

Autotag: A Tool for Creating Structured Document Collections from Printed Materials

Kazem Taghva, Allen Condit, and Julie Borsack

Information Science Research Institute
University of Nevada, Las Vegas
Las Vegas, NV 89154-4021, USA

Abstract. We report on the design and implementation of a system which automates the process of capturing structured documents from the optically recognized form of printed materials. The system is intended to be used to convert printed collections into their corresponding tagged electronic versions with little or no manual intervention. This conversion process has some unique problems associated with it, these are discussed, along with our attempts to solve them. This system also establishes a mapping between the bitmap image and its corresponding ASCII representation that can be used to design flexible image-based interfaces for IR-related applications.

1 Introduction

Most reproduction of printed material produces a replica of the original. But if printed material is to be reproduced in a computer-usable form, complete duplication can be difficult if not impossible. The importance of transferring hardcopy documents to their computer-usable form should not have to be justified. But what is the most efficient and effective method of transferal? The projects that require this kind of conversion are usually on a grand scale; the cost of transformation, the efficiency of conversion and the eventual usefulness of the electronic media will determine the rationale for the project's continuance. Moreover, how do we retain the document features that were included in its original form when it was created by its author? It has been suggested by Southall[17], that "every perceptible feature of an actual document that is present, whether it is there by the author's intention or otherwise, plays some part in forming the reader's understanding of the meaning the document conveys." In many cases, when a document is converted to its electronic form, either by keyed-entry or Optical Character Recognition (OCR), a document's hierarchical structure, artwork, points of emphasis, and spatial relationships between its components are lost.

We have read or heard about a number of planned projects in which the intentions were to transfer a document collection to its electronic form and have it be immediately usable for its intended purpose or experimentation. In every case, the conversion process became a huge time-consuming and costly effort. One example is the Biological Knowledge Laboratory's design and implementation

of the Scientist's Assistant (SA)[7]. The SA is a system that will access the knowledge in a collection of scientific research documents. Their intentions were to scan, OCR, and tag the document set using the Standard Generalized Markup Language (SGML)[8]. After their initial attempts, they came to the conclusion that "the effort required to review and correct each article after it is scanned takes more time than to type in the complete article from scratch." Another example is the proposed Licensing Support System (LSS)[16]. The LSS is a planned system that will capture and track documents that pertain to the site licensing proceedings of the Nuclear Regulatory Commission. A prototype for this collection was constructed with both manual keyed-entry and OCR. After their prototype system was complete, they claimed "costs of conversion of hard copy documents to electronic form dominate the life cycle cost of the Licensing Support System." Further, the LSS contractors indicated that this system would incorporate only minimum document format information even though they felt other structural features would be useful:

Use of special type fonts such as boldface or italics is generally used for emphasis. Attempting to preserve such information in the LSS text database would be prohibitively expensive[16].

For technical documents to be efficiently converted, the process of scanning, recognizing, and tagging must be streamlined and automated. But to be effective, meta document information is necessary, revealing a document's elements, attributes, and structure. In this paper, we introduce a system called *Autotag* that automates the conversion process for the general class of technical documents. We use an OCR device that provides font and geometry information which we then use to determine a document's logical structure. A description of our project's ultimate purpose, the overall flow of the Autotag system, and its immediate applications follow.

2 Project Purpose

As the richness of document presentation evolves, so must our means of document representation. Documents are more than just a contiguous list of words; they are made up of sentences, paragraphs, sections, special fonts, diagrams, formulas and various other forms of artwork. Authors purposely add these document components to their text to help convey their ideas. These components should be embodied in the document's electronic representation. One way of capturing this information is through manual extraction; it can then be stored in document headers for later querying. This was the method chosen by the contractors of the Licensing Support System[16]. They knew certain kinds of information would be important to the users of the LSS, but were unable to represent it in the IR systems available at the time the prototype was developed. It was the set of queries that accompanied the prototype that initiated the course of this project's work. Figure 1 shows some example queries that were expected by the contractors of the LSS. These queries expect document representation to

be more than just a set of words. A document's logical structure and information about special objects contained in the text is required.

Query ID TEJB-T2-Q2: You would like a list of all reports authored by D. L. Bish of LANL discussing mineralogical characteristics of tuff at Yucca Mountain.

Query ID TEJC-T3-Q1: You would like to browse through the abstracts of all Sandia documents related to performance assessment modeling.

Query ID RLJC-T1-Q2: You would like to see a report by Sandia on repository sealing concepts of the Nevada Nuclear Waste Storage Investigations Project (NNWSIP).

Query ID PLJD-T1-Q1: You would like to see a map or diagram showing the location of the water table (known as the Calico Hills aquifer) in relation to the repository.

Fig. 1. Queries for the Licensing Support System requiring structured information.

The notion of attaining a richer document description is not new. Other trends in research suggest that a document's hierarchical structure and a means of recognizing its logical components is important. Standards such as the Open Document Architecture, (ODA)[2] and SGML show there is universal interest in a more profound representation. Macleod has investigated ways of taking advantage of document markup through a more flexible IR model and more powerful query languages[14, 13]. These topics pursue different aspects of the same issue.

The problem we address in this paper is the generation of these rich collections. As pointed out in the introduction, the time and effort required to produce a tagged collection, especially from documents only available in their printed form, is impractical without introducing some form of automation. The system we have designed, Autotag, helps solve the problems encountered by the LSS contractors and others who have faced the obstacle of capturing structural knowledge. Autotag automatically tags logical and structural components from OCR generated text, thereby reducing manual intervention and giving a more integral representation of the document. Autotag accepts a physical document representation as input. It analyzes and combines the information contained in this form and maps it to a logical representation. Figure 2 diagrams this process.

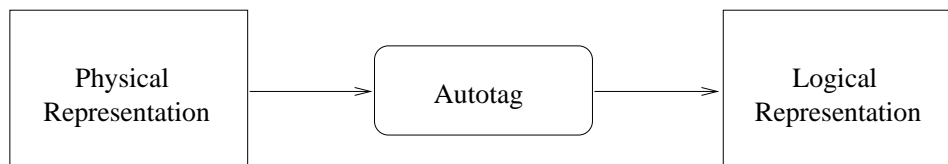


Fig. 2. Autotag maps a document's physical representation to its logical representation.

```

[a;"XDOC.9.0"]
[d;""]
[p;1;P;1;-212;0;0;2183;2810]
[s;1;1519;0;284;p;0] [e;1] [c;0] 0000Fj309 [y;1685;0;284;0;H]
[s;2;331;0;508;p;1] [e;2] [c;1] THE [h;414;18] 1980 [h;510;16] ERUPTIONS
[h;763;17] OF [h;833;14] MOUNT [h;1008;18] ST. [h;1084;20] HELENS,
[h;1272;19] WASHINGTON [y;1595;0;508;0;H]
[s;3;163;0;671;p;2] [e;3] [c;2] VOLCANO [h;449;19] MONITORING [h;849;20] BY
[h;937;20] CLOSED-CIRCUIT [h;1420;20] TELEVISION [y;1763;0;671;0;H]

```

Fig. 3. Example XDOC output.

3 Document Representations

One notable problem in our various projects[20–23] dealing with the combined use of OCR and IR is the conversion of an OCR page representation to a document form usable in an IR environment. An OCR device accepts an image of a hardcopy page. From this, the device produces a physical translation in ASCII of the page’s structure. Typically, this translation first separates text segments of the page from the non-text objects such as photographs, maps, and graphs (this is known as *zoning*), then produces the page’s text “words”. We have shown that with some automated processing[19], this document representation can be successfully used in an IR system. But there are some devices that provide other physical page attributes. For example, the ScanWorX OCR device from Xerox Imaging Systems produces a special output format, called XDOC[4], providing font and geometry information in addition to the recognized words. We believed this added information could be useful in an IR environment, but similar to our previous projects, for this information to be useful, the format had to be massaged and manipulated before it could be applied in IR.

3.1 Physical Representation

The initial representation we must deal with is the XDOC text markups. For each page of a document, this description provides recognized strings of text, whitespace dimensions, line delimiters, zone coordinates, zone types, font information, and other special text features such as superscript, subscript, and underlining. Example XDOC output appears in Figure 3. It is obvious how this kind of information can be useful in IR, but what is also apparent is that it requires logical interpretation before it can be exploited.

A direct translation from XDOC to a logical description was one of our options. But instead of this direct method, we decided to create an intermediate, device-independent representation and have Autotag process that. We felt that an intermediate representation would release Autotag from any direct relationship to the device used, and help avoid situations where we might become

dependent on some transient feature of the OCR device output. Second, many of the attributes are useful in their own right but would be irrecoverable if they were only used in combination with other information for marking logical structure. So a precursor step to Autotag is the parsing of the XDOC output to an independent physical document representation.

Parsing the XDOC language and extracting the information available turned out to be a nontrivial task. Some low-level computations are done during this processing phase to make character and spacing units of measure consistent, and make other obscure XDOC data more accessible to Autotag. Some of the other physical page data is simply parsed and output in a more convenient format. Figure 4 shows the physical structure of our intermediate representation after the XDOC format has been parsed. Currently, we store this physical representation in an SGML file format.

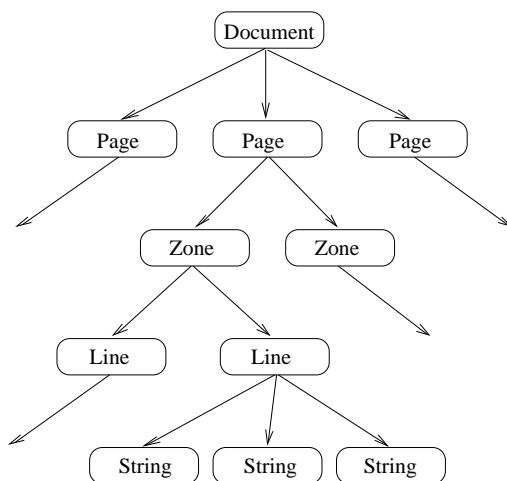


Fig. 4. Document's physical representation after parsing.

3.2 Logical Representation

Autotag deciphers, manipulates, and combines the information provided in the physical description and delivers information that is beneficial and useful for an IR collection. In general, this is an extremely difficult task. The logical components that are important in any particular collection may differ from application to application. So we decided to identify a particular class of documents (scientific journal articles) and focus on logical components that in the literature have shown importance in an IR context. To illustrate, text analysis in [10, 9] for identification of topics relies heavily on sentences and paragraphs. Salton

and others[1, 15] have shown that individual sentences can be used to identify relevant documents. Callan[3] has demonstrated that passage level evidence combined with document level evidence yields better recall than document level evidence alone. He also suggests that larger units such as sections may improve retrieval effectiveness. Experiments done by Wilkinson[25] imply that retrieval of whole documents can be done solely by measuring the similarity of queries to sections so long as the section types are known. Some studies have shown that words that have been weighted based on their structural context (e.g. within the title, abstract or reference section) may also improve retrieval[12].

In our own studies we have found that recognizing document structure is important when OCR text is used in an IR environment. For example, in [20] if floating objects had been marked prior to IR loading, many of the problems we encountered with “graphic text” would have been avoided. Further, with word-for-word correspondence between the ASCII text and a document’s image word location, the problems with showing the “noisy” ASCII to the user can be avoided by displaying document images with search terms highlighted on the image.

Keeping these logical structures and text attributes that have applications in IR in mind, Autotag will eventually mark the following components in an optically recognized document if they exist:

Front Matter: title, author list.

Body: abstract, section titles, sections, paragraphs, sentences, words.

Other Document Components: floating objects, references, running header and footer, footnotes.

Note that there are no hard-and-fast requirements for Autotag to successfully process and deliver a logically tagged document. There are two reasons for not imposing strict logical structure. One, we felt that Autotag would be generally more useful if it did not impose many requirements on the documents to be processed. Two, we are dealing with OCR text with possible errors in font and geometric information as well as character, case, and punctuation recognition. Sometimes the necessary information to mark a component is corrupt or absent entirely.

Autotag treats the logical document structure flexibly so that not every component is required to appear. As an example, documents which have no abstract or where the *tagging clues* for marking the abstract are corrupt, the integrity of Autotag’s logical output is not threatened; the abstract markup simply does not appear. Also, some documents may have other special components not represented by our logical structure. Currently, these components are simply parsed as best as can be expected while still marking words, sentences, paragraphs, etc. These structures do not necessarily cause Autotag to reject the document. Moreover, Autotag has been written in a modular way so that adding routines to locate new components will not require major design changes to the way Autotag functions. The lax rules incorporated into the Autotag system allow it to process almost *any* kind of document while continuing to mark words, sentences, and paragraphs at the very least.

Our current logical document structure is shown in Figure 5. Like the physical file format, we use SGML for the logical document file format.

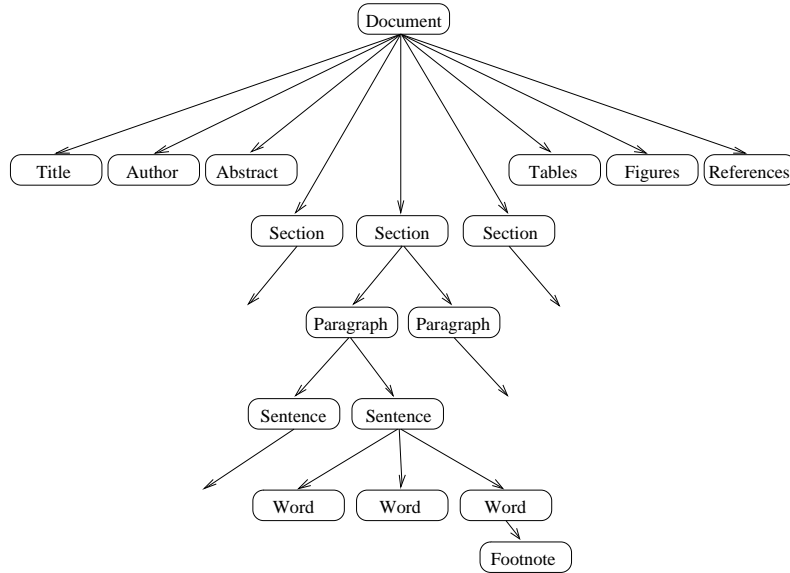


Fig. 5. Document's logical representation defined by Autotag.

4 Autotag Design

The input and output of Autotag have been described in Sections 3.1 and 3.2. This section gives an overview of the conversion process from its input (physical representation) to its output (logical representation). We also report on problems we have encountered with this translation.

Given the physical document representation shown in Figure 4, there are two distinct methods described in the literature for constructing its logical counterpart: the top-down approach and the bottom-up approach[11, 18, 24]. The top-down method would begin at the root of the physical tree. Based on information found at the document level, upper level logical objects such as the title, sections, and references, would be located first. Sections would then be divided into paragraphs and so on, until finally words were identified. The bottom up approach starts with the leaves of the tree (strings of text) and translates strings into words, words into sentences, and continues to identify more complex logical objects as it builds the complete document. The first approach seems to be the preferred approach in past work. But looking at Figure 6, the string node contains more directly applicable information than other physical tree nodes. Information about strings include font and exact string location on the image. The

conversion to words is a natural step, transferring the string information to the word nodes. The physical clues stored in the word simplify sentence construction and the propagation of this information to higher level objects continues up the tree. Since our physical document structure provides such detailed information at the string level, we decided to use the bottom-up method for logical document construction.

```
<string id="14" font-id="2" x1="1340" y1="770" x2="1640" y2="870">
Montana
</string>

<string id="541" font-id="2" x1="333" y1="3022" x2="670" y2="3064">
<superscript>0148</superscript>227/82/oo2B-0791
</string>
```

Fig. 6. Example contents of string nodes.

The translation from the physical document structure to the logical document structure, like previous work in this area[5, 6], depends heavily on heuristics. And many of the heuristics in Autotag depend on information provided by the device. Unfortunately, this information, like character translation, is not error-free. In fact, Autotag does not make use of some of the OCR information due to its low level of correctness. As an example, automatic zoning is unreliable, at least on pages with some level of complexity. This obstacle makes locating floating objects complicated and involved. Erroneous zoning also undermines the recognition of other objects. Automatic zoning not only indicates whether a particular zone is text or graphic material, it also specifies reading order. If the device enumerates the zones in the wrong order, any object identified by Autotag that crosses zone boundaries will be in error. Further, it may take Autotag some time to recover; in other words, several objects following the zone error may be identified incorrectly by Autotag.

So Autotag must be robust enough to encounter problems passed to it from the OCR device and recover from them quickly. A majority of our efforts in writing Autotag have been directed at exactly this objective. Autotag has successfully parsed hundreds of documents having a variety of formats. Although not all our heuristics are complete, Autotag's base is sound and has potential to be an extremely useful automatic tagging system. Currently, all the main body document components and front matter can be marked with Autotag. Of course not always correct, simply formatted documents are easily and correctly parsed. More difficult documents having graphic material not handled properly by the OCR device, formulas, lists, and other objects that Autotag has not been designed to handle yet are not as cleanly tagged. The next section shows both

Autotag's successes and deficiencies. We also present ways we already apply Autotag in other applications.

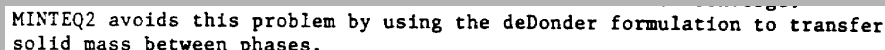
5 Evaluation and Immediate Applications of Autotag

Autotag's most notable success is the ability to process such a varied collection. We believe this accomplishment can be attributed to its layered design (physical then logical) and the grammar of the SGML document type definitions (DTD) defined for both the physical and logical representations. Like the base design, the structure finding heuristics are general but at the same time directed at the information given by the OCR device.

As described in Section 4, the logical document structure is built from the bottom up. At each level, we have encountered difficulties with tagging a document automatically. Some examples follow.

1. **Words:** In most cases, words are easily translated from strings. Even end-of-line hyphenation is simple most of the time. But there are cases that can cause problems. For example, OCR devices tend to have difficulty with characters that are not alpha-numeric—like hyphens. A number of times, the device will output a tilde (~) or some other character instead of a hyphen. In these cases the concatenation of the word parts will be missed. Since line breaks are no longer a part of the logical structure, a misspelling will result in the output. Another problem occurs when a hyphenated word crosses figures, footnotes, running headers or footers. If these objects are not correctly identified, then the word will not be correctly concatenated.

```
<sentence id="30">  
MINTEQ2 avoids this problem by using the deDonder formulation to  
transfer solid mass between phases THERMODYNAMIC DATABASE The  
thermodynamic database for MINTEQ continues to be expanded and do cu  
mented.  
</sentence>
```



```
MINTEQ2 avoids this problem by using the deDonder formulation to transfer  
solid mass between phases.
```

Fig. 7. The OCR device missed the period and in turn Autotag missed the sentence end, the paragraph, the section end and the beginning of a new section, paragraph, and sentence.

2. **Sentences:** Sentences are easily identified as long as they follow a few simple rules. Of course, punctuation and capitalization are important. Figure 7 shows one of our first attempts at sentence recognition. Note that when the period after **phases** is missed, Autotag continues to pick up words, missing the end of the paragraph and the beginning of a new section. These kinds of mistakes are fixed by using spacing and font information.

3. **Title and Author List:** The title and author of a document are usually obvious to human readers, even considering the many different ways these objects can be typeset on the page. OCR systems may do a reasonable job of recognizing characters and dividing up the page into several text zones, but unfortunately due to these variations in typesetting styles, OCR systems sometimes incorrectly order the text zones. This will most likely cause Autotag to incorrectly identify the title and authors. If zone ordering is the only issue, given a document with similar format with correct zone ordering, Autotag is usually able to correctly identify the title and author.

When we begin writing each heuristic for Autotag, we focus first on the clues given to us by the device and second, on common characteristics of a particular object. For example, when recognizing a document's title we focus on the location of the text and its font size and style. Second, we take into consideration, the number of words in a candidate string, whether the words are found in the dictionary, and capitalization. For something like the title, more emphasis is put on information given by the device about geometry and font. Sentences on the other hand rely more heavily on typical sentence characteristics—the same ones used in clean text. The reason is the information given by the device that can be used for sentence recognition is insignificant and not very reliable (horizontal spacing). Still, it is not until we can go back and carefully evaluate each recognized object with regard to what the OCR device has output that we can fine tune our heuristics. We are currently in the process of evaluating Autotag version 1.1. From these results we hope to gain insight into unanticipated problems and overcome them in the next version of Autotag.

Autotag is part of a larger system called *MANICURE* that we have designed at ISRI. *MANICURE* stands for *Markup ANd Imaged-based Correction Using Rapid Editing*. It is a system designed to correct misspellings caused by recognition either completely automatically, or semi-automatically at the user's choice. Autotag marks up the document, and this tagged document version is then passed to our automatic error correction module called the Post-Processing System (PPS)[19] which in turn transfers this partially corrected document to our user interface system, *Rummage*. *Rummage* applies the tagged information designated by Autotag to:

1. Search for strings on the image
2. Highlight document elements on the image
3. Spell check the document, highlighting misspellings on the image, and
4. Export the document to an HTML format

Although *MANICURE* is just a couple of months in the making, its components are well-developed and therefore is already an extremely useful and robust system.

Our eventual goal for Autotag is to be able to tag a general class of technical journal articles with an acceptable level of accuracy. Since error with OCR and Autotag is inevitable, we have designed *Rummage* which can be used as a back-up correction system to Autotag. In other words, if Autotag didn't mark a

document correctly, Rummage can be used to view the image and easily retag the document's text.

6 Conclusion

Autotag was designed and implemented with the hope of producing large structured document collections from currently available printed archives. The intention is to capture every conceivable object that may add to a document's representation. The set of document components selected for tagging have been chosen based on the document class and their applications in IR. Using the solid foundation of Autotag though, these could easily be changed or augmented for other document classes. Like other software that deals with free text, Autotag will not produce perfect results for every document. But we believe that by automating the conversion process, a number of rich electronic document collections can be constructed that previously would have been infeasible. Further, the mapping between the ASCII text and corresponding images has significant implications for many IR-related applications. We were able to apply this correspondence to our MANICURE system quickly. In any application where images and text are used, this relationship can be invaluable. We intent to exploit this capability in many of our future projects.

7 Acknowledgments

We would like to thank George Nagy and Donna Harman for their careful reading of the previous draft of this paper.

References

1. J. Allan, C. Buckley, and G. Salton. Automatic routing and ad-hoc retrieval using smart. In *Proc. TREC 2*. NIST, 1994.
2. W. Appelt and N. Tetteh-Lartey. The formal specification of the ISO open document architecture (ODA) standard. *The Computer Journal*, 36(3), 1993.
3. James P. Callan. Passage-level evidence in document retrieval. In *Proc. 17th Intl. ACM-SIGIR Conf. on Research and Development in Information Retrieval*, pages 302–310, Dublin, Ireland, July 1994.
4. Daniel S. Connelly and Beth Paddock. *XDOC Data Format Technical Specification*. Xerox Imaging Systems, Inc., Peabody, MA, March 1992.
5. Scott C. Deerwester, Keith Waclena, and Michelle LaMar. A textual object management system. In *Proc. 15th Intl. ACM/SIGIR Conf. on Research and Development in Information Retrieval*, pages 126–139, Denmark, June 1992. ACM Press.
6. Michael Fuller, Eric Mackie, Ron Sacks-Davis, and Ross Wilkinson. Structured answers for a large structured document collection. In *Proc. 16th Intl. ACM/SIGIR Conf. on Research and Development in Information Retrieval*, pages 204–213, Pittsburgh, PA, June 1993. ACM Press.

7. Robert P. Futrelle et al. Document analysis, understanding, and knowledge access. In *Proceedings of the International Conference on Document Analysis and Recognition (ICDAR)*, pages 101–111, St. Malo, France, 1991.
8. C. F. Goldfarb. *The SGML Handbook*. Oxford University Press, 1990.
9. Udo Hahn. Topic parsing: Accounting for text macro structures in full-text analysis. *Inf. Proc. and Management*, 26(1):135–170, 1990.
10. Marti A. Hearst and Christian Plaunt. Subtopic structuring for full-length document access. In *Proc. 16th Intl. ACM/SIGIR Conf. on Research and Development in Information Retrieval*, pages 59–68, Pittsburgh, PA, June 1993. ACM Press.
11. Rolf Ingold, Rene-Pierre Bonvin, and Giovanni Coray. Structure recognition of printed documents. In *Proc. Intl. Conf. on Electronic Publishing, Document Manipulation and Typography*, pages 59–70. Cambridge University Press, April 1988.
12. K. L. Kwok. The use of title and cited titles as document representation for automatic classification. *Inf. Proc. and Management*, 11:201–206, 1975.
13. I. A. Macleod. A query language for retrieving information from hierarchic text structures. *The Computer Journal*, 34(3):254–264, 1991.
14. Ian A. Macleod. Storage and retrieval of structured documents. *Information Processing and Management*, 26(2):197–208, 1990.
15. Gerard Salton, J. Allan, and Chris Buckley. Approaches to passage retrieval in full text information systems. In *Proc. 16th Intl. ACM/SIGIR Conf. on Research and Development in Information Retrieval*, pages 49–58, Pittsburgh, PA, June 1993. ACM Press.
16. Science Applications Intl. Corp. Capture station simulation: Lessons learned, Final Report, for the Licensing Support System, November 1990.
17. R. Southall. Visual structure and transmission of meaning. In *Proc. Intl. Conf. on Electronic Publishing, Document Manipulation and Typography*, pages 59–70. Cambridge University Press, April 1988.
18. A. Lawrence Spitz. Style directed document recognition. In *Proc. of IDCAR-91*, pages 611–619, St. Malo, France, 1991.
19. Kazem Taghva, Julie Borsack, Bryan Bullard, and Allen Condit. Post-editing through approximation and global correction. *International Journal of Pattern Recognition and Artificial Intelligence*, 9(6):911–923, 1995.
20. Kazem Taghva, Julie Borsack, and Allen Condit. Results of applying probabilistic IR to OCR text. In *Proc. 17th Intl. ACM/SIGIR Conf. on Research and Development in Information Retrieval*, pages 202–211, Dublin, Ireland, July 1994.
21. Kazem Taghva, Julie Borsack, and Allen Condit. Effects of OCR errors on ranking and feedback using the vector space model. *Inf. Proc. and Management*, 32(3):317–327, 1996.
22. Kazem Taghva, Julie Borsack, and Allen Condit. Evaluation of model-based retrieval effectiveness with OCR text. *ACM Transactions on Information Systems*, 14(1):64–93, January 1996.
23. Kazem Taghva, Julie Borsack, Allen Condit, and Srinivas Erva. The effects of noisy data on text retrieval. *J. American Soc. for Inf. Sci.*, 45(1):50–58, January 1994.
24. Lynn D. Wilcox and A. Lawrence Spitz. Automatic recognition and representation of documents. In *Proc. Intl. Conf. on Electronic Publishing, Document Manipulation, and Typography*, pages 47–57. Cambridge University Press, April 1988.
25. Ross Wilkinson. Effective retrieval of structured documents. In *Proc. 17th Intl. ACM-SIGIR Conf. on Research and Development in Information Retrieval*, pages 311–317, Dublin, Ireland, July 1994.