

The Skeleton Document Image Retrieval System

Carl Weir Suzanne Liebowitz Taylor
 Lockheed Martin C² 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

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 ScanWorX, 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 ScanWorX Modules

Since Skeleton relies upon ScanWorX to perform the initial segmentation of document images, it inherits that component's image input constraints. For ScanWorX 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 ScanWorX 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

¹ The ScanWorX OCR module can recognize text in images whose resolutions range from fax to 600x600 dpi, but for

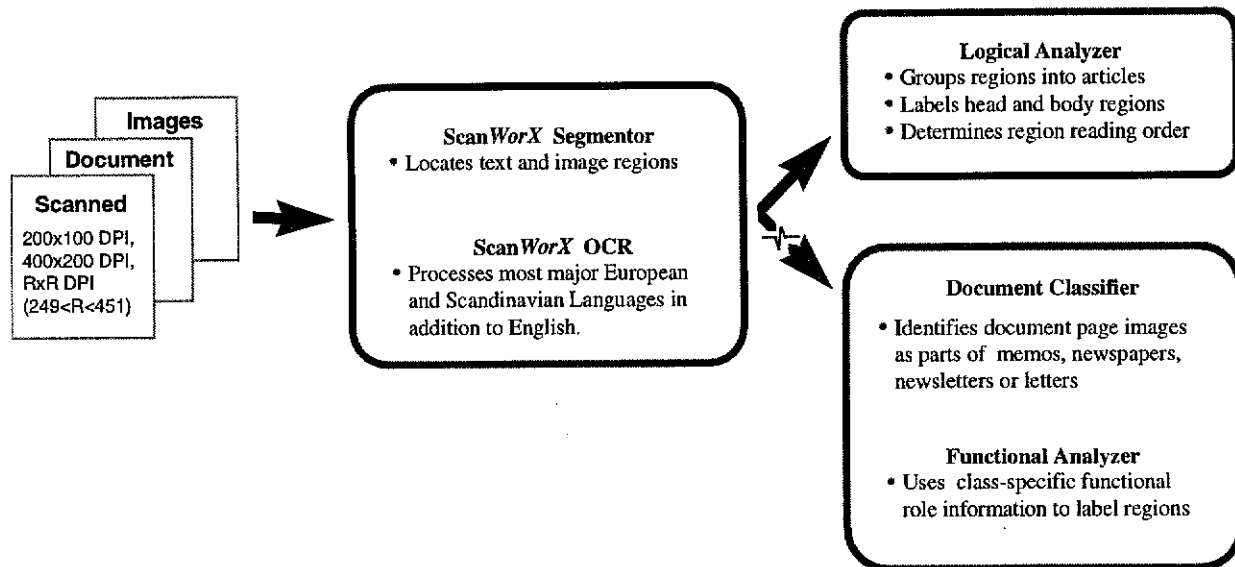


Figure 1: Skeleton's image analysis components are capable of providing a rich array of information about document structure and content. In the current implementation of Skeleton, information about logical groupings of regions into articles is used to build a document collection. Document class and 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.

supported by ScanWorX, Skeleton development has been based solely upon the analysis of images in TIFF format.

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

The ScanWorX 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

2.2 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

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,

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 subsequences (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

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 $N=18$ → exica
and division of 14 by 3 is rounded up
Second of "middle three" ngrams at position $((N-4)/2)+2 = 9$ → xic
Third of "middle three" ngrams at position $((N-4)/3)+2 \times 2 = 14$ → 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

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 identified 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

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

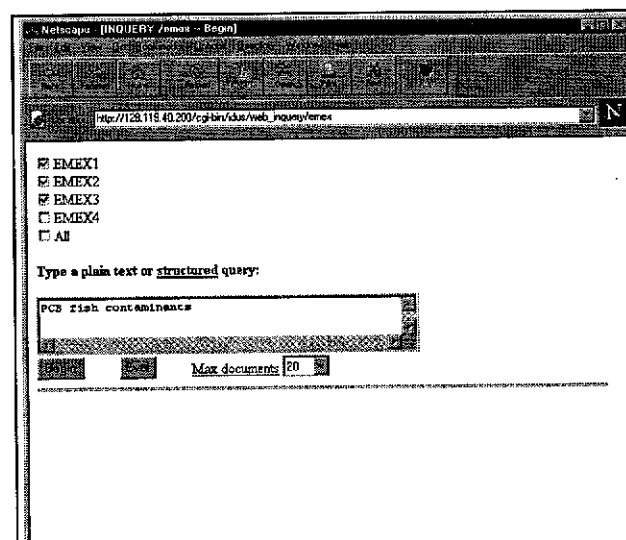
modification to maintain legal syntax in ngram binding operations when field based operators are present in the query.

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



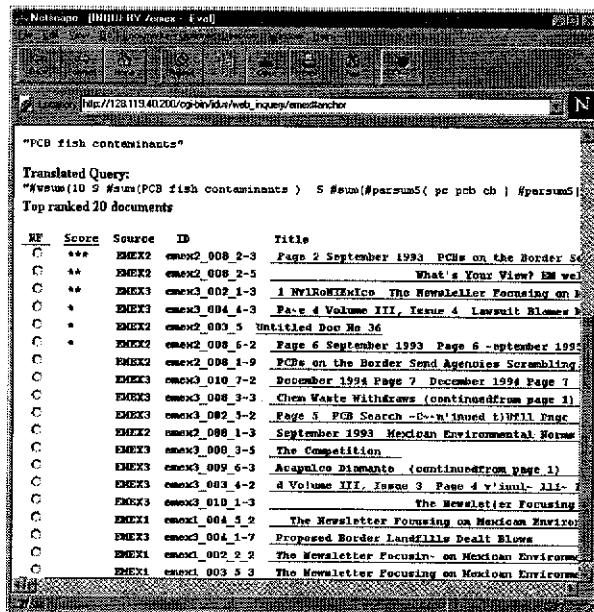


Figure 5: Skeleton Retrieval Results Form

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 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 facilitate 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

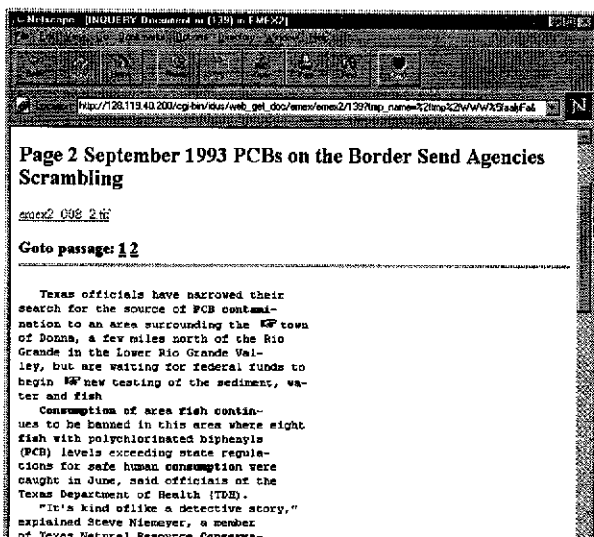


Figure 6: A display of a document page image containing an article

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 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

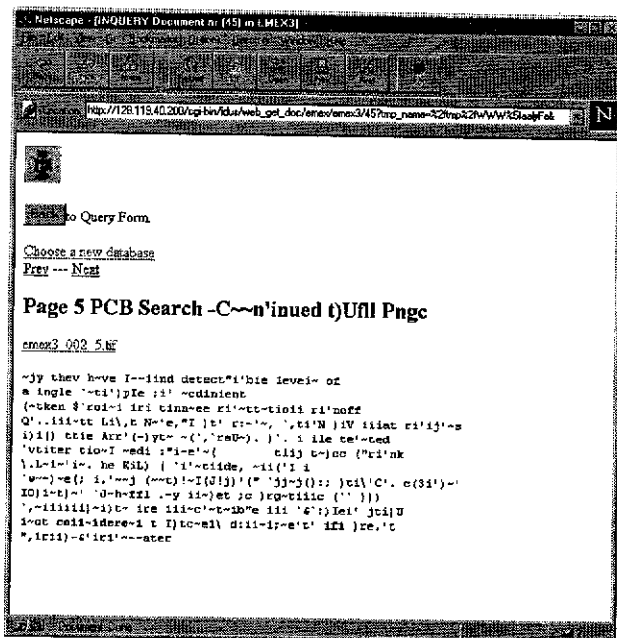


Figure 9: A display of an article's text containing significant OCR errors

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.

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.

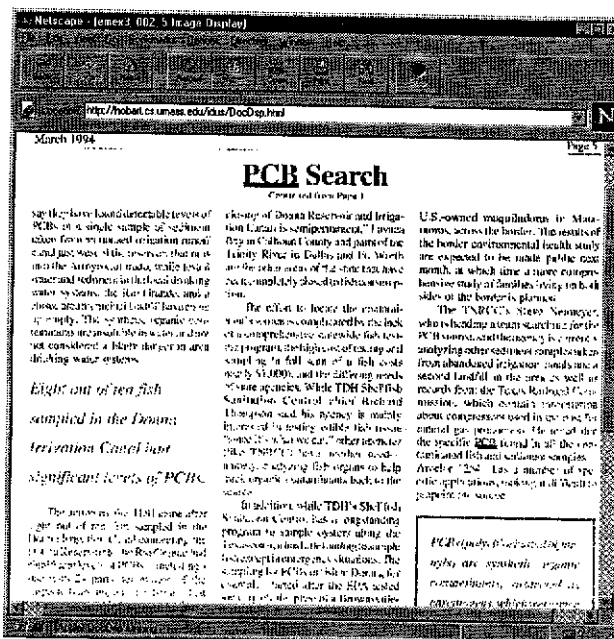


Figure 8: A display of a document page image containing an article with significant OCR errors

- [4] ScanWorX 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 *8th 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.