Integrated Text and Image Understanding for Document Understanding

Suzanne Liebowitz Taylor, Deborah A. Dahl, Mark Lipshutz, Carl Weir, Lewis M. Norton, Roslyn Nilson and Marcia Linebarger

Unisys Corporation
Paoli, PA 19301

ABSTRACT

Because of the complexity of documents and the variety of applications which must be supported, document understanding requires the integration of image understanding with text understanding. Our document understanding technology is implemented in a system called IDUS (Intelligent Document Understanding System), which creates the data for a text retrieval application and the automatic generation of hypertext links. This paper summarizes the areas of research during IDUS development where we have found the most benefit from the integration of image and text understanding.

1. INTRODUCTION

As more and more of our daily transactions involve computers, we would expect the volume of paper documents generated to decrease. However, exactly the opposite is happening. Considerable amounts of information are still generated only in paper form. This, compounded by volumes of legacy paper documents still cluttering offices, creates a need for efficient methods for converting hardcopy material into a computer-usable form. However, because of the sophistication of applications requiring electronic documents (e.g. routing and retrieval) and the complexity of the documents themselves, it is not sufficient to simply scan and perform OCR (optical character recognition) on documents; deeper understanding of the document is needed.

Comprehensive document understanding involves determining the form (layout), as well as the function and the meaning of the document. Document understanding is thus a technology area which benefits greatly from the *integration* of text understanding with image understanding. Text understanding is necessary to operate on the textual content of the document and image understanding is necessary to operate on the pixel content of the document. We have found great benefit from intertwining the two technologies instead of employing them in a pipeline fashion. We expect that more sophisticated document applications in the future will require even closer knitting.

Our document understanding technology is implemented in a system called IDUS (Intelligent Document Understanding System, described in Section 2), which creates the data for a text retrieval application and the automatic generation of hypertext links. This paper summarizes areas of research during IDUS development where we have found the most benefit from the integration of image processing with natural language/text processing: Document layout analysis (Section 3), OCR correction (Section 4), and Text analysis (Section 4). We also discuss two applications we have implemented (Sections 5 and 6) and future plans in Section 7.

2. GENERAL IDUS SYSTEM DESCRIPTION

IDUS employs four general technologies - image understanding, OCR, document layout analysis and text understanding - in a knowledge-based cooperative fashion[1, 2]. The current implementation is on a SPARCstation^{TM} II with the UNIX^{TM} operating system using the 'C' and Prolog programming languages. OCR is performed with the Xerox Imaging Systems ScanWorX^{TM} Application Programmer's Interface toolkit. All features are accessible via an X-Windows^{TM}/Motif^{TM} user interface.

After scanning the document page(s), IDUS works page-by-page, performing image-based segmentation to initially locate the primitive regions of text and nontext which are manipulated during the logical and functional analysis of the document. Each text unit's content is internally homogeneous in physical attributes such as font size and spacing style.

The ASCII text associated with each block is found through OCR and a set of features based both on text attributes (e.g., number of text lines, font size and type) and geometric attributes (e.g., location on page, size of block) is used to refine the segmentation and organize the blocks into proper logical groupings, i.e., "articles". The ASCII text for each "article" is assembled in a proper reading order. During this process the column structure of the document is determined, and noise and nontext blocks are eliminated. A text processing component performs a linguistic analysis to extract key ideas from each article and then represent them by a semantic component, the case frame. Each "article" text is saved as part of the document corpus and may be retrieved through a query interface.

3. DOCUMENT LAYOUT ANALYSIS

Document layout analysis determines the intra- and inter-page *physical*, *logical*, *functional* and *topical* organiz ation of a document. Many applications (e.g. automatic indexing or database population) require some level of understanding of the document's textual components. However, it is also critical to discover the document layout structure for two reasons: (1) document layout attributes such as position of text and font specifications are key clues into the relative importance of textual content on a page and (2) understanding the layout puts the text in the correct reading order for OCR conversion to ASCII/machine-readable text. Although image-based techniques provide valuable information for the physical decomposition of the document and feature input to the logical and functional areas, the incorporation of textual information ensures a more comprehensive layout analysis.

The current implementation of the document layout analysis module of IDUS consists of physical analysis, logical analysis, and a rudimentary functional component. We are currently developing a more sophisticated functional analysis.

3.1. Physical Analysis

Physical analysis of the document determines the primitive segments that make up each page and their attributes. Primitives include both text and non-text segments (photographs, graphics, rule lines, etc.). Attributes may include chromatic features (color or gray scale) and typesetting features (margins, point size and face style). Initially, physical analysis is achieved through page segmentation of the image of each page of the document and OCR (also an image-based process). Page segmentation starts with a binary image of a document page and partitions it into fundamental units of text or non-text. A desirable segmentation will form text units by merging adjacent portions of text with similar physical attributes, such as font size and spacing, within the framework of the physical layout of the document. The final output of the segmentation is a list of polygonal blocks, their locations on the page, and the locations of horizontal rule lines on the page[2].

Each text region in the document image is fed to the OCR to obtain its ASCII characters, as well as text attributes such as facestyle, pointsize, and indentation. These attributes are used as input to the logical page analysis along with the image-based feature output from the image segmentation.

3.2. Logical Analysis

Logical analysis groups the primitive regions into functional units both within and among pages and determines the reading order of the text units through manipulation of both image-based and text-based features. Logical analysis is first done on a page level and then on a document level.

The emphasis of the current logical analysis module has been at the page level[2] where we group appropriately (e.g., into articles in the case of a newspaper page) the text components which comprise a document page, sequence them in the correct reading order and establish the dominance pattern (e.g., find the lead article). A list of text region locations, rule line locations, associated ASCII text (as found from an OCR) for the text blocks, and a list of text attributes (such as face style and point size) are input to logical page analysis.

Transforming the blocks produced by the image segmentation into a logical structure is accomplished via the rule-based methods developed by Tsujimoto and Asada[3]. A geometric structure tree is created from the image segmentation, then a set of rules are applied to transform the geometric structure tree to a logical structure tree. In addition to the location and size of the blocks, these rules employ a gross functional classification as to whether a block is a *head* or a *body*. *Head* blocks are defined as those which serve in some way as labels or pointers, such as headlines and captions; *body* blocks are those which are referenced by head blocks and which have substantive content. Our head/body classification relies on features calculated for each block and for the page as a whole. The inputs to the feature calculations are the outputs of image segmentation and OCR. Construction of the geometric structure tree is predicated on the way that head blocks are positioned relative to their associated body blocks.

The Tsujimoto and Asada method for building the geometric structure tree assumes that the layout is well behaved in terms of uniformity of column configurations and placement of blocks along column boundaries. By finding ways of creating the geometric structure tree when the layout does not behave in the above manner, we have extended their approach.

Key to both the basic method and the enhancements is the determination of a page's underlying column structure. We compute column boundaries by applying area-based criteria to the set of all page-spanning column chains, where neighboring links represent contiguous blocks. The method is independent of the width of individual columns.

To handle complex layouts we create multiple geometric structure trees for the page which are merged during the creation of a logical structure tree. For a document with multiple sets of column bounds, we look for consistent column patterns in subregions of the document and construct a geometric structure tree within each subregion. The individual trees can be merged by treating each as a child of "root", to form a single tree for the page. Phenomena like inset blocks which interrupt straight column flows are "peeled off" into an overlay plane. Separate trees are constructed for this plane and the base plane and then merged so that an inset block is placed in the context of, but logically subordinate to, its surrounding article. The logical transformation then applies layout rules which associate blocks in the geometric structure tree(s) into "article" groupings in the proper reading order.

Text analysis enhances logical analysis particularly in the area of determining the reading order. Continuity of articles in a document may be very simple or very complex. An article may require many pages, but flow through the document in a linear fashion. In this case continuity may be determined by proximity. Popular periodicals, though, often have multiple articles per page with continuations separated by many pages. In fact, continuations may occur on pages which precede the initial text, as well as the more typical case of those which succeed the initial text.

To determine where an article continues, it may be sufficient to look for text such as "continued from page 13" or "continued on next page". However, there are several cases where a deeper linguistic interpretation will be necessary to find the article flow. Consider the scenario of multiple articles per page. If reading order is determined solely from geometric and typesetting attributes, then it is possible that some text components will be grouped incorrectly. If the reading order in this case must be verified, it may be necessary to incorporate more sophisticated linguistic processing, for example, comparing the last sentence fragment of one text region and verifying that is consistent with the sentence fragment of the candidate following region (either syntactically or semantically).

3.3. Functional Analysis

Functional analysis labels each unit (as determined through logical analysis) by role or function within the document. For example, a popular journal article will have a title, author reference, body text and photos. Functional analysis combines image-based and text-based features.

Functional labeling of regions, first at a high level to discriminate between head and body or text and non-text, and ultimately to determine the particular role a region plays, can be enhanced through the use of textual clues. Often these clues would be used in conjunction with format clues and clues derived from pixel-level features[4].

The simplest type of textual clue is string-based pattern matching or basic keyword search (such as looking for "Fig." or "Figure" to identify a non-text region). Since OCR systems tend to be overeager in recognizing regions as text, a reasonable clue combination would have the system look for the presence of a left-justified or centered figure tag below a region with a high number of character recognition errors. If both conditions are met, there is strong support for designating that region as non-text.

While much can be gained from simple string-based pattern matching, there are other information retrieval techniques of varying kinds and degrees of complexity which can be applied to advantage. We mention a few below, using informal descriptions.

String-based pattern matching with boolean operators and positional constraints. The "Figure" example above falls in this category. As another example, we might classify a block as an affiliation block (hence giving evidence that the document is a scholarly article) if it contains "College" or "University" within two lines of a (U.S.) state name or abbreviation.

Adherence to a simple grammar. The section headings for scholarly articles can be characterized by a grammar which specifies a sequence of digits and dots followed by an alphanumeric string. As another example, newspaper headlines and captions may conform to a certain style which has a syntactic basis, such as noun phrase followed by present participle for a picture caption : "Heroic firefighter rescuing cat from tree".

Frequency of string occurrences. In government *Requests for Proposals* (RFPs), phrases of the form *the offerer shall* and *the contractor shall* occur with much higher frequency than in other types of documents. This frequency information can be used to infer document type.

Frequency of syntactic structure occurrences. Newspaper articles typically contain a higher frequency of appositive constructions than other types of prose. Using syntactic analysis to detect the presence or absence of appositive constructions would license the inference that a newspaper writing style was present in a given text block, and therefore that a newspaper document type should be expected.

3.4. Topical Analysis

Topical analysis includes referential links among components, such as from the body of a text to a photograph it mentions. Examples of references include both explicit references (e.g., "see Figure 1.5") and implicit references (e.g., "following the definition described in the introduction"). The topical organization could also accommodate text summary templates or other interpretive notes. Linguistic processing should play a key role in determining the topical organization.

4. OPTICAL CHARACTER RECOGNITION CORRECTION

Language understanding serves two functions in document interpretation. First it is important if the text of the document must be interpreted and processed in some way, for example, to fill a database. In addition, natural language understanding technology can be used to improve the accuracy of optical character recognition (OCR) output, whether the ultimate goal is to interpret the text, to find textual clues, or simply to obtain accurate character recognition.

Our method for using linguistic context to improve OCR output[5] is based on extending work previously done in spoken language understanding[6, 7], where a natural language system is used to constrain the output of a speech recognizer. We have applied linguistic constraints to the OCR using a variation of the N-best interface with the natural language processing system, PUNDIT. PUNDIT is a large, domain-independent natural language processing system which is modular in design, and includes distinct syntactic[8], semantic[9] and application[10] components.

The data used for this experiment were obtained from processing facsimile output, since optical character recognition for these documents typically results in a high error rate. We selected a domain of input texts on which the natural language system was already known to perform well, air travel planning information[11]. The OCR output is sent to the *Alternative Generator*, which consults with a lexicon of allowable words and a spelling corrector, developed as part of this project. The spelling corrector by itself successfully corrects many OCR errors. The output of the spelling corrector is also used to generate an ordered list of N-best alternative sentences for each input. These sentences are sent to PUNDIT, which selects the first meaningful alternative sentence.

The generation of an N-best list of sentence candidates is a simple matter of taking a "cross-product" of all word alternatives across the words of a given sentence. The score of each candidate sentence is the product of the scores of the words in it; words without multiple alternatives are assigned a score of 1.0. N-best sentence candidates are presented to the natural language system in ascending order of score. The sentence with the lowest score that is accepted by the natural language system is taken to be the output of the integrated OCR-natural language recognition system.

The system was tested with 120 sentences of air travel planning data on which the natural language system had previously been trained. A known text was formatted in LaTeX in both roman and italic fonts, and printed on an Imagen^{TM} 8/300 laser printer. It was then sent through a Fujitsu dex9^{TM} facsimile machine and received by a Lanier 2000^{TM} facsimile machine. The faxed output was used as input to a Xerox ScanWorX^{TM} scanner and OCR system. Figure 1 shows a typical sentence after scanning, after spelling correction (with N-best sentence candidates), and after natural language correction.

Performance was measured using software originally designed to evaluate speech recognizers, distributed by the National Institute of Standards and Technology [14]. The word error rate for output directly from the OCR was 14.9% for the roman font and 11.3% for the italic font. After the output was sent through the alternative generator, it was scored on the basis of the first candidate of the N-best set of alternatives. The error rate was reduced to 6% for the roman font and to 3.2% for the italic font. Finally, the output from the natural language system was scored, resulting in a final error rate of 5.2% for the roman font and 3.1% for the italic font.

Most of the improvement seen in these results is the effect of the spelling corrector, although there is also a small but consistent improvement in word accuracy due to natural language correction. Spelling correction and natural language correction have a much bigger effect on *application accuracy.*

Sending the uncorrected OCR output directly into the natural language system for processing without correction leads to a 73% average weighted error rate as measured by the ARPA error metric defined for ATIS[14].

This high error rate is due to the fact that the entire sentence must be nearly correct in order to correctly perform the database task. Spelling correction improves the error rate to 33%. Finally, with natural language correction, the application error rate improves to 28%.

5. TEXT UNDERSTANDING

Once the ASCII text is assembled in the correct reading order, we can employ text understanding to support a document application. The text understanding module is a generic, fast, robust, domain-independent, natural language processing system consisting of components for syntactic and semantic processing. The syntactic component includes a statistical part-of-speech tagger developed at the University of Pennsylvania[15], and a robust Prolog parser developed at New York University[16]. The semantic component is a case frame interpreter developed at Unisys.

Trigram and unigram probabilities are used by the part-of-speech tagger to assign parts of speech to words in a corpus. It was trained on the 1M word Brown corpus of general English. The tagger includes models for part-of-speech assignment for unknown words and heuristics for recognizing proper nouns. Thus, unknown words resulting from OCR errors, new proper names, and unusual technical terminology can be correctly tagged, and will not cause problems during later processing.

The tagged articles are input to the parser, which uses a grammar consisting of 400 productions to derive regularized parses which reflect each sentence's logical structure. The parser achieves domain independence by using an 80,000 word lexicon, and by using time-outs and a skip-and-fit mechanism to partially parse sentences for which it lacks appropriate productions. Thus, no sentence will be left completely unanalyzed; the system will extract as much information as it can from every sentence. Average parsing time is approximately 0.5 sec/sentence.

The semantic component represents the meaning of the sentence in the form of a *case frame,* a data structure which includes a *frame*, representing a situation and typically corresponding to a verb, as well as its *case roles,* which represent the participants in the situation and which typically correspond to the subjects and objects of verbs[17]. The semantic component of IDUS takes the parsed output and maps it into case frames reflecting each sentence's semantic content.

The domain-independent case-frame generator uses unigram frequencies of case frame mappings based on training data obtained from a corpus in the Air Travel Planning domain to select the most probable case frame mappings for the arguments of verbs[7, 18]. This training data is supplemented by additional mapping rules based on the semantics of prepositions occurring in the University of Pennsylvania Treebank[19].

6. APPLICATIONS

We have developed two applications to demonstrate the utility of document understanding from hardcopy material: text retrieval and automatic hypertext generation.

Articles found on multi-article pages during the document layout analysis are extracted separately and stored in a corpus. The text retrieval interface to IDUS allows the user to pose a query in natural language.

A natural language query is made and a search for matches in the tagged corpus is performed based on part of speech tagging of the query. Every sentence in the tagged corpus receives a score based on: (1) the ratio of the number of distinct words and their associated part-of-speech in common between the query and the sentence and (2) the total number of words in the sentence and the query. The score of an entire article is the maximum score of any sentence in the article. Grammatical function words such as prepositions, conjunctions, and pronouns are ignored during the matching process. By basing the comparison on tagged words rather than raw words, fewer irrelevant articles are retrieved because parts-of-speech corresponding to grammatical function words can be ignored and because words in the query and corpus must have the same part of speech in order to count as a match. After articles have been retrieved, a menu then pops up with a list of "hits". The user then has an option to examine the ASCII text associated with the article and/or the image from which the article came which affords the user the full richness of context. An example of the text retrieval application is shown in Figure 2.

We have also developed a hypertext generation module which

Uncorrected OCR Output:

```
What is the latest fgght fiom Phaadelphi& to Boston
```

After Spelling Correction:

$$\texttt{What is the latest} \left\{ \begin{array}{l} \texttt{eight} \\ \texttt{flight} \\ \texttt{night} \\ \texttt{right} \end{array} \right\} \texttt{from Philadelphia to Boston}$$

After Natural Language Correction:

```
What is the latest flight from Philadelphia to Boston
```

Figure 1: **Successive improvement in OCR output occurs with spelling and natural language correction.**

converts the output of the IDUS system to a hypertext representation. The target of this application was legacy technical manuals.

An initial prototype system was developed that generates SGML[20] and hypertext[21] links from raw OCR output. The output of this prototype is provided directly as input to the Unisys hypertext system IDE/AS,^{TM}, which encodes hypertext links as SGML tags. The main steps are the classification of the individual text lines, the grouping of them into functional units, the labeling of these units and embedding the corresponding SGML tags.

We employ a two-stage fuzzy grammar to classify and group lines into logical units regardless of physical page boundaries in the original document. The first stage tokenizes each line, recognizing key character sequences, such as integers separated by periods, which would be useful downstream in determining line type, in this case some level of (sub)section. This method allows us to overcome certain kinds of OCR errors. The second stage finds a best match of the sequence of tokens representing each line against a dictionary of token sequences to arrive at a line classification. Classifications include page header, page footer, blank line, ordinary line, four levels of section indicator lines and list item lines. The validity of the individual line classifications is then checked in context. If a line type does not fit with respect to its neighboring lines, it is reclassified.

A further set of rules is applied to group lines together into functional units, label the units and choose titles for them. With this preparation, the SGML tags upon which the IDE/AS^{TM} hypertext system relies are generated and inserted.

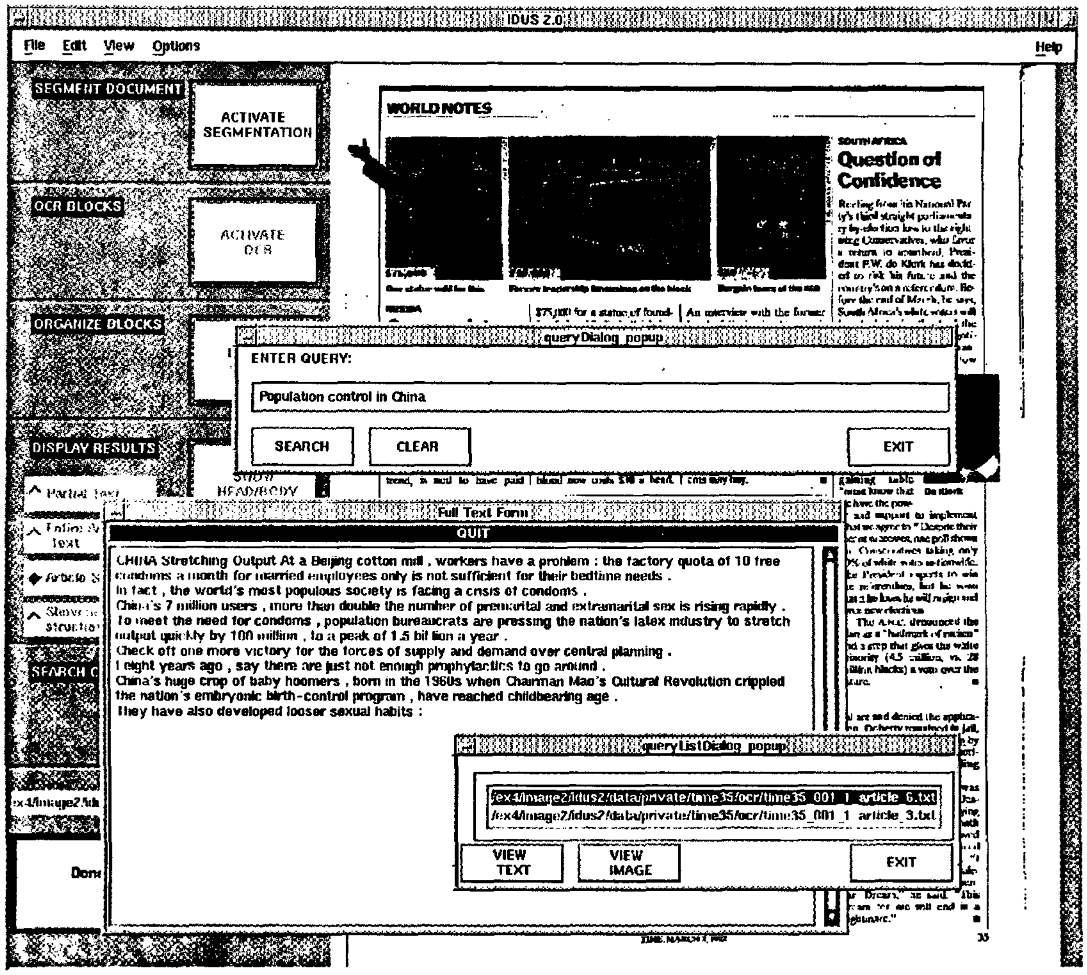
From the unit labels and titles an active index is generated; a click on any entry takes the reader directly to the corresponding frame. Additional processing of each line searches for cross-references to paragraphs (e.g., "See paragraph 1.2"), tables and figures in order to insert tags which transforms the references into "hotspots". Paragraph, table and figure cross-references are also activated. Text blocks are fed to the hypertext generation module from IDUS as lines of ASCII characters.

7. FUTURE PLANS

Currently, we are working on enhancements to IDUS include the ability to analyze new classes of information, specifically multi-lingual business documents and newspapers. This involves extending logical analysis to multiple page documents and the tighter coupling of text and image understanding.

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*, (Tsukuba City, Japan), October 1993.
2. S. L. Taylor, M. Lipshutz, and C. Weir, "Document structure interpretation by integrating multiple knowledge sources," in *Symposium on Document Analysis and Information Retrieval*, (Las Vegas, Nevada), 16–18 March 1992.
3. S. Tsujimoto and H. Asada, "Understanding multi-articled documents," in *10th Int. Conf. on Pattern Recognition*, (Atlantic City, NJ), 16–21 June 1990, pp. 551–556.
4. J. Fisher, "Logical structure descriptions of segmented document images," in *First International Conference on Document Analysis and Recognition*, (Saint-Malo, France), 30 September – 2 October 1991.
5. D. A. Dahl, L. M. Norton, and S. L. Taylor, "Improving OCR accuracy with linguistic knowledge," in *2nd Symposium on Document Analysis and Retrieval*, Apr. 1993.
6. D. A. Dahl, L. Hirschman, L. M. Norton, M. C. Linebarger, D. Magerman, and C. N. Ball, "Training and evaluation of a spoken language understanding system," in *Proceedings of the DARPA Speech and Language Workshop*, (Hidden Valley, PA), June 1990.
7. L. M. Norton, M. C. Linebarger, D. A. Dahl, and N. Nguyen, "Augmented role filling capabilities for semantic interpretation of natural language," in *Proceedings of the DARPA Speech and Language Workshop*, (Pacific Grove, CA), February 1991.
8. L. Hirschman and J. Dowding, "Restriction grammar: A logic grammar," in P. Saint-Dizier and S. Szpakowicz, eds., (*Logic and Logic Grammars for Language Processing*), pp. 141–167. Ellis Horwood, 1990.
9. M. Palmer, *Semantic Processing for Finite Domains*. Cambridge, England: Cambridge University Press, 1990.



ENTER QUERY: Population control in China

```
CHINA Stretching Output At a Beijing cotton mill , workers have a problem : the factory quota of 10 free
condoms a month for married employees only is not sufficient for their bedtime needs .
In fact , the world's most populous society is facing a crisis of condoms .
China's 7 million users , more than double the number of premarital and extramarital sex is rising rapidly .
To meet the need for condoms , population bureaucrats are pressing the nation's latex industry to stretch
output quickly by 100 million , to a peak of 1.5 bil lion a year .
Check off one more victory for the forces of supply and demand over central planning .
I eight years ago , say there are just not enough prophylactics to go around .
China's huge crop of baby boomers , born in the 1960s when Chairman Mao's Cultural Revolution crippled
the nation's embryonic birth-control program , have reached childbearing age .
They have also developed looser sexual habits :
```

```
/ex4/image2/idus2/data/private/time35/ocr/time35_001_1_article_6.txt
/ex4/image2/idus2/data/private/time35/ocr/time35_001_1_article_3.txt
```

Figure 2: Text retrieval application for the IDUS prototype system.

10. C. N. Ball, D. Dahl, L. M. Norton, L. Hirschman, C. Weir, and M. Linebarger, "Answers and questions: Processing messages and queries," in *Proceedings of the DARPA Speech and Natural Language Workshop*, (Cape Cod, MA), Oct. 1989.
11. P. Price, "Evaluation of spoken language systems: the ATIS domain," in *Proceedings of the Speech and Natural Language Workshop*, Morgan Kaufmann, 1990, pp. 91–95.
12. C. Fillmore, "The case for case," in Bach and Harms, eds., (*Universals in Linguistic Theory*), pp. 1–88. New York: Holt, Reinhart, and Winston, 1980.
13. D. A. Dahl and C. N. Ball, "Reference resolution in PUNDIT," in P. Saint-Dizier and S. Szpakowicz, eds., (*Logic and logic grammars for language processing*). Ellis Horwood Limited, 1990.
14. D. S. Pallett, "DARPA resource management and ATIS benchmark poster session," in *Proceedings of the DARPA Speech and Language Workshop*, (Pacific Grove, CA), February 1991.
15. K. W. Church, "A stochastic parts program and noun phrase parser for unrestricted text," in *Proceedings of the Second Conference on Applied Natural Language Processing*, (Austin), ACL, February 1988, pp. 136–143.
16. T. Strzalkowski and B. Vauthey, "Information retrieval using robust natural language processing," in *Proceedings of the Thirtieth Annual Meeting of the Association for Computational Linguistics*, 1992, pp. 104–111.
17. C. Fillmore, "The case for case reopened," in P. Cole and J. Sadock, eds., (*Syntax and Semantics, Volume 8: Grammatical Relations*). New York: Academic Press, 1977.
18. C. T. Hemphill, J. J. Godfrey, and G. R. Doddington, "The ATIS spoken language systems pilot corpus," in *Proceedings of the DARPA Speech and Language Workshop*, (Hidden Valley, PA), June 1990.
19. M. Marcus, "Very large annotated database of American English," in *Proceedings of the DARPA Speech and Language Workshop*, (Hidden Valley, PA), Morgan Kaufmann, June 1990.
20. E. van Herwijnin, *Practical SGML*. Norwell, MA: Kluwer Academic Publishers, 1990.
21. J. Nielsen, *Hypertext and Hypermedia*. Academic Press, Inc., 1990.