IR-164

Automatic Construction of a Personal Reference Library: A Self-Organized File Cabinet

Dawn Lawrie
Department of Computer Science
University of Massachusetts
Amherst, MA 01003*

Daniela Rus
Department of Computer Science
Dartmouth College
Hanover, NH 03755

Abstract

The self-organizing file cabinet is a personal digital library associated with a user's physical file cabinet. It enhances the file cabinet with electronic information about the papers in it. It can remember, organize, update, and help the user find documents contained in the physical file cabinet. The system consists of a module for extracting electronic information about the papers stored in the file cabinet, a module for representing and storing this information in multiple views, and a module that allows a user to interact with this information. The focus of this paper is on the design and evaluation of the self-organized file cabinet.

1 Introduction

Most ordinary offices contain a desk with a computer and lots of papers scattered on it. They also have file cabinets and shelves full of books. If one considers only the physical documents in this room, it is highly probable that the owner of the office could only locate a proportion of what is in it. She may remember a great article she read a year ago but can't remember who wrote it, and therefore, cannot retrieve it to show it to an interested student. Or perhaps a deadline is approaching and she needs a document, but can't remember where she put it. Is it still in the file cabinet? Or did she already take it out and put it on her desk?

These problems are faced by many people today, and the only remedy they see is that they need to be better organized. There is another solution. Rather than trying to alter human nature, one can go to a computer for help. By sharing the task of organization, one could be a lot more efficient. All that is needed is one more tool in the office, a scanner. Then a link between her file cabinet and the physical world is created. Each new document that is placed in the file cabinet is first scanned. A drawer is chosen by the user or the system and then placed in the front of the drawer. Now the author is no longer the only information she can use to search for the document she wants to give that student. She will be able to query the system through text strings or color or both to find the document

In this paper we present such a system, a self-organized file cabinet. We can view this system as a personal digital library associated with one's files. We do not regard digital libraries as replacements of traditional libraries. For sociological, economical, and legal reasons we envision digital collections that co-exist with paper collections. When the same information is present in paper and digital form it is important to coordinate the indexing, cataloging, and access patterns.

The self-organizing filing cabinet is a system that enhances a physical filing cabinet with electronic information to support better filing and queries. The system works as follows: each document is scanned before being filed away. The scanned image can be filtered with OCR to recover the words of the document. The scanned image can also be processed by color and other layout filters such as those we developed in our previous work [RS95b, RS97]. This information is added to a database. The scanned document is then compared with the rest of the database to identify the best physical filing location. The contents of the database are kept organized as a hierarchy of clusters. When looking for information in this physical filing cabinet, a digital query can be used to compute location of all relevant documents in the physical space. The database can also be used to browse the contents of the collection.

This paper also contains three evaluation studies: the first study measures the performance of the system; the second study measures the effectiveness of using the file cabinet to organize information for the

*A majority of the work was completed while an undergraduate at Dartmouth College

user; finally, a small user study explores the utility of such a system.

1.1 Related Work

Efforts to enhance physical environments with electronic information include The Intelligent Room project at the AI Lab at MIT and the ALIVE project at the Media Lab at MIT. The goal of the Intelligent Room project is to create a room surveyed by cameras that can recognize and understand physical gestures. Progress on this project has been reported in [Tor95]. The ALIVE project allows users to interact with animated electronic characters and has been described in [Mae95].Other related projects include efforts from Euro Xerox and Hitachi to create interactive desks, where the user can write with a stylus pen on the desk top. The desk-top consists of a display that can capture the user's input. A camera mounted on the desk top is used to project on the desk top rather than extract information [AM+94]. Finally, [RS97] describe a self-organizing desk.

The Intelligent Room [Tor95] has been design to help groups of people collaborate in a shared meeting space. The Intelligent Room offers many features which include: the computer's ability to track the room's occupants and understand human gestures; intelligent agents that "autonomously navigate National Information Infrastructure to retrieve information or to provide computational service to the room" [Tor95]; and agents that keep a repository of data and plans in a persistent database.

The room uses cameras to tract the occupants of the room. The computer selects two of the most interesting views of the image to track the proceedings in the room. The room watches for humans to point at objects in the room, and if the human is pointing at a place where the computer registers a command, the command will be carried out. They are also implementing speech recognition commands to enhance a speaker's presentation.

The designers of the desk [AM+94] believe that the desk is the center for intellectual human activities and that the object will retain its importance even with the increasing capabilities of computers. For this reason, they have developed a desk that allows the user to interactive with the computer as if it were a more traditional desk. The components of the desk include a computer with keyboard and mouse. It is important to have large displays to make it easy to view the working environment. They use a 26 inch CRT display as well as an A1-sized transparent tablet as the desktop display with a pen-input facility.

The InteractiveDesk allows the user to add personal notes in writing to files. It also allows one to link real world objects to files on the computer. This plays on the principal that a person would rather retrieve a scrapbook instead of digging through the hierarchy of directories to find the documents of interest. The desk uses a camera to recognize these objects. The camera is also used to anticipate which work space the user to using. This determines which monitor is used to display the items.

The self organizing desk [RS97] shares many features with the self-organizing file cabinet since the desk was the predecessor to the file cabinet. The desk uses a camera to monitor the traffic on a desktop. It takes pictures of papers and processes the contents of a paper so that a user can submit queries to the computer when trying to find a paper rather than sifting through papers on the desktop. It then keeps track of the location of the paper as it moves around on the desktop. The system relies on the fact that papers tend to be stacked on the desk so it allows one to move a stack of pages at a time without loosing track of a document.

Our virtual file cabinet system draws from progress made in several areas: self-organizing systems [Koh90, Sam69, CKP93, APR98, APR99], information retrieval and organization [Sal91, SA93, Wil88, RA95], robotics and vision [MRR96, HKR93], automated document structuring [TA92, MT*91, RS95b, JB92, NSV92], and user interfaces.

2 Example

Figure 1 shows a typical setup for the self-organizing file cabinet. The user is presented with a GUI. The right side of the GUI contains a picture of the physical file cabinet setup in the office. The left-hand side gives an electronic visualization of the physical file cabinet. We call this image the virtual file cabinet. The bottom part of the file cabinet contains a series of buttons that can trigger the retrieval and organization capabilities of the system.

The user would typically add a paper to the file cabinet by scanning it. After the electronic information is captured from the scanned image (in the form of test, images, color, and layout information) the user has the choice of (1) inputting the drawer location for the new document in the file cabinet library; or (2) asking the system for a drawer recommendation, based on the information that is currently contained in the system.

The user can query the file cabinet based on text, color, layout information about the document, or a combination of all these types. The system can answer queries such as "where is the paper with the red

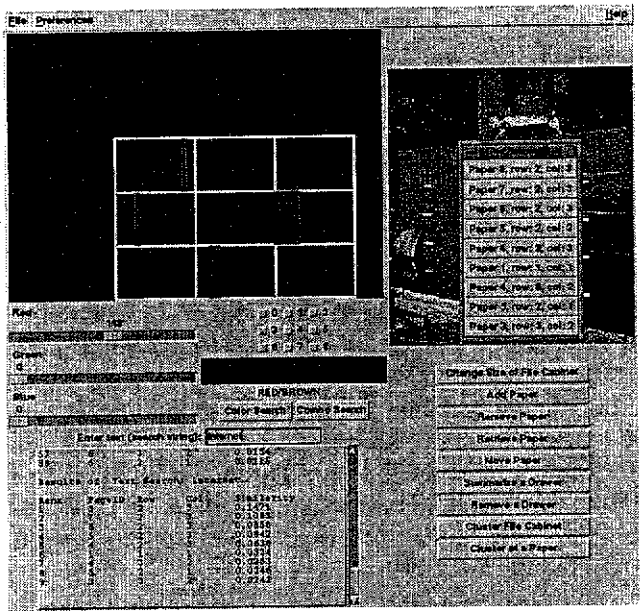

Figure 1: A snapshot of the virtual file cabinet system. This figure shows the basic GUI for the virtual file cabinet. It also shows the response of the file cabinet to a query. Note that each drawer is represented in a side view so that the approxinate location of the documents found by the system can be marked on the virtual drawers.

table in the lower right hand corner?" Or "which papers are about intelligent agents and have red pictures on the left side?" The library system responds by highlighting the location of the relevant documents in the virtual file cabinet. These locations correspond to the physical location of the documents in the real file cabinet. The user can then proceed to the relevant physical drawer to retrieve the document.

The user can also ask the file cabinet library for a table of contents. The self-organizing file cabinet employs an information organization algorithm that presents the user with a visual summary of all the topics present in the file cabinet. The user may select a topic, which causes the highlighting of the location for all the documents contained in the physical file cabinet that are relevant to the given topic.

3 The System Description

The vision of a self-organizing file cabinet is achieved by the system shown in Figure 2. Our main goal in creating this system was to develop tools to create automatically a reference library associated with the documents in an office space. We believe this to

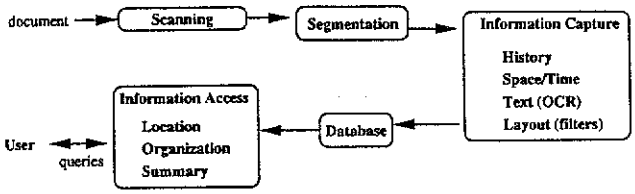

Figure 2: The system components of the self-organizing file cabinet. Documents to be filed are first passed through a scanner. The scanned images are filtered by OCR and color. Other image segmentation modules can easily be added that would recover the figures, table, and other layout information. Each of the features is then entered into a database that contains multiple views of the documents according to these features. In addition, the database contains a field for the physical location of the document in the real file cabinet. The user has the option of specifying a physical location for the document in the file cabinet, or of asking the system for a recommendation based on the current contents of the file cabinet. The database is indexed, organized, and available for queries.

be a useful system because humans in general have great difficulty maintaining organized systems of documents. This difficulty is often due to the fact that one document could belong logically in several different locations. Thus, there may be inconsistencies in placing and retrieving physical documents.

Our system addresses this problem by maintaining electronically an image of the physical cabinet. The virtual cabinet retains a picture of each document in it. This picture is computed by a scanning operation which is done manually by the user. We believe that although the initial time spent scanning a document can be much greater than the time to place a document in a drawer, the ease with which one can find that document out-ways the initial time investment.

The user can query the virtual file cabinet by specifying a combination of word and color information. The system also offers the capability of finding other similar documents stored in the file cabinet. In addition, the system can present the user with a "table of contents" that captures the topics and subtopics of the documents contained in it. Finally, the system can help a user file documents by suggesting the best drawer for the current document based on the contents of the file cabinet.

There are four main components to the the self-

organized file cabinet: (1) the database that captures multiple representations for each document; (2) the search engine used to query the cabinet; (3) the organization component (used to compute a table of contents for the file cabinet and to recommend drawers for filing); (4) the GUI visualization scheme for the file cabinet. The rest of this section is devoted to describing each of these modules.

3.1 The Database

The file cabinet database is structured to mirror a file cabinet in the physical world from the design of the structure to the way in which manipulations are performed. For example, in order to retrieve a paper, one first thinks of retrieving the paper from the cabinet. The first question that needs to be answered is which drawer. Once a drawer is known, the task can be reduced to retrieving a paper from the drawer, and the files are thumbed through to find the correct one. We designed classes in C++ to make normal operations on a filing cabinet as natural and intuitive as possible. The parent structure is the filing cabinet itself. A filing cabinet consists of drawers. The implementation structure of the drawers is a dynamically allocated array.

To support queries that combine textual with layout information, we created a database indexed by words, color, and physical information. The database comprises a collection of inverted indices, one for each attribute. An inverted index associates each attribute instance with a list of documents containing it. The advantage of this representation scheme is a speed-up in search: given a specific attribute (a word, a color, *etc.*), the list of documents containing this attribute is available in constant time. Each index is computed incrementally by a filter that operates on the document's scanned image. Currently, our filter library consists of two filters: OCR and color, but the architecture is expandable and we will add new filters in the future.

OCR. The OCR filter is built-into the scanner.

Color. The color filter works by building a color histogram annotated with layout information for each object. The filter determines the 24 most prevalent colors occurring in the document and the location of each color. Location is a layout attribute determined by placing a 3×3 grid on the paper.

Space, time, and history. Each paper gets assigned a location in the file cabinet using coordinates that can be provided by the user or computed by the system. Each paper is also time stamped. In addition, each paper is associated with a list of papers before it and papers after it in the physical drawer, to capture the drawer history. This information is necessary to compute approximately the location of a document within a drawer.

3.2 Filing and Retrieving Documents

The filters defined in Section 3.1 generate a web of representations for each document. We compile this multiplicity of representations in a database. This database supports the following desk operations: adding a paper to the file cabinet, removing a paper from the file cabinet, computing a table of contents for the file cabinet, and computing the best location for filing a new document. The information in this database changes dynamically, as driven by these operations. In response to each event, the database is updated automatically.

The file cabinet can be queried with keywords, with an entire document (full text), with color and layout information, and any boolean combination of these attributes. We use an augmented version of the *Smart System* [Sal91], which is a sophisticated text-retrieval system. We augmented Smart to also support color and layout indices. Smart copes well with partially corrupted (by OCR) text. Its basic premise is that two documents are similar if they use the same words. Documents and queries are modeled as points in a *vector space* defined by the important words occurring in the corpus. When all texts and text queries are represented as weighted vectors, a similarity measure can be computed between pairs of vectors that captures the text similarity. We use this similarity measure as the basis for computing hyper-links between documents that are similar to each other in this statistical framework.

The textual information contained in the documents in the file cabinet is organized by topic using *the star clustering algorithm* we developed [APR98, APR99] on the document space. The star algorithm gives a hierarchical organization of a collection into clusters. Each level in the hierarchy is determined by a threshold for the minimum similarity between pairs of documents within a cluster at that particular level in the hierarchy. This method conveys the topic-subtopic structure of the corpus according to the similarity measure used.

We have developed an off-line version of the star algorithm to handle static collections, and an on-line version to handle dynamic collections [APR98, APR99]. Both algorithms are used in the file cabinet system. These two algorithms compute clusters

induced by the natural topic structure of the space. To compute accurate clusters, we formalize the clustering problem as one of covering a thresholded similarity graph by cliques. Covering by cliques is NP-complete and thus intractable for large document collections. Recent graph-theoretic results have shown that the problem cannot even be approximated in polynomial time [LY94, Zuc93]. We instead use a cover by dense subgraphs that are star-shaped[1], where the covering can be computed off-line for static data and on-line for dynamic data. We show that the off-line and on-line algorithms produce high-quality clusters very efficiently. Asymptotically, the running time of both algorithms is roughly linear in the size of the similarity graph that defines the information space. We have derived lower bounds on the topic similarity within clusters guaranteed by a star covering, thus providing theoretical evidence that the clusters produced by a star cover are of high-quality (in other words, have a high degree of precision). The file cabinet uses the off-line version of the star clustering algorithm to organize an existing file cabinet and extract a table of contents. The file cabinet uses the on-line version of the star algorithm to recommend a drawer and to insert a new document in an existing file cabinet organization.

3.3 The Graphical User Interface

The GUI was spatially designed with four quadrants as shown in Figures 1 and 3. In the upper left-hand corner a graphical representation of the file cabinet appears. In the upper right hand corner, a picture of a file cabinet is shown. In the lower left-hand corner, the search manipulations appear. The output from the search is printed in the scrolling text area. In the lower right hand corner there are a series of buttons that allow for the manipulations of documents and the file cabinet as well as buttons pertaining to the clustering by topic features.

The display of the file cabinet draws a picture of the file cabinet. The file cabinet is blue with drawers outlined in yellow on a black background. Each drawer is actually a side view of the open drawer rather than a front view. This enables the interface to display the approximate position of a paper within the drawer by drawing a red line to highlight the document. When a drawer becomes full, many papers could be represented by one line. The results of most searches yield more than one paper. In order to figure out exactly which paper is the one the user is interested in, the user can highlight a specific paper in green.

The lower left part of the screen is devoted to the searching mechanisms. The top part of this section is used to construct colors by mixing red, green, and blue. This forms twenty-four different colors. The constructed color is displayed to the right of the sliders. Above this color display are nine radial buttons. These are used to select portions of the page to search. When the "Color Search" button is pressed, search commences. Below the color search part of the display is a line to enter text queries. The text query commences by pressing enter in that box. Next to the "Color Search" button is the "Combo Search" button. A combination search takes the results of a text query and color search and displays only those papers that are found by both searches. The results of queries are displayed in the scrollable text area at the bottom of the screen. The search identifies pages by page identification number, drawer row, and drawer column. The text query also includes information about the simi-

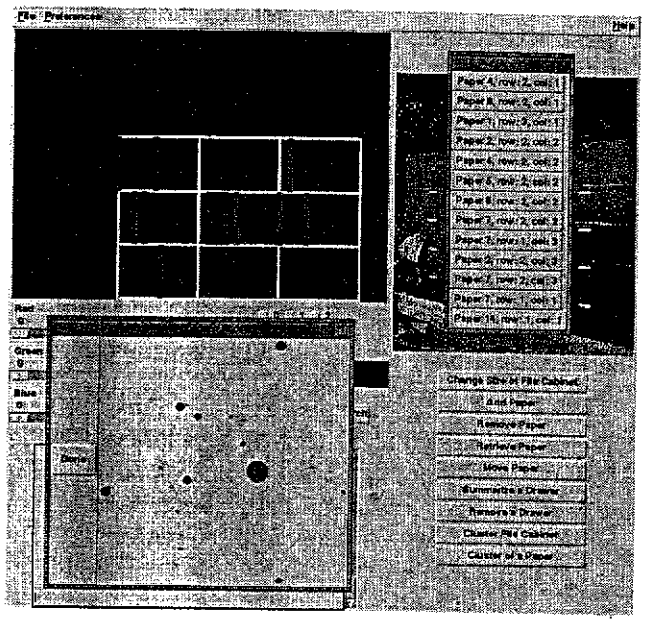

Figure 3: The Graphical User Interface to the virtual file cabinet shows the output of the system in response to a request for the table of contents. Each blob represents a topic contained within the file cabinet that was computed by the star algorithm. The distance between the blobs is proportionate to the distance between their corresponding topics computed by the star algorithm in the vector space model. The user can select a topic by clicking on a blob. This causes the highlighting of the location of all the documents contained in that cluster in the file cabinet.

[1] In [SJJ70] stars were also identified to be potentially useful for clustering.

larity of the document to the query. Documents are listed in rank-order.

The lower right part of the screen consists of all the buttons used to manipulate the file cabinet as well as the ones related to clustering documents by topic. These buttons included changing the size of the file cabinet; adding, removing, showing, and moving a paper; summarizing and removing a drawer; and clustering the whole file cabinet or finding the clusters that include one particular paper to discover similar papers.

In the Menu Bar there are two pull down menus. One is "File" which allows one to load and save a file cabinet and to quit the application. To load and save one is asked to specify a full path directory. The second pull-down menu is labeled "Preferences". There are two menu items. The first, "Set Image Directory," asks the user to specify the directory where all scanned documents in the file cabinet are located. The second, "Recommendation Threshold," allows the user to set up the range of the thresholds to be used when the file recommends the placement of a document and at what frequency thresholds will be used within the range.

4 Experiments

We have identified three different ways of evaluating the virtual file cabinet system. First, we measured the performance of each module in the system. Second, we devised an experiment for evaluating the most complex function of the system, which is the recommendation of the filing location. Finally, we did a small user study to find out people's reaction to such a system.

4.1 Performance

The system was tested to make sure that all operations occurred at a reasonable speed so that users would not become impatient with the system. The results of this test are shown below. Note that the time taken by all operations other than scanning is given in clock ticks. The scanning time is given in minutes.

Function	*Time*
Scan	2 mins.
Initialize	85.59
Load	241.13
Save	103.77
Change File Cabinet Size	8.27
Add Page	334.71
Recommend Drawer	814.90
Remove Page	572.55
Move Page	0.72
Print Drawer	5.16
Delete Drawer	0.09
Find - Single	0.54
Find - Group	16.80
Cluster on a paper	871.44
Cluster Cabinet	710.78
Search by Color	8.18
Smart Search	61.80
Combo Search	54.80

4.2 Evaluating the Filing Location Recommendation

The most computationally complex part of the system is the drawer recommendation operation. This feature of our system leads to a virtual file cabinet that is more than just a mirror image of a real file cabinet. The ability of identifying the most logical filing location for a new document leads to a system with limited self-organizing capabilities.

We chose two measures for evaluating the filing location recommendation features. The first measure captures how well the cabinet was able to evenly distribute the papers throughout the drawers. The second measure captures the quality of the topic and subtopic clusters computed by the system.

Our experimental document collection consists of the pages from Communications of the ACM July 1994–Volume 37, Number 7, a special issue on Intelligent Agents [CAC93]. Some 88 of the total pages of this issue contain articles about different aspect of intelligent agents. We regarded each page in the magazine as a separate document. We hypothesized that all pages within an article would be more closely related to each other than pages of other articles. For the most part this proved to be correct, except for on a few notable exceptions where pages were not related to any other page in the magazine. One was in the article "A Conversation with Marvi Minsky about Agents" where American society and education are discussed. Occasionally, a large portion of final page of an article consists of references, which greatly effects its similarity to previous pages.

We set up three different types of experiments. In

the first group of experiments we entered the entire collection at a range of thresholds 0.45 to 0.5 (in increments of 0.05) for the threshold parameter used by the star clustering algorithm. In one run the pages were entered in order (since it is a deterministic process) and 10 were entered in an order generated by a random order generator. In the second group we entered all but the first pages of each article using the same threshold range as in the first experiment, one in order and 10 in a randomly generated order. (We chose this variation because we observed that the introduction pages of all our articles tended to be very similar, as they covered similar issues.) In the third group of experiments we used a higher range of thresholds, 0.55 to 0.75. The entire collection was used. One run consisted of pages that were entered in order and ten in random order.

The high threshold variation gave the best results as far as distribution of papers among the file cabinet drawers. The largest drawer had an average size of 26.0 papers. The smallest drawer had an average size of 20.2. The other two variations performed almost equally, but the distribution of papers was more uneven. The results are shown as follows.

Experiment	*Smallest Drawer*			*Largest Drawer*		
	Avg	*Min*	*Max*	*Avg*	*Min*	*Max*
Exp. 1	15.2	10	19	39.3	28	56
Exp. 2	15.6	11.6	18.5	37.4	27.8	47.5
Exp. 3	20.2	19	21	26.0	23	31

In contrast the variations that used a lower threshold performed significantly better at grouping pages from a particular article. When all pages were filed at thresholds of 0.5 and 0.45, 82% of pages of an article grouped together when filed in random order. This value increased to 91% when the documents were pages were filed in order. Without the first page of each article, no preference was given towards the pages being filed in order verse random order. Eighty-two percent of the pages of an article were found grouped in a drawer when the pages were file in order and 83% when the pages were filed in random order. The high threshold variation only had 70% of the pages of an article group in the same drawer when pages were filed in random order. This number decreased to 62% when the pages were filed in order.

4.3 User studies

User studies were performed to evaluate how well the interface was able facilitate use of the file cabinet and how useful the users thought the overall system was. Besides having users use the system, the Self-Organized File Cabinet was presented at a Dartmouth College Poster Symposium. The people that learned of the system were intrigued and interested in learning more about the system.

The user studies revealed attributes of the file cabinet that were difficult or uncomfortable to use. The design in general, however, was easy to understand. To instruct people in using the file cabinet, we gave a brief demonstration that pointed out buttons and showed how to cluster the file cabinet. Finally, we demonstrated a color, text, and combination search before quitting the application and allowing them to use it. We gave them a piece of paper with sixteen tasks as follows:

- load: /usr/plum/lawrie/save
- set pref: /usr/plum/lawrie/images
- Find out how the documents in the file are related.
- Examine a paper in the file cabinet in more depth.
- Make the file cabinet bigger.
- Add page 102 to a new drawer
- Print the drawer.
- Move the paper to another drawer.
- Print that drawer.
- Find the paper in its new location.
- Remove a drawer.
- Remove the paper that you added.
- Make the file cabinet smaller.
- Find the articles written by Doug Riecken
- Look at page 63 in the magazine: what might you remembered about this page a month later and then try to find it based on those details.
- Search for the article on pages 72-76.

The main complaint of the users as shown in Figure 4 (below) was that they didn't like trying to construct a color, and they wanted to do more things with the mouse and less with the keyboard. In general they wanted the capability to review actions performed by the system. Some users also would like to be able to assign titles to pages rather then see the identification number assigned by the file cabinet. Finally, many expressed an interest in adding information to what smart was already indexing to emphasize their interest.

Characteristic	*People*
Dislike the Color Set-up	5
More Mouse Use	5
List Actions	4
Logical Search Operators	3
Titles Papers	3
Add Keywords to Papers	2

Figure 4: This table lists some of the reactions by the users and how many of those 6 users had the same reaction.

5 Discussion

We believe that the experiments show that the virtual file cabinet is a usable and useful system for computing and maintaining references to a library of documents contained in a physical file cabinet. Most of the file operations are very fast, seeming nearly instantaneous to the user. The experiments performed on the ability of the file cabinet to recommend drawers shows that it is this functionality can be used. The user studies show that, although the interface design was lacking in some areas with minor adjustments it would facilitate use of the system. Further discussion of the experiments follows.

Within the performance of the file cabinet, there are two time consuming tasks. The first is scanning the document. This only needs to be done once for a document, so it is believed that it would not be too much of deterrent to using the system considering the benefits once the document is scanned. The other is the recommendation of a drawer when a large range of thresholds are used, which is another feature that is not used as often over the life of a document in the file cabinet. These time consuming operations are offset by the speed that searches are performed, a much more frequent activity in the file cabinet.

Based on the results of the drawer recommendation, the most important choice the user will make is choosing a range of thresholds to find related documents. This requires the user to have an idea about what the purpose of the file cabinet before one begins adding documents to it. Of course, the user can change the range of thresholds at anytime, but documents already filed will stay in current places. The importance of drawers containing similar topics is to facilitate the gathering of documents on any given topic. If all documents are found in one or two drawers it will be simpler to retrieve them. For almost all articles (93%) filed in the cabinet, no matter what thresholds were used, at least half of the pages could be found in one drawer.

The user studies showed that constructing a color was very problematic. One thing is that people don't really know which mix of colors makes the one they are looking for so it is not very fast to have to build each color. The second problem was that while someone might call a color one name, they might select another if they were trying to match colors. For example, a person might call a color "green" when the computer identifies it a "yellow-green." A selection method would elevate this ambiguity.

Users found the "Find Results" window intimidating because all the buttons basically looked the same. They were not easily distinguished, so people forgot which button they had clicked. The last button clicked should change to some other color besides the default color, so that the user can easily tell which paper she is looking at.

The users wanted the scrolled text on the lower left side to be used as a record for all actions instead of just those involved with search. This would enable one to keep track of all the file manipulations during a session.

Adding keywords to the document was a feature that many desired. If only the first page of a paper is scanned and added to the file cabinet, it is possible that the reason the person is filing the paper is not discussed on the first page. It would be very easy to append a copy of the OCRed file with the added keywords and have smart index the new file. The keywords could be kept with the filter data and if in the future, the user changed the key words, the paper could be re-indexed by Smart. One user also desired that his keywords be given more weight than what appeared on the page. This might also be able to be implemented using Smart.

6 Acknowledgements

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. Any opinions, findings and conclusions or recommendations expressed in this material are the author(s) and do not necessarily reflect those of the sponsor.

References

[APR98] J. Aslam, K. Pelekhov, and D. Rus, Static and Dynamic Information Organization with Star Clusters in *Proceedings of the 1998 Conference on Intelligent Knowledge Management*, Washington, DC (November 1998).

[APR99] J. Aslam, K. Pelekhov, and D. Rus, A practical clustering algorithm for static and dy-

namic information organization, in Proceedings of the 1999 Symposium on Discrete Algorithms (SODA99), Baltimore, MD (January 1999).

[AM+94] T. Arai, K. Machii, S. Kuzunuki, and H. Shojima, InteractiveDESK: A computer augmented desk which responds to operations on real objects, in *Proceedings of Computer Human Interactions*, 1994.

[CKP93] D. Cutting, D. Karger, and J. Pedersen. Constant interaction-time scatter/gather browsing of very large document collections. In *Proceedings of the* 16^{th} *SIGIR*, 1993.

[HKR93] D. Huttenlocher, G. Klanderman, and W. Rucklidge, Comparing images using the Hausdorff distance, in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(9):850–863, 1993.

[JB92] A. Jain and S. Bhattacharjee. Address block location on envelopes using Gabor filters. *Pattern Recognition*, vol. 25, no. 12, 1992.

[Koh90] Teuvo Kohonen, The self organizing map, in *Proceedings of the IEEE*, 78(9):1464–1480, 1990.

[LY94] C. Lund and M. Yannakakis. On the hardness of approximating minimization problems. *Journal of the ACM* 41, 960–981, 1994.

[Mae95] P. Maes, Artificial Life meets Entertainment: Interacting with Lifelike Autonomous Agents, Special Issue on New Horizons of Commercial and Industrial AI, Vol. 38, No. 11, pp. 108-114, *Communications of the ACM*, ACM Press, November 1995.

[MRR96] R. Manmatha, S. Ravela, and E. M. Riseman, Retrieval from Image Databases Using Scale Space Matching, in *Proceedings of the ECCV '96.*

[MT*91] M. Mizuno, Y. Tsuji, T. Tanaka, H. Tanaka, M. Iwashita, and T. Temma. Document recognition system with layout structure generator. *NEC Research and Development*, vol. 32, no. 3, 1991.

[NSV92] G. Nagy, S. Seth, and M. Vishwanathan. A prototype document image analysis system for technical journals. *Computer*, vol. 25, no. 7, 1992.

[RA95] D. Rus and J. Allan. Structural queries in electronic corpora. In *Proceedings of DAGS95: Electronic Publishing and the Information Superhighway*, May 1995.

[RS97] D. Rus and P. deSantis. The self-organizing desk. In *Proceedings of the 1997 International Joint Conference on Artificial Intelligence*, August 1997.

[RS95b] D. Rus and K. Summers. Using whitespace for automated document structuring. To appear in *Advances in digital libraries*, N. Adam, B. Bhargava, and Y. Yesha, editors. Springer-Verlag, LNCS 916, 1995.

[Sal91] G. Salton. The Smart document retrieval project. In *Proceedings of the Fourteenth Annual International ACM/SIGIR Conference on Research and Development in Information Retrieval*, pages 356-358.

[SA93] G. Salton and J. Allan. Selective text utilization and text traversal. In *Hypertext '93 Proceedings*, pages 131-144, Seattle, Washington, 1993.

[Sam69] John W. Sammon Jr., A nonlinear mapping for data structure analysis, in *IEEE Transactions on Computers*, 18(5):401–409, May 1969.

[SJJ70] K. Spark Jones and D. Jackson. The use of automatically-obtained keyword classifications for information retrieval. *Information Storage and Retrieval*, 5:174-201, 1970.

[Tor95] Mark C. Torrance, Advances in Human-Computer Interaction: The Intelligent Room, in *Working Notes of the CHI 95 Research Symposium*, 1995.

[TA92] S. Tsujimoto and H. Asada. Major components of a complete text reading system. In *Proceedings of the IEEE*, vol. 80, no. 7, 1992.

[Tur90] H. Turtle. Inference networks for document retrieval. PhD thesis. University of Massachusetts, Amherst, 1990.

[Wil88] P. Willett. Recent trends in hierarchical document clustering: A critical review. *Information Processing and Management*, 24:(5):577-597, 1988.

[Zuc93] D. Zuckerman. NP-complete problems have a version that's hard to approximate. In *Proceedings of the Eight Annual Structure in Complexity Theory Conference*, IEEE Computer Society, 305–312, 1993.

[CAC93] Special issue on visualization. *Communications of the ACM*, 36(4), 1993.