 1029-1058.

[4] S. Mori, C. Y. Suen, K. Yamamoto, Historical Review of OCR research and development, in *Proceedings of the IEEE*, **80** (1992) 1029-1058.

[5] R. Vogt, Neural network recognition of machine-printed characters, in *Proc. USPS Advanced Technology Conference,* November 1992, 715-726.

[6] Jonathan J. Hull, A database for handwritten text recognition research, in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **16** (1994) 550-554.

[7] Standard Reference Data, National Institute of Standards and Technology, NIST Test Data 1: Binary Images of Hand-printed Segmented Characters, Gaithersburg, MD, 1991.

[8] I. T. Phillips, S. Chen, R. M. Haralick, English document database standard, in *Proc. Second International Conference on Document Analysis and Recognition,* Tsukuba, Japan, October 1993, 315-318.

[9] R. M. Haralick, *et al*, UW-II English/Japanese Document Image Database, Seattle, Washington, 1995.

[10] D. H. Kim, *et al*, Handwritten Korean character image database PE92, in *Proc. Second International Conference on Document Analysis and Recognition,* Tsukuba, Japan, October 1993, 470-473.

[11] H. S. Baird, Document image defect models, in *Structured Document Image Analysis*, H. S. Baird, H. Bunke and K. Yamamoto, eds. (Springer-Verlag, New York, 1992), 546-556.

[12] M. Bokser, K. Choy, Synthetic training for OCR, in *Proc. DIMUND Workshop on Page Decomposition, Character Recognition, and Data Standards,* Harper's Ferry, WV, 1993.

[13] T. Kanungo, H. S. Baird, R. M. Haralick, Validation and estimation of document degradation models, in *Proc. Fourth Symposium on Document Analysis and Information Retrieval,* Las Vegas, NV, April 1995, 217-225.

[14] G. Nagy, Validation of OCR data sets, in *Proc. Third Annual Symposium on Document Analysis and Information Retrieval,* Las Vegas, NV, April 1994, 127-135.

[15] H. S. Baird, Document image defect models and their uses, in *Proc. Second International Conference on Document Analysis and Recognition,* Tsukuba, Japan, Oct 1993.

[16] R. G. Casey, Character segmentation in document OCR: progress and hope, in *Proc. Fourth Annual Symposium on Document Analysis and Information Retrieval,* Las Vegas, NV, April 1995, 13-40.

A Performance Evaluator for Engineering-drawing Recognition Systems

Ihsin T. Phillips and Jisheng Liang[†]

Department of Computer Science/Software Engineering,
Seattle University, Seattle, Washington 98122
[†]Department of Electrical Engineering,
University of Washington, Seattle, Washington 98195

Systems which convert existing paper-based engineering diagrams into electronic format are in demand and a few have been developed. However, the performance of these systems is either unknown, or only reported in a limited way by the system developers. An evaluation for these systems, or their subsystems, would contribute to the advancement of the field. Responding to this needs, a dashed-line detection competition for developers of dashed-line detection algorithms was proposed and took place during the first IAPR Workshop on Graphics Recognition at Penn State University, in 1995. A benchmark was developed and used in that competition. That benchmark includes a performance evaluator and a software tool that automatically generates dashed-lines test images and the corresponding groundtruth.

In this paper, we discuss a performance evaluator for engineering-drawing recognition systems on images that contain binary digital logic schematic diagrams. The evaluator accepts inputs of IGES files containing IGES primitives of straight lines, circles, partial arcs of circles, and IGES label block objects. Our evaluator takes two IGES files– the recognition algorithm's output and the corresponding *groundtruth*. First, the evaluator parses the two IGES files to extract IGES entities (primitives and labeled objects) and the parameter information of these entities.

To evaluate the primitives produced by the detection algorithm against the primitives in the groundtruth file, we compute the matching score and mark either *match* or *non-match* for each pair of primitives, one from the algorithm and the other one from the groundtruth. Since three types of primitives (lines, arcs, and circles) are allowed in our protocol, the evaluation protocol and the matching criteria are designed differently for each of the combinations. Our evaluator allows one-to-many matches on the primitives.

To evaluate the labeled block-objects produced by the detection algorithm against the labeled block-objects in the corresponding groundtruth file, we compute the union and the intersection of the areas of the two bounding boxes corresponding to the pair of labeled objects. If the ratio of the intersection and the union is less than 80 percent, we reject the match. If the primitives (sub-components) of this pair of objects are given in both files, we perform the *primitive matching protocol* on the two primitive lists. If less than 80 percent of the subcomponents match from one list to the other, we reject the match. Finally, if the names of the labels of this pair are match, the two objects are considered as a match, otherwise, we reject the match.

The results of our evaluator is a table of numbers which when weighted by application specific weights can be summed to produce an overall score relevant to the application. Although our evaluator accepts only limited IGES primitives (straight lines, circles, arcs, and label block objects), nevertheless, it is useful, since all straightforward digital logic diagrams use only a combination of these geometric elements. However, we are in the process of making an extension of our protocol to include other IGES primitives.

We understand that IGES is only one of many possible graphical file formats. We choose IGES file format because it is widely used and supported in the computer graphics industry. It has the distinct advantages of being standardized, non-proprietary, and having been designed from the beginning as an exchange format, not merely intended for storage of drawings within a single system. Users of our evaluator need only produce a standard IGES output file compatible with the format described in the paper and the corresponding groundtruth of the test images. The groundtruthing protocol for the preparation of groundtruth is also included in the paper. (Note that Haralick and Phillips's group at the University of Washington recently released a CDROM, UW-III, containing a collection of engineering drawing images. The groundtruth for those images are all in IGES file formats.)

SRI's Interest Page

Computer Engineering, Information, Telecommunications, and Automation Division
SRI International, Menlo Park, California 94025

Jeff DeCurtins: decurtin@erg.sri.com

Greg Myers: myers@erg.sri.com

Prasanna Mulgaonkar: prasanna@erg.sri.com

1 Introduction

The ability to search digitized document images for relevant information is a growing need in many business and government applications. Conventional approaches to this problem involve the use of segmentation processes followed by recognition processes to convert the pixel information into a symbolic representation that can be manipulated. However, such approaches have difficulty extracting information from poor-quality documents or documents with complex structure. SRI International has several related research efforts underway that are exploring the use of model-based or contextual information that can compensate for the degradation-induced information loss in complex documents. These methods use shape information from entire words to complement character recognition, lexicons organized in domain-specific ways to enhance recognition, and information combined from graphical and textual modalities within a single document. Several individual efforts currently underway are described below.

2 Key Word Spotting

With the advent of on-line access to very large collections of document images, electronic classification into areas of interest based on keyword content has become possible. An alternative to the use of OCR is the use of whole word shape recognition applied directly to the image. SRI has developed a system, called Scribble, based on this alternative. Testing has demonstrated that it is both faster and more robust in the face of poor image quality than OCR. Extensions being explored included recognition of Cyrillic, Arabic, and handwritten text and the detection of text in video streams.

3 Mail-Piece Address Reading

SRI has been developing systems to locate and read addresses on machine-printed mail pieces for the U.S. Postal Service. Machine-printed mail pieces carry on their covers not only the destination address we wish to find, but also the return address, a postmark, and often one or more advertisements. Our approach consists of two processing steps: address location and address recognition. Our approach to address location relies on the conventions of address structure to distinguish the address from other printing on the mail piece: these conventions include left justification as well as the length, height, and spacing of the text lines. Our approach to address recognition uses as a subsystem as an off-the-shelf OCR software package modified for the environment of scanned letter mail, which, as we have noted, includes a significant amount of poorly printed text and interfering background patterns. To correctly interpret the character recognition results, SRI took advantage of the logical relationships among the words' addresses, to use a method based on hypothesis generation and verification.

4 Information Extraction from Maps

Because complex color topographic maps contain several layers of information that overlap substantially (often within a single color plane), geometrically segmenting raster-scanned map image data into distinct graphical objects and text regions is difficult. SRI has been using verification-based recognition approaches that use contextual knowledge and constraints to formulate and then verify interpretation hypotheses. Because these approaches are based on the same principals as those used to recognize occluded objects in computer vision domains, they can operate successfully amid extraneous graphic information, even where the graphical object of interest is touching or overlapping other information. SRI has demonstrated the extraction of roads, symbols, and text from USGS maps.

Participant Interest Summary

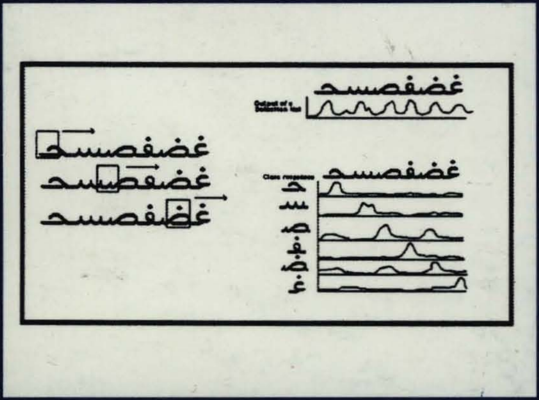
Michael Cannon, Judy Hochberg, Patrick Kelly, and James White
Computer Research & Applications Group
Los Alamos, National Laboratory
Los Alamos, NM 87545

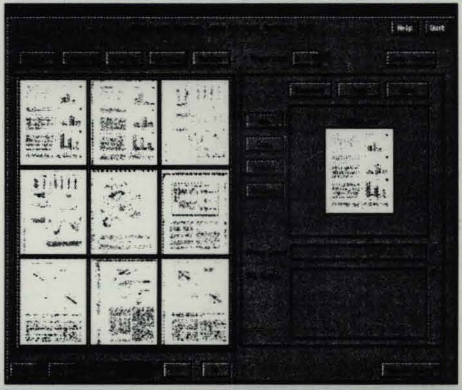
Our interest in document processing covers the entire document processing stream, starting with compression of document images, extending through optical character recognition to create textual documents, and ending with information extraction from textual documents. Our program draws on Laboratory expertise over several technical areas and organizations. It has applications in areas of national concern such as document declassification and archiving, as well as business applications.

Our past and current sponsors are government organizations such as the CIA,DOD., FBI, and DOE. In particular, the concerns we address are part of the DOE efforts in declassifying 337 million pages of documents. They also apply to archiving projects such as the Laboratory's Nuclear Weapons Archival Program (NWAP). We are active in the field of language identification from machine printed and handwritten documents.

Some of our specific research foci are the following:

- Our work on language identification has led to a successfully completed project to recognize the script of a machine printed document. Our method successfully differentiates between thirteen scripts, including Cyrillic, Roman, Arabic, Japanese, and Korean. We are currently examining methods to determine the script and/or language of handwritten documents.
- Other work on document image processing focuses on automatic assessment of document image quality. Quality assessment is a necessity when analyzing archives of documents that have been mimeographed or poorly typewritten, photocopied several times, and/or faxed. The quality assessment indicates how a document image might be best restored and also can be used to predict OCR accuracy.
- When document images are converted to textual documents via OCR, the resulting text contains many incorrect characters. This makes standard word-based information extraction techniques, such as keyword spotting, unreliable. Our work in *n*-gram analysis provides a rapid and accurate information extraction technique that is relatively robust in the face of such noisy text.
- From the DOE we have obtained some initial funding to categorize documents with neural networks. The categorization will lead to more efficient processing of documents by classification experts, either human or computer based.


Foreign Languages and Document Quality


Multimedia

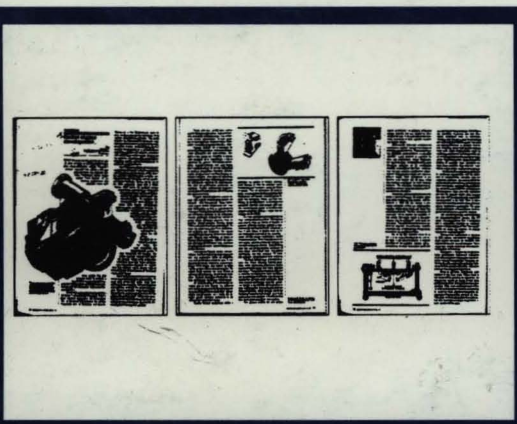
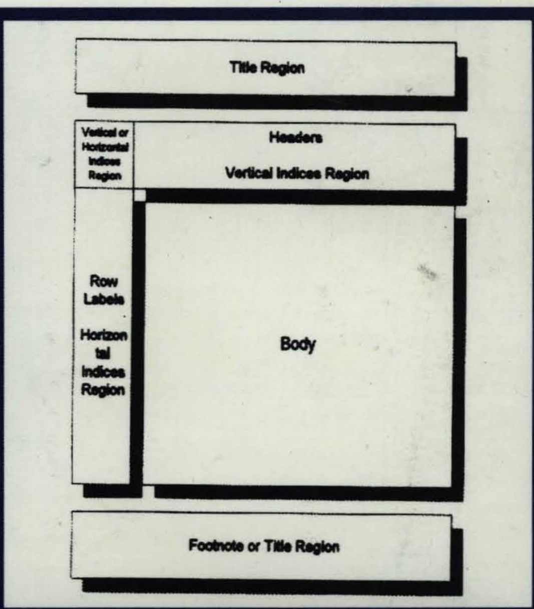


Image-Based Indexing



Applications