

UC: A Fluid Interface for Personal Digital Libraries

Lance Good, Ashok C. Popat, William C. Janssen, Eric Bier

Palo Alto Research Center

3333 Coyote Hill Road

Palo Alto, California 94304

{good,popat,janssen,bier}@parc.com

ABSTRACT

An advanced visual interface system is presented for fluid interaction in a personal digital library system. The system employs a zoomable planar representation of a collection using hybrid continuous/quantum treemap visualizations to facilitate navigation while minimizing cognitive load. By providing both fluidity and a means of reading documents within the same visualization, the system obliterates the traditional boundary separating the acquisition of materials from their use. In addition, the system provides a means of streamlining and largely automating the addition of new documents into a collection. The system is particularly well suited to user tasks which, in the physical world, are normally carried out by laying out a set of related documents on a physical desk — namely, those tasks that require frequent and rapid transfer of attention from one document in the collection to another. Discussed are the design and implementation of the system as well as its relationship to previous work.

Categories and Subject Descriptors

H.5.2 [Information Interfaces and Presentation]: User Interfaces—*graphical user interfaces*; H.3.7 [Information Storage and Retrieval]: Digital Libraries—*user issues*; H.3.3 [Information Storage and Retrieval]: Information Search and Retrieval—*search processes*

General Terms

Design, Human Factors, Performance

Keywords

Personal digital libraries, human-computer interaction, zoomable user interfaces, treemaps, desktop search

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

JCDL '05 Denver, Colorado USA

Copyright 2005 ACM X-XXXXX-XX-X/XX/XX ...\$5.00.

1. INTRODUCTION

The persistence of paper as a preferred medium for document interaction, in the face of distinct and growing advantages of electronic representations, has in recent years been well-studied [35]. Perhaps nowhere are the advantages of paper over extant electronic alternatives so strong as in the case where the reader wishes to compare a set of passages appearing in several documents, or to cycle attention among passages from several documents in rapid succession. Such patterns of interaction arise naturally during a variety of knowledge tasks, including the preparation of lectures, the reviewing of papers, the analysis of intelligence briefings, and the writing of a report, among many others. We shall refer to such patterns as “reading from multiple sources,” with “reading” to be taken in its general sense to include such activities as annotation, extraction, summarization, and the like.

We consider reading from multiple sources in the context of a personal digital library. By the qualifier “personal” here we mean that the user (1) already has the right to use all of the data objects in the library, and (2) already has local possession of those objects. An example of such a personal digital library is the result of scanning one’s office filing cabinet onto a local hard drive. Such a library might also include any electronic document for which the copyright statement allows such use.

In recent years, the importance of effective visualization tools and visual interfaces to digital library collections has been recognized [19, 30, 39]. In the case of a personal digital library, an opportunity exists to blur the traditional separation between searching for materials and using them. In particular, provision to use the material can be integrated into the same interface used to interact with the collection. The system presented in this paper is an instance of such an interface.

In the physical world, reading from multiple sources often involves spreading out a set of documents on a large desk or table. During the course of the activity the reader is at various times absorbed in the close study of one document or another, while at other times, be it in transition or in respite, the reader broadens his or her focus momentarily to regard the workspace as a whole. The transitions among these modes of attention is fluid, and the continuity of orientation maintained, by virtue of the persistence of the layout of the physical documents in a single visual field, and through ability of the reader to adjust his or her center and field of attention rapidly and at will.

The contribution of the present paper can be understood

as an attempt to simulate and, at least with respect to navigation, surpass in the electronic realm the affordances of reading from multiple paper sources. Zooming technology is used to simulate on a limited-resolution electronic screen the reader’s ability in the physical world to vary simultaneously the point of foveation and the viewer’s proximity to the object of current attention. Analogous to our ability to simply view any paper document in the physical world independent of the specific machinery that was used to typeset it, and to employ in the reading task certain common tools such as pens, highlighters, magnifying glasses, and post-it notes independent of what brand of paper was used in the composition of the documents, a uniform and consistent interaction interface is provided to every document in this zoomable virtual workspace, irrespective of its native electronic “format” or “type.” This universality is in turn enabled by providing a fast, transparent, and automatic means of converting arbitrary documents of interest (PDF files, Word files, scanned images, etc.) into a common intermediate working representation. Careful attention is paid to supporting and enhancing the navigational cues that get built up in spatial memory during the course of the activity; these enhancements are through the application of advanced visualizations known as *treemaps* in the design of zoomable interfaces. Finally, in common with most current electronic document representations, provision is made for navigation by incremental textual search, but with the significant enhancement that the search results are indicated interactively and dynamically *in situ*.

While the foregoing discussion has focused on the task of reading from multiple related documents in a collection, many of the advantages offered by the proposed system hold for interacting with isolated documents in the collection as well. Specifically, the ability to search, navigate, and browse visually and fluidly through a scalable, spatially persistent representation of a digital library’s contents facilitates finding the right document. Also, the ability to read from any document in the space without leaving the space — that is, without incurring a significant temporal or cognitive cost — enables on-the-spot checking of the relevance, level, and quality of a candidate information source prior to committing to it.

1.1 Universality of representation and consistency of interface

Users are interacting with an ever widening variety of document formats as part of their daily information diet. Each of these formats carries with it implications for how the document can be read, arranged, and annotated based on the capabilities of its native application. Phelps and Wilensky [31] put it well: “picking a format is often tantamount to choosing a browser/editor/viewer with its packaged bundle of features and limitations.” As noted, such a restriction is not present in the physical realm where documents can be consistently spread out on a table, bookmarked, highlighted, scribbled on, and otherwise annotated. The diversity of applications required to interact with different document types also leads to a proliferation of user interfaces, placing the additional burden on the user of having to learn and remember their proper operation. In the paper world, the user can rely on a single set of tools to interact with nearly all documents. This disparity is likely to be felt even more acutely as pen-based computers become more prevalent, and users

are prompted to expect functionality that at least matches that offered by paper on all significant dimensions, including universality and consistency of interface.

Personal digital library systems as well as commercial content management systems can mitigate these problems to some extent by allowing documents to be converted into a common intermediate format for purposes of indexing and viewing. Reading applications can then support this single common format. Yet the latency and overhead experienced by the user in this separate conversion step is burdensome, particularly when contrasted with the comparable action in the physical world, where a new document is incorporated into the workspace by simply placing it there. The need for fast, seamless, and transparent conversion is all the more apparent when one recognizes that the addition of new information sources to the workspace is prompted frequently and integrally within the reading activity. Indeed, the incorporation of new materials, including the extraction and processing of any requisite metadata, is properly viewed as part of the reading process itself, and should be supported within the interface in a way that minimizes distraction and cognitive load.

The current state-of-the-art in support for electronic reading lies on an evolutionary path premised on the idea that the user will interact primarily with a single document for the entire span of the activity; navigation among and transition to other documents was a functionality outside the reading application proper, and either relegated to operating system calls or to separate applications such as browsers. Missing is support for the sort of fluidity and seamlessness required and taken for granted in the physical realm — the ability to spread out a collection of documents on a desk or a table, to quickly moving among them, to compare related sections, and to gain a visual overview of the structure and contents of a collection.

1.2 UC: a system for fluid, seamless interaction

This paper describes a system for reading from multiple documents which addresses the above mentioned shortcomings of current digital reading systems. The system, called *UC* (a name deriving from its initial but no longer used internal name “UpLib Client”) is built on the UpLib personal digital library platform [23], which provides an extensible, format-agnostic, and secure document repository layer. UpLib achieves a degree of universality in accommodating multiple document formats through heavy reliance on two principal transductions of the original document, one into the *image* domain and the other into the *text* domain. Suggestive of their lossy nature, their complementarity, and their collective expressiveness, these transductions are referred to in the UpLib work as the two fundamental *projections*. From the perspective of UC, the image projection facilitates simulation of the visual experience of reading from multiple paper documents, while the text projection enables search-based navigation. By virtue of its leverage of UpLib, incorporating a new document into the UC system becomes largely a matter of producing its two fundamental projections. To accomplish this in a manner that imposes the least amount of latency and cognitive load on the user involved in a reading task, an area of the user’s file system is designated for continual scanning and processing, so that any document placed there is automatically incorporated into the workspace. By

default the spatial layout in the workspace induced by the incorporation of new documents reflect that of the filesystem sub-hierarchy within the designated scanned area, but schemes involving textual clustering, multidimensional scaling, and citation graph relationships are also possible.

For each document in the workspace, the UC system uses an instance of a single consistent reader application, *ReadUp*, described in detail in a parallel submission to the present Proceedings. *ReadUp* provides functionality that simulates and extends many of the affordances of paper, including thumbnail overviews, highlighting, freehand scribbles, sticky notes, bookmarks, and interactive keyword search. The use of this common interface for interacting with individual documents ensures a consistent set of functionality for reading any document in the system while limiting the number of unique tools the user must learn to use.

Despite the compositional relationship between UC and *ReadUp*, the user experiences fluid interaction within which what appears to be a single application. This is accomplished by making every document behave as if it were always open in a separate instance of *ReadUp*, independent of whether it is an object of current focus. Thus, from the user's point of view, there is no need to launch a separate viewer application to interact with a document; one merely zooms, pans, and focuses on a document to interact with it.

In this way, the UC workspace allows the user to search, browse, read, and manipulate the documents via zoomable interactive treemap visualizations. This interface supports standard recall based keyword searching found in existing desktop search tools but also recognition based searching and exploration based on document location and appearance. Moreover, because the document reader appears integrated within the large zooming application, it moves the interaction paradigm away from one of "opening" documents to one where all documents are always available for reading and manipulation, thus achieving support for effectively reading from multiple sources.

The remainder of this paper is arranged as follows. Section 2 summarizes relevant prior work. Section 3 motivates key design aspects of UC by considering those affordances most valuable when reading from multiple documents. A detailed description of the interface aspects of the UC is provided in Section 4. Section 5 discusses implementation issues, and Section 7 concludes.

2. PRIOR WORK

The importance of effective visual interfaces to collections has been recently recognized by a number of researchers [19, 30, 39], but all in the context of remote digital libraries where finding and using materials are constrained to be separate activities.

Several previous systems have used treemaps and zoomable user interfaces (ZUIs) similar to those described in this paper. The original implementation of this idea was in the PhotoMesa image browser [14]. Many of the principles driving our system design are derived from this work. However, PhotoMesa focuses on browsing images and does not explicitly address reading at all. The International Children's Digital Library (ICDL) [21] builds on PhotoMesa by laying out children's book cover thumbnails in zoomable treemaps. Our work differs from the ICDL in that the document collections can be automatically generated. The UC also handles extremely large collections consisting of a variety of docu-

ment types.

Instant Bookplex [17] is another reading system that employs ZUIs for reading. In contrast to our current work, Instant Bookplex focuses on manually constructed collections of scholarly documents. The visualizations used in Instant Bookplex also focus on spatial stability rather than handling the types of large automatically generated document collections found in our work.

There is also a substantial amount of previous work on helping users interact with document collections. The Visual Knowledge Builder (VKB) [36] uses spatial hypertext as an interface to find, collect, organize, and annotate information about documents in digital libraries. This approach differs from ours in that it does not attempt to manage the documents themselves. As a result, VKB does not try to change the cost structure of reading *per se* but focuses instead on sensemaking through metadata about the documents.

Another popular technique used to explore collections is clustering. One well known interactive clustering technique is Scatter/Gather [20]. This technique modifies standard clustering algorithms to support interactive re-clustering and iterative refinement of document sets. A similar approach is Visual Relevance Analysis [29] which clusters documents and presents an interactive treemap visualization of topics. These types of clustering techniques are compatible with the visualizations used in our work. They can help the reader find documents and understand the structure of the collection but they do not explicitly address problems in reading.

Another common technique that allows users to interact with their personal document collections is keyword search. One commercial example of this technique is the Google Desktop [4]. This system automatically indexes the text of common document types on your hard drive to support keyword searching. Search results can then be filtered by types such as email, chat logs, files, and cached web pages. A variant on standard keyword search is Stuff I've Seen [22], which orders search results on access time. Both of these approaches are similar to ours in that they do not require the user to manually add documents to the system. Keyword search is also compatible with our techniques and is in fact supported in The UC. However, these techniques again differ from our work in that they do not focus on reading.

Document management systems also have several aspects in common with our work. The Docuworks [3] and PaperPort [10] systems allow the user to work with a variety of common document formats in a 2D workspace. They also provide a consistent external reader for all document types and support spreading out and re-combining document pages. Our work differs in that it focuses on visualizations and interaction techniques for large automatically generated document collections. We also integrate the document reader with the browser application to support a more fluid reading experience.

Several user interface techniques have also been introduced specifically for reading tasks involving web pages. The Web Book and Web Forager [18] uses a 3D book metaphor to organize collections of web pages into a three level hierarchy for navigation and reading. Our work differs in that it provides visualizations of multiple collections simultaneously. It is also not limited to web pages or a three level hierarchy so it supports larger collections of diverse document types. The Data Mountain [33] is another technique for interacting with collections of web pages. It provides a planar 3D

workspace in which the user can manually organize web page thumbnails. In contrast, our techniques provide automatic visualization layouts, overviews of multiple collections, and support for a variety of document types.

A large body of previous work has also been focused on reading individual documents. Adobe Acrobat [1] is the most well known commercial example of a standard document reader. It provides a scrollable page view, zooming, thumbnail overviews, text and image selection, and annotation support. Another notable single document reader is the Document Lens [34], which presents a fisheye view of a grid of page thumbnails. XLibris is yet another reader [32] that primarily focuses on techniques for enhancing a user's natural freeform annotations on the document. The multivalent browser [31] is a reader that shares a common goal with our work to create a standardized reading and annotation platform for multiple document types. In each of these cases, our work differs in that it focuses on multi-document reading tasks where getting overviews of collections and comparing documents are important operations.

3. EFFECTIVE INTERACTION

3.1 Fluid Reading

Current document browsers and reading tools have overhead that can be cognitively disruptive for reading. One limitation in these systems is that not all document types are supported by the same reading application. As a result, not all documents support the same functionality, such as thumbnail overviews, highlighting, freehand annotations, text notes, or keyword search. Moreover, even when similar functionality is available in a document's native applications, the user interface controls to that functionality are not standardized.

The lack of a standardized reading environment is particularly problematic on tablet computers. In this environment, a user's natural impulse is often to pick up a document and begin scribbling on it with a pen. For many document types, this is not supported by their native application. Several tablet reading applications, such as the Microsoft Journal [6] or the FranklinCovey TabletPlanner [2], address this problem by allowing the user to print unsupported documents to a format that is supported by the application. However, this extra step can interrupt the user's thoughts about the task at hand and discourage the practice.

A related limitation is that document finding and browsing is divorced from document reading. As a result, reading a document typically involves opening a new application external to the one used to find it. This act of opening an application often introduces small delays both for the system to respond and for the user to orient to the change. Moreover, if the new application is displayed in its own window, it can also introduce window management problems. These issues effectively increase the cognitive overhead of reading.

A more subtle effect of the separation between browsing and reading is that the reader is often forced to prematurely commit to opening a document. Applications for finding and browsing typically display a fixed amount of information about the document so the user must make a decision whether to incur the costs described above based on limited information. An ideal reading interface in this sense would provide progressively more details about a document based on the user's level of interest.

3.2 Multi-Document Reading

Because document finding is typically separated from document reading, multiple document reading tasks are particularly difficult. Many applications provide facilities for navigating between multiple documents of the same type, such as tabs in a web browser. However, reader tasks may involve moving between several types of documents such as web pages, word processing documents, presentation slides, etc. This type of task suggests the need for techniques to quickly jump between documents, regardless of type, in small working sets.

A related problem is being able to view the pages of multiple documents at once. This is often accomplished in the physical world by spreading out documents on a desk or table. Reading applications, such as Adobe Acrobat [1], often provide thumbnail facilities for viewing multiple pages of a single document but not for viewing multiple pages of multiple documents. This feature can be important for making comparisons between the pages of several documents [13] or for getting an overview of the contents of a collection of documents [28].

3.3 Finding Documents

Finding documents, even on users' own file systems, is a widely recognized problem (for example, see [4, 12, 24]). One of the most widely accepted solutions to this problem is keyword search. However, keyword search has a number of limitations for finding documents in personal digital libraries. Perhaps the most widely recognized limitation is that it is fundamentally a recall-based user interface. A classic principle in user interface design is to minimize the memory load on the user [37]. This is often referred to informally as supporting recognition rather than recall [26].

Keyword search also has different characteristics for file systems than for web searching. With keyword searching on the web there are often many pages that satisfy a given query. Moreover, even when a page does not match the user's intentions, it may have a link to a document that does match. These both loosen the constraints for searching on the web. In contrast, on the desktop there is often a single specific document or folder that a user wants to find. Many of a user's documents are also not linked together so they often contain no information about neighboring or related documents. The result is that returning a list of matching hits to the user may not address users needs as well as it does for web searching.

A more general problem with keyword search is that a user's vocabulary often does not match the desired document's vocabulary. This means that users may not be able to find certain documents or they may have to experiment with several queries before they find what they want. A number of techniques have been introduced to deal with this problem such as term aliasing or personalizations like those in Haystack [12]. Nevertheless, these are heuristics that are not likely to work in all cases. An even more extreme example of this problem is when the user is not able to formulate a query at all. Some examples include searching for a picture, a specific page layout, or some other visual property of a document. In some situations, the user may not even have a specific target at all for less directed activities such as browsing.

In these latter cases, the user needs techniques for interacting with documents visually rather than textually. Tra-

ditional file browsers, like Windows Explorer [7] or Konqueror [5], have begun to address this problem to some extent by providing thumbnail renderings and previews for certain document types. However, the user interfaces provided by these file browsers are scrollable workspaces. It is typically not possible to get variable granularity overviews of a large number of documents or documents in multiple folders at once, which restricts visual search.

More generally, the need for searching indicates that users forget where they put things. This problem may be reduced by user interfaces that better exploit human spatial memory [27]. In particular, Lewis *et al.* [24] present evidence that the problem stems in part from a lack of scenery or landmarks in current file browsers and other document management systems.

4. THE UC SYSTEM

This paper introduces a system called UC (originally from “UpLib Client”) that integrates a number of recent user interface, information visualization, and digital library techniques with the goal of addressing the problems described in the foregoing. Although many of the techniques in UC have been employed in other settings, they have often been used in isolation or for limited types of data. We believe that combining these techniques and applying them to a range of common document types can yield a significantly better reading environment.

The system is also designed to work well with pen-based tablet computers where traditional user interface controls, such as scrollbars and text boxes, can be somewhat clumsy. These traditional widgets are still used in some cases but the goal is to reduce the dependency on them.

4.1 Connecting to UpLib

UC is essentially a user interface for interacting with documents in an UpLib [23] repository. As such, it is able to connect to existing UpLib repositories, such as those that have been manually constructed. However, it also provides a separate utility to find documents in the user’s file system and automatically add them to the UpLib repository. This allows a user to quickly populate their UpLib repository with their existing document collections.

4.2 Continuous and quantum treemaps

UC uses continuous and quantum treemap [16] layouts to present collections of documents. Continuous treemaps are space filling visualizations of trees that assign area to tree nodes based on the weighting of the nodes. In continuous treemaps, the aspect ratio of the cells is not constrained even though square cells are often preferred. Quantum treemaps extend this idea by guaranteeing that cell dimensions are an even multiple of a unit size. These layouts are described in more detail by Bederson *et al* [16].

The quantum treemap layouts in UC lead to views similar to those found in the ICDL [21] or the PhotoMesa [14] photo browsers. The page thumbnails used in UC are based on document icons created by the UpLib system. In addition to portraying the often unique appearances of the documents themselves, these icons can also be augmented with overlays such as important text or pictures to further assist users in differentiating one document’s icon from another. These unique document icons can act as scenery to enhance spatial memory of document locations within UCs spatial treemap

layouts.

Users often collect large numbers of documents that are more than can be meaningfully displayed using these thumbnail based visualizations. As a result, the system has a threshold number of documents above which thumbnails are no longer displayed and standard continuous treemaps are shown instead. The continuous treemap attributes of cell size and color can then be used to display quantitative information about the underlying collections such as number of documents, number of pages, file sizes, last modified or last viewed dates, alphabetic ordering, values for manually assigned metadata, or other semantic properties extracted from the documents. Table 1 lists some common operations in UC and their possible starting and ending treemap types. When an operation supports both treemap types, the system chooses between them based on the number of documents in the current view.

There are a number of different algorithms that can be used to layout continuous treemaps. Our system supports both squarified and strip treemap layouts. Because our data is typically ordered by date, file name, etc., the default layout is a strip treemap. Bederson *et al.* present evidence that the strip layout is preferable over other available treemap algorithms for presenting ordered data [16]. This layout also has reasonable behavior as documents are added to collections or the window aspect ratio changes.

One alternative to showing continuous treemaps when the number of documents exceeds the specified threshold is to assign more space to certain documents in a collection. This is an extension of the ideas used in Microsoft Windows Explorer to assign a limited number of images to a folder [7]. A range of criteria could be used to select representative documents including most recently accessed, most globally unique, or most representative of the collection. This remains a topic of research for our future work.

4.3 Navigation

An important aspect of the interface is the fluidity of navigation. This allows the user to focus on the documents rather than on interacting with the tool. In UC, the navigation controls are similar to those in other ZUIs. Left click on an object or group of objects zooms in and either button clicked on the background zooms out.

One problem that arose from combining a zoomable user interface with continuous treemaps involved conflicts with aspect ratios. The cells in continuous treemaps have a range of aspect ratios, as demonstrated in Figure 1. Each of these different aspect ratios may differ from that of the view window. As a result, zooming in on these cells, as you might zoom into a country on a map, may not increase the amount of screen space devoted to a cell. For example, if a cell consumes the full width of the window but only half the window’s height, then there is no way to increase the size of the cell while keeping it entirely on screen.

There are two solutions to this problem. The first is to zoom in so that some of the cell is off the screen. In the previous example, this would mean zooming in to the point where the cell’s height is equal to the window’s height and the cell’s width is twice the window’s width. The advantage of this approach is that it preserves the kind of standard geometric zooming used in many other systems. The disadvantage is that the change in scale needed to zoom a cell is essentially unlimited. In our previous experience with ZUIs,

	Zoom Out	Zoom In	In Context Search	Limit View to Search Results	Explode to Pages	Read Document
C → C	×	×	×	×		
C → Q		×		×		
Q → C	×					
Q → Q	×	×	×	×	×	×

Table 1: The available transitions between the two types of treemaps. In the row header, “C” represents continuous treemaps and “Q” represents quantum treemaps. An “X” in a cell indicates that the column’s operation can start in the first treemap type and end in the second.

we have found that this type of large change in scale can be disorienting even when the transition is smoothly animated.

The second solution, and the one used in UC, is to zoom and morph the cell to the window size and aspect ratio while leaving the rest of the layout in place. The main disadvantage of this technique is that it strays from the standard notion of geometric zooming. This could potentially interfere with the user’s spatial memory by modifying the spatial relationships inherent in the layout. However, the impact on spatial memory is not clear since cells retain their spatial relationships in the static layouts presented at each level. The primary advantage of this approach is that it minimizes the visual disturbance of the display since only a single cell moves. Animating the transition can further help orient the user during the change. Moreover, this technique also mimics other familiar interface actions such as maximizing a window.

The continuous treemap views also provide previews of the layout at the next level down when the user moves the mouse into a cell. One bit of information this preview provides is whether the layout at the next level down is a continuous treemap or a quantum treemap. The preview can also give a rough idea of the number and structure of groups at the next level. A coordinated textual tooltip also displays summary information about the group.

The quantum treemaps in UC are navigated with standard viewpoint animations while the document layouts remain static. This allows the user to build awareness and memories for spatial relationships within smaller working sets of documents.

4.4 Refining and searching collections

If the documents in the UpLib repository were automatically acquired from a file system, then UC presents them as groups based on their containment within the file system hierarchy. If the UpLib was generated manually, then UC uses any manually added category and collection tags to group the documents. Future versions of the software will also allow the user to group objects by other attributes such as original file name, access date, number of annotations, or automatically defined clusters based on document contents. A small extension to the current user interface could also allow the user to select multiple groups for reclustering to support Scatter/Gather [20] functionality.

The system also currently provides controls to refine which documents are displayed. First, the interface provides mechanism to search for specific content within the documents. The search box incrementally highlights matching documents as letters are added to the search query to immediately indi-

cate matching documents. The user can also choose to update the view to re-layout with only documents that match the current query. An example of this type of search, ending in reading a document, is shown in Figure 1.

For UpLib repositories generated from a file system, the interface also provides controls for finding patterns in the file path name and for limiting which file types are displayed. We also plan to generalize this to include controls for limiting file size, number of pages, last access or modification times, and other types of document metadata.

4.5 Explode to pages

For multiple document reading, users need a mechanism to easily make comparisons between multiple documents. This is accomplished in the UC by allowing the user to explode a set of documents into their pages. As with collections of document icons in UC, these page thumbnails are laid out in quantum tree maps. Using an identical layout at both the document and individual page level allows the user interface to provide a consistent set of functionality and interaction techniques between the two types of views. These exploded document layouts not only support comparisons between pages of multiple documents, they can also provide overviews of collections and a mechanism for quickly jumping between pages in multiple documents. Figure 2 shows the layouts in use.

Currently, exploding to pages displays all pages for all documents in the current working set. This behavior could be modified to let the user limit the maximum number of pages shown for each document. This could provide better overviews for document collections that contain extremely long documents. A similar variant on this approach would be to display a subset of the pages based on some semantic criteria such as visual salience. Some other criteria that could be used to choose pages are non-textual elements, unusual text formatting, most frequent visits, most recent visits, relevance to a specified topic, presence of annotations etc.

Another technique to reduce the number of exploded pages displayed is to allow users to choose only the pages they are interested in. For example, this would allow the user to choose all pages in a collection that mention a specific search term. This would be particularly useful for comparing pieces of documents that have multiple sections or topics.

These page layouts could also be used as a document authoring tool. Specifically, the interface could support drag-and-drop between document page layouts to modify page orderings or to create new documents from existing documents. This is particularly useful for authoring presenta-

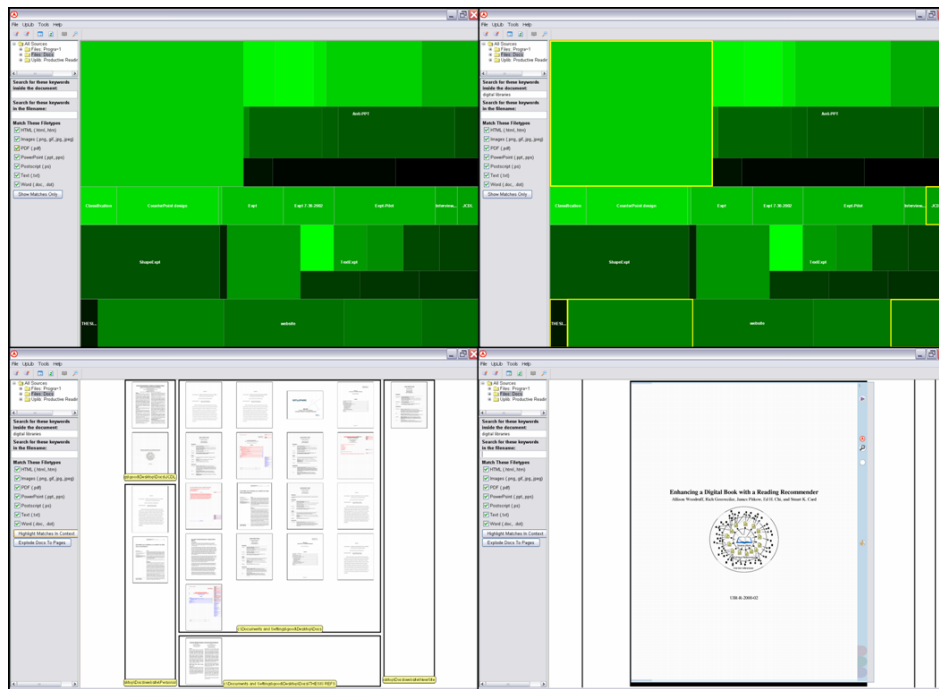


Figure 1: An example keyword search scenario in UC. (a) The scenario begins with a continuous treemap visualization of a document collection. (b) As the user types search terms, interactive highlights appear for groups with matching documents. (c) The user presses a button to limit the view to only matching documents. (d) The user zooms in on a document and begins reading with the ReadUp reader.

tions since in common practice this involves reusing pieces of existing presentations.

4.6 Integrated ReadUp reader

An important principle in the system design is that reading be fluid and not require opening a separate application. This is achieved in UC by integrated the ReadUp document reader provided by the UpLib system. This reader provides thumbnail overviews, freehand pen annotations, highlighting, text sticky notes, bookmarks, and full text keyword search.

Annotations made in the reader are automatically stored in the same UpLib repository that stores the image and text projections. This allows the user to fluidly read and annotate documents without having to manage annotated files or explicitly save changes. The user’s activity, such as how long the user viewed a particular page, is also stored in the UpLib repository which can be used to inform the visualizations provided by UC.

5. IMPLEMENTATION

UC is a Java application that connects to one or more UpLib repositories as a network client. Documents in such a repository are available in the form of *projections* onto several standard views, such as page images or text streams. The UpLib server also runs an extensible set of document analysis and processing operations on each document when it is acquired. These operations extract metadata, create various thumbnail versions of each page image, and full-text index the document. The UC system requests information from the server, such as thumbnails, word bounding boxes,

or document metadata, when necessary to visualize or interact with documents in the repository.

5.1 From file system to UpLib

UC also includes a utility to scan a portion of the file system specified by the user. This scanner then adds supported document types that it finds to a specified instance of an UpLib repository. Additional information As in other desktop search tools, this process could also run in the background periodically to update new or modified documents in the file system.

One consequence of UC communicating with UpLib via network protocols is that the UpLib repository is free to reside on a different machine from the one in which the documents themselves reside. This allows users to store all of their annotations for documents on multiple machines on a single central server, for example. An additional benefit to this approach is that it allows individual users to add personal annotations to documents on shared file servers since the original documents can remain unmodified.

5.2 Handling multiple document formats

UpLib originally handled a number of image formats including TIFF, JPG, GIF, and PNG as well as some standard document formats such as text, HTML, Postscript, PDF. However, because the goal of our system was to support a wider range of user’s everyday reading materials we also wanted to be able to handle common office document formats such as Microsoft Word and PowerPoint. As a result, a Microsoft document conversion server was added to process these documents as an optional component of UpLib. The Microsoft document conversion service, implemented in

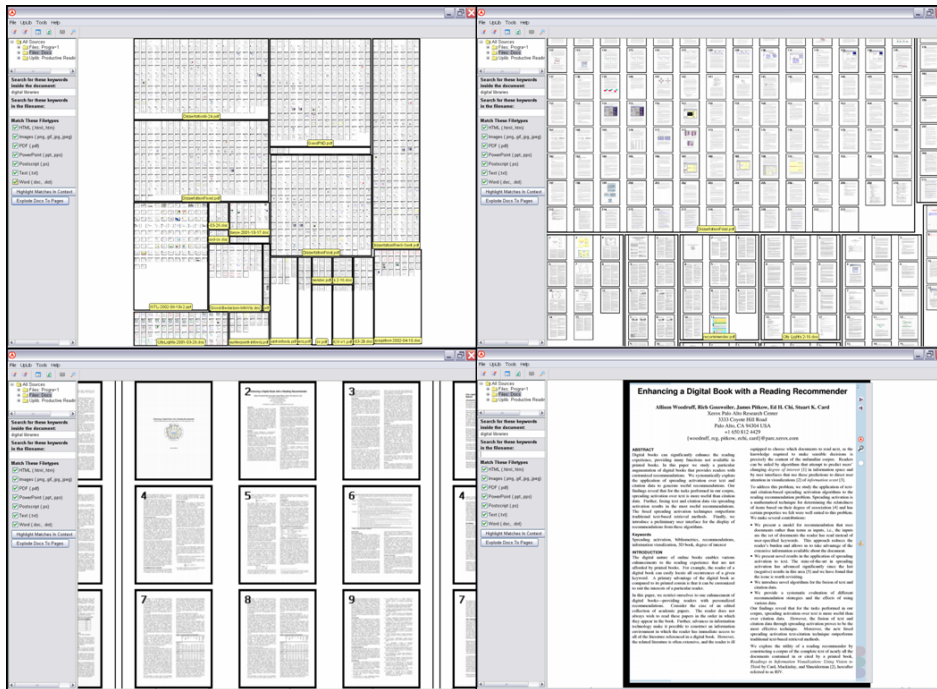


Figure 2: An example of exploding a document collection to pages. (a) A working set of documents is obtained through a search. Pressing a button explodes the documents to pages. (b) The user zooms in on the pages of several interesting documents. (c) The user continues zooming in to a portion of a single document. (d) The user selects a document page and begins reading at that page in ReadUp.

Python, is called by the UpLib document acquisition tool to handle Office format documents. Once the document has been sent to the conversion server it is printed as a PDF file using the open source PDFCreator software [8] for Windows. The PDF file is returned to the UpLib acquisition tool and processed by UpLib as a pseudo-PDF file. UpLib maintains a bit-for-bit copy of the original Office document.

Because the conversion server is implemented as a network service, a single server can be set up to process requests for a number of UpLib repositories. For a Windows machine, the conversion server can also be installed locally to reduce network delays or to support a machine that is not on a network.

5.3 Managing images

The quantum treemap visualizations in the UC system often require large numbers of images. As a result, some care must be taken in loading images to maintain interactivity for the user and a reasonable memory footprint. When a document or page overview is first displayed, such as that shown in Figures 1 and 2, an empty gray rectangle is displayed for each of the image objects. The smallest available page thumbnails are then fetched from the UpLib in a background thread for each of the image objects. When the user zooms in on one of these image objects beyond a specified threshold, a higher resolution page thumbnail is then loaded for the document. The system keeps track of and limits the number of high resolution page thumbnails that are loaded into memory at once in the system.

The ReadUp document reader also manages its own lists of page thumbnails and full page images. ReadUp implements a list of soft references to the page images that can

be reclaimed by the Java garbage collector when necessary. More details on the ReadUp implementation will be provided in an upcoming paper. A future version of the system will integrate the page caching architectures by ReadUp and the other visualizations in the system.

6. FUTURE WORK

There are a number of directions in which the ideas in UC could be extended. One area we want to explore is creating more informative reduced-size representations of documents and collections. The techniques used in the Enhanced Thumbnails [40] are a promising approach for creating more visually unique document icons by overlaying page thumbnails with important keywords. Automatic thumbnail cropping techniques, such as those by Suh et al. [38], are another promising technique that might be extendable to less visual document types. Both of these techniques may also generalize to enhancing document collection representations as well.

UC's user interface currently supports refineable automatic treemap layouts. We would also like to look at combining these automated layouts with user defined layouts such as those in Data Mountain [33] and VKB [36]. The automatic layouts could then bootstrap the users manually constructed working sets for particular tasks.

We also plan to look at how the interface can be adapted for specific kinds of reading tasks, such as legal research. From these specific tasks, we hope to derive a set of user studies to evaluate our tools in both controlled and naturalistic settings.

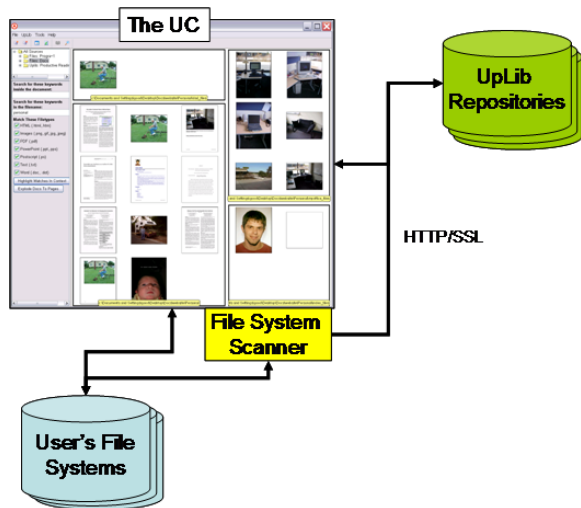


Figure 3: UC acts as a front-end network client for UpLib repositories. UC also provides a utility to scan a user's file system and automatically add documents to an UpLib repository. A single UC instance can scan multiple file systems and connect to multiple UpLibs.

7. CONCLUSION

This paper has presented UC, an advanced visual interface system for navigating within a personal digital library and interacting with its contents in a fluid and consistent manner. The system is based on a zoomable planar representation of the collection in which sub-collections and individual items become accessible as the scale is varied. In contrast to most existing collection visualization and management systems, provision is made for interacting with elements of the collection seamlessly and *in situ*, without having to open a separate application. In contrast to most document reading interfaces, the system facilitates rapid and fluid movement from one document to another document, again without leaving the overarching spatial representation of the collection, thus preserving a sense of orientation and reinforcing spatial memory throughout the course of interaction. In addition, the system streamlines the incorporation of new documents within the personal digital library by automating the conversion of those documents into a common compatible format.

UC combines interactive standard treemaps with zoomable quantum treemap layouts to support visualizations of unlimited sized collections of documents. A document reader provided with the UpLib digital library system, ReadUp, is integrated into the workspace to support rapid transitions between multiple documents without the need to manage an external reading application.

Considered individually, many of the features in UC resemble those found in several restricted domains. The standard treemap visualizations of document collections are similar to those in systems such as SequoiaView [11]. The quantum treemap layouts of image thumbnails resemble those in PhotoMesa [14]. UC's consistent reading and annotation environment mirrors the vision of the multivalent browser [31]. The automatic file system scanning and interactive search capabilities resemble features in desktop search tools such

as Google Desktop [4] or Stuff I've Seen [22]. The exploded page layouts build on ideas from systems such as PowerPoint [9], Docuworks [3], and PaperPort [10]. Yet by combining these diverse functionalities behind the purpose of supporting multi-document reading in personal digital libraries, a qualitative improvement is achieved in the interaction experience.

Finally, we note that this work may be particularly relevant for tablet computers where the user, given the ability to employ a stylus for both inking and command input, will particularly appreciate an interaction experience that can equal or surpass paper in most important respects, including the ability to spread out and pore over a set of documents on a large desk. The techniques presented here enable the user to browse, read, and annotate documents laid out in a large virtual planar arrangement, largely irrespective of originating format, using a consistent set of pen-compatible tools.

8. ACKNOWLEDGMENTS

This research has been funded in part by contract #MDA904-03-C-0404 to Stuart K. Card and Peter Pirolli from the Advanced Research and Development Activity, Novel Intelligence from Massive Data program.

9. REFERENCES

- [1] Adobe acrobat reader. <http://www.adobe.com/products/acrobat/readermain.html>.
- [2] Franklincovey tabletplanner. <http://www.franklincovey.com/tabletplanner/>.
- [3] Fuji xerox docuworks. <http://www.fujixerox.co.jp/soft/docuworks/>.
- [4] Google desktop. <http://desktop.google.com/>.
- [5] Konqueror. <http://www.konqueror.org/>.
- [6] Microsoft journal. <http://www.microsoft.com/tabletpc/>.
- [7] Microsoft windows. <http://www.microsoft.com/windows/>.
- [8] Pdftcreator. <http://sourceforge.net/projects/pdftcreator>.
- [9] Powerpoint. <http://office.microsoft.com/powerpoint/>.
- [10] Scansoft paperport. <http://www.scansoft.com/paperport/>.
- [11] Sequoiaview. <http://www.win.tue.nl/sequoiaview/>.
- [12] ADAR, E., KARGAR, D., AND STEIN, L. A. Haystack: per-user information environments. In *Proceedings of the eighth international conference on Information and knowledge management* (1999), ACM Press, pp. 413–422.
- [13] ADLER, A., GUJAR, A., HARRISON, B. L., O'HARA, K., AND SELLEN, A. A diary study of work-related reading: design implications for digital reading devices. In *CHI '98: Proceedings of the SIGCHI conference on Human factors in computing systems* (1998), ACM Press/Addison-Wesley Publishing Co., pp. 241–248.
- [14] BEDERSON, B. B. Photomesa: a zoomable image browser using quantum treemaps and bubblemaps. In *UIST '01: Proceedings of the 14th annual ACM symposium on User interface software and technology* (2001), ACM Press, pp. 71–80.

- [15] BEDERSON, B. B., AND HOLLAN, J. D. Pad++: a zooming graphical interface for exploring alternate interface physics. In *UIST '94: Proceedings of the 7th annual ACM symposium on User interface software and technology* (1994), ACM Press, pp. 17–26.
- [16] BEDERSON, B. B., SHNEIDERMAN, B., AND WATTENBERG, M. Ordered and quantum treemaps: Making effective use of 2d space to display hierarchies. *ACM Trans. Graph.* 21, 4 (2002), 833–854.
- [17] BIER, E., GOOD, L., POPAT, K., AND NEWBERGER, A. A document corpus browser for in-depth reading. In *JCDL '04: Proceedings of the 4th ACM/IEEE-CS joint conference on Digital libraries* (2004), ACM Press, pp. 87–96.
- [18] CARD, S. K., ROBERTSON, G. G., AND YORK, W. The webbook and the web forager: an information workspace for the world-wide web. In *CHI '96: Proceedings of the SIGCHI conference on Human factors in computing systems* (1996), ACM Press, pp. 111–ff.
- [19] CHANG, M., LEGGETT, J. J., FURUTA, R., KERNE, A., WILLIAMS, J. P., BURNS, S. A., AND BIAS, R. G. Collection understanding. In *JCDL '04: Proceedings of the 4th ACM/IEEE-CS joint conference on Digital libraries* (2004), ACM Press, pp. 334–342.
- [20] CUTTING, D. R., PEDERSEN, J. O., KARGER, D., AND TUKEY, J. W. Scatter/gather: A cluster-based approach to browsing large document collections. In *Proceedings of the Fifteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (1992), pp. 318–329.
- [21] DRUIN, A., BEDERSON, B. B., WEEKS, A., FARBER, A., GROSJEAN, J., GUHA, M. L., HOURCADE, J. P., LEE, J., LIAO, S., REUTER, K., ROSE, A., TAKAYAMA, Y., AND ZHANG, L. The international children's digital library: Description and analysis of first use. *First Monday* 8, 5 (2003).
- [22] DUMAIS, S., CUTRELL, E., CADIZ, J., JANCKE, G., SARIN, R., AND ROBBINS, D. C. Stuff i've seen: a system for personal information retrieval and re-use. In *SIGIR '03: Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval* (2003), ACM Press, pp. 72–79.
- [23] JANSSEN, W. C., AND POPAT, K. Uplib: a universal personal digital library system. In *ACM Symposium on Document Engineering* (2003), pp. 234–242.
- [24] LEWIS, J. P., ROSENHOLTZ, R., FONG, N., AND NEUMANN, U. Visualids: automatic distinctive icons for desktop interfaces. *ACM Trans. Graph.* 23, 3 (2004), 416–423.
- [25] MARSHALL, C. C., PRICE, M. N., GOLOVCHINSKY, G., AND SCHILIT, B. N. Designing e-books for legal research. In *JCDL '01: Proceedings of the 1st ACM/IEEE-CS joint conference on Digital libraries* (2001), ACM Press, pp. 41–48.
- [26] NIELSEN, J. Ten usability heuristics. http://www.useit.com/papers/heuristic/heuristic_list.html.
- [27] O'HARA, K., AND SELLEN, A. A comparison of reading paper and on-line documents. In *CHI '97: Proceedings of the SIGCHI conference on Human factors in computing systems* (1997), ACM Press, pp. 335–342.
- [28] O'HARA, K. P., TAYLOR, A. S., NEWMAN, W. M., AND SELLEN, A. Understanding the materiality of writing from multiple sources. *Int. J. Hum.-Comput. Stud.* 56, 3 (2002), 269–305.
- [29] PEDIOTAKIS, N., AND HASCOËT-ZIZI, M. Visual relevance analysis. In *DL '96: Proceedings of the first ACM international conference on Digital libraries* (1996), ACM Press, pp. 54–62.
- [30] PERUGINI, S., MCDEVITT, K., RICHARDSON, R., PEREZ-QUIÑONES, M., SHEN, R., RAMAKRISHNAN, N., WILLIAMS, C., AND EDWARD A. FOX, . Enhancing usability in citidel: multimodal, multilingual, and interactive visualization interfaces. In *JCDL '04: Proceedings of the 4th ACM/IEEE-CS joint conference on Digital libraries* (2004), ACM Press, pp. 315–324.
- [31] PHELPS, T. A., AND WILENSKY, R. The multivalent browser: a platform for new ideas. In *DocEng '01: Proceedings of the 2001 ACM Symposium on Document engineering* (2001), ACM Press, pp. 58–67.
- [32] PRICE, M. N., SCHILIT, B. N., AND GOLOVCHINSKY, G. Xlibris: the active reading machine. In *CHI '98: CHI 98 conference summary on Human factors in computing systems* (1998), ACM Press, pp. 22–23.
- [33] ROBERTSON, G., CZERWINSKI, M., LARSON, K., ROBBINS, D. C., THIEL, D., AND VAN DANTZICH, M. Data mountain: using spatial memory for document management. In *UIST '98: Proceedings of the 11th annual ACM symposium on User interface software and technology* (1998), ACM Press, pp. 153–162.
- [34] ROBERTSON, G. G., AND MACKINLAY, J. D. The document lens. In *UIST '93: Proceedings of the 6th annual ACM symposium on User interface software and technology* (1993), ACM Press, pp. 101–108.
- [35] SELLEN, A. J., AND HARPER, R. H. *The Myth of the Paperless Office*. MIT Press, 2003.
- [36] SHIPMAN, F. M., HSIEH, H., MOORE, J. M., AND ZACCHI, A. Supporting personal collections across digital libraries in spatial hypertext. In *JCDL '04: Proceedings of the 4th ACM/IEEE-CS joint conference on Digital libraries* (2004), ACM Press, pp. 358–367.
- [37] SHNEIDERMAN, B. *Designing the user interface: strategies for effective human-computer interaction*. Addison-Wesley Longman Publishing Co., Inc., 1998.
- [38] SUH, B., LING, H., BEDERSON, B. B., AND JACOBS, D. W. Automatic thumbnail cropping and its effectiveness. In *UIST '03: Proceedings of the 16th annual ACM symposium on User interface software and technology* (2003), ACM Press, pp. 95–104.
- [39] SUMNER, T., BHUSHAN, S., AHMAD, F., AND GU, Q. Designing a language for creating conceptual browsing interfaces for digital libraries. In *JCDL '03: Proceedings of the 3rd ACM/IEEE-CS joint conference on Digital libraries* (2003), IEEE Computer Society, pp. 258–260.
- [40] WOODRUFF, A., ROSENHOLTZ, R., MORRISON, J. B., FAULRING, A., AND PIROLI, P. A comparison of the use of text summaries, plain thumbnails, and enhanced thumbnails for web search tasks. *J. Am. Soc. Inf. Sci. Technol.* 53, 2 (2002), 172–185.