Document Icons and Page Thumbnails: Issues in Construction of Document Thumbnails for Page−Image Digital Libraries

William C. Janssen

10 July 2004

TR−04−11

parc℠
Palo Alto Research Center

Proceedings of the 8th European Conference on Research and Advanced Technology for Digital Libraries, ECDL 2004, Bath, UK, September 2004. LNCS 3232, pp. 111−121.

Document Icons and Page Thumbnails: Issues in Construction of Document Thumbnails for Page-Image Digital Libraries

William C. Janssen

Palo Alto Research Center
3333 Coyote Hill Road
Palo Alto, California, USA
janssen@parc.com

Abstract. Digital libraries are increasingly based on digital page images, but techniques for constructing usable versions of these page images are largely folklore. This paper documents some issues encountered in creating various kinds of renderings of page images for the UpLib digital library system, and suggests approaches for each, based on both problem analysis and user feedback. Several factors important in determining useful sizes for small visual representations of the documents, called *document icons*, are discussed; one algorithm, called *log-area*, seems most effective.

1 Introduction

The UpLib personal digital library system provides a secure long-term storage and retrieval system for a wide variety of personal documents such as papers, photos, books, clippings, and email. It is suitable for collections comprising tens of thousands of documents, and provides for ease of document entry and access as well as high levels of security and privacy. It is highly extensible through user scripting, and is also intended to be useful as a platform for further research into digital libraries and computer-augmented reading. The general architecture and design of UpLib is more fully described in [6] and [5].

UpLib creates a searchable repository accessed through an active agent via a Web interface. This interface is highly visual, displaying documents as *document icons* (figure 1), laid out in a two-dimensional space, for the user to select from. The built-in reader application also uses thumbnail images of each page (figure 5), in two sizes, for various purposes, but primarily for reading. Other applications, such as the corpus browser discussed in [2], also use the thumbnails in their interfaces. In the process of designing these interfaces for UpLib, a number of issues arose regarding effective generation of document icons and page thumbnails. The rest of this paper presents these issues, and discusses our approach to resolving them.

Fig. 1. Document icons in an UpLib overview. These icons are generated using a *constant-area* algorithm, which allocates the same amount of area to each thumbnail, regardless of orientation. In this interface, clicking on an icon opens that document in a reader program.

2 Document Icons

In the user interfaces of many digital library systems, when a selection of documents is shown to the user, it is presented as lines of text [12], [11]. Sometimes these lines contain document titles, sometimes they contain descriptions of the documents, sometimes they include information such document size or owner or creation date. In many systems, these description lines also include small icons, typically indicating the document format or genre, such as "folder" or "Word document" or "web page". This is also the presentation system used on Windows and MacOS desktops, for Google search results, and for many other multi-document systems. The SOMLib system uses generic icons of books, colored to indicate genre, but otherwise connected to the content of each icon only with textual labels on the icons [8].

In the UpLib system, however, the primary display mode for a selection of documents is as as a set of graphical *document icons*, as shown in figure 1. These icons are not genre icons; rather, they are intended to remind the user of the source document, in terms of appearance, shape, and size. They are small but compelling visual representations of the documents they represent. In constrast to more general-purpose digital library systems, UpLib is intended for personal use by an individual; almost all the documents in any collection have already been seen (and often handled) by the user of the system. This reinforces the ability of visual representations to remind the user of the content of the associated document. The use of document icons also capitalizes on the human perceptual preference for pictures over words when locating an item amidst a number of other similar items [7].

2.1 Computing Document Icon Size

To visually represent the physical document in the digital space, it is usually necessary to scale the size of the document page to a thumbnail representation. It is customary to anti-alias the scaled image, and to preserve the aspect ratio of the image. However, it is less clear what an appropriate size for the icon should be.

In UpLib, page thumbnails are generated for each page of a document (see section 3.1). These are typically constrained to fit in a particular rectangular region, as part of the reading system showin in figure 5. In the first implementation of UpLib, no special document icons were generated; the page thumbnail for the first page of the document was used as its iconic representation. This posed some interesting problems, and led to a series of algorithms for determining icon sizes.

Figure 2 shows seven representative documents arranged in four rows, each row illustrating a particular icon sizing algorithm. From left to right, the documents are a scanned store receipt, actual size 63.8x82.8 mm; an A4 technical paper, actual size 210x297 mm; a US-letter saved Web page, actual size 216x279.4 mm; a Powerpoint presentation, actual size 279.4x216 mm; a map, actual size 355.6x279.4 mm; a scanned newspaper clipping, actual size 99.1x222.7 mm; and a photograph, actual size 162.6x121.9 mm. The top row of the figure shows the effects of the original algorithm, using page thumbnails. While this algorithm worked well for the pages of a single document, all of which were the same size, it does not perform particularly well for a juxtaposed assortment of documents of different sizes. In particular, landscape-oriented documents were shortchanged in the display, when arranged near portrait-oriented documents.

The second row of the figure shows our first attempt at rectifying this problem. A special "document icon" was generated for each document, instead of simply using the thumbnail of the first page. Instead of using a portrait-oriented rectangle to size the icon, we use a square. You will note that the Powerpoint document now receives as much display area as the paper to its left.

However, this algorithm fails to preserve some salient physical differences. Users complained that they were unable to distinguish A4 papers from US-

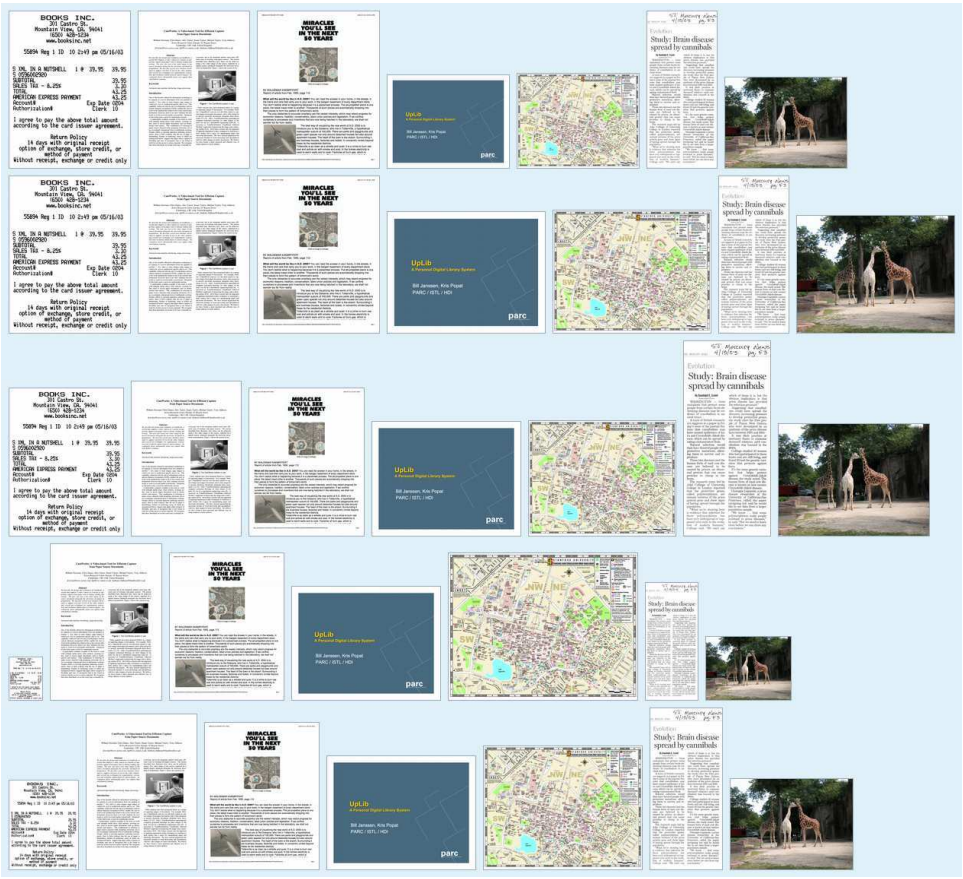

Fig. 2. This figure illustrates five possible icon size generation strategies. At the top is *constant-rectangle*, which attempts to scale each icon to fit in a constant-size rectangle. The second row illustrates *constant-square*, a version of constant-rectangle which provides parity for landscape and portrait mode documents. Both of these suffer from the inability to quickly distinguish US-Letter from A4 documents by their primary distinguishing characteristic, height. This problem is addressed by the *constant-area* algorithm shown on the third row, which allocates the same amount of area to each document icon. However, this algorithm tends to distort the relative sizes of documents. The fourth row illustrates the *linear* algorithm, in which each icon has the same size relative to the others as the physical document would. With this algorithm, large documents such as maps, posters, or blueprints dominate the display. Finally, the *log-area* algorithm in the bottom row provides the same amount of area as the linear algorithm for an US-Letter document, but allocates more area (about four times more) to the small receipt, and somewhat less area to the large map (about two-thirds of the linear algorithm).

Letter papers, even though the A4 document icon shown is somewhat "thinner" than the US-Letter icon. Apparently, the memorable characteristic of the A4 page size is that it is somewhat "taller" or longer than US-Letter; our iconic representation did not preserve that relationship.

The third row of figure 2 illustrates a different algorithm that gives each document icon an equal amount of display area, while preserving the document's aspect ratio. The A4 document can be clearly distinguished from the US-Letter document by height. However, this still masks other relative differences in size. The small sales receipt and small photograph seem to be as large as the US-Letter document. The large map seems to be the same size as the Powerpoint presentation, while the relatively small newspaper clipping towers above the other icons.

To address these issues, we looked for a sizing algorithm that would preserve some elements of the relative sizes of the documents. A strict linear reduction in size, shown in the fourth row of figure 2, would be problematic, as it would tend to make very large icons for very large documents, and vanishingly small ones for very small documents. We decided to use a smooth non-linear function of the area of a document to calculate the icon size. To increase the size of small documents, and reduce the size of very large ones, we chose a function relatively linear around the area of a US-Letter document, which would still reveal small differences, such as those between A4 and US-Letter, but which would still result in smaller icons for smaller documents, and larger ones for larger documents.

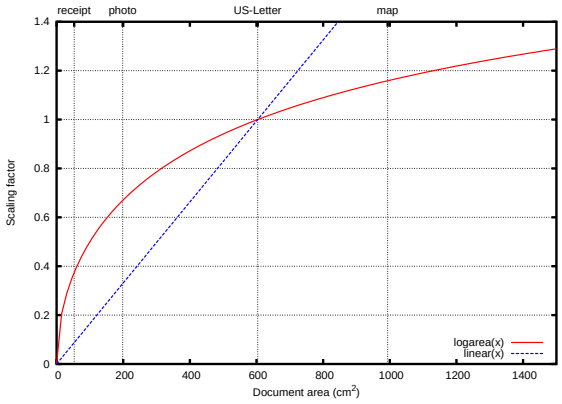

Fig. 3. The *linear* and *log-area* scaling functions.

The graph in figure 3 shows this function, which has the following formula:

$$factor = \sqrt{\ln\left(\frac{area_{document}(e-1)}{area_{US-Letter}} + 1\right)} \tag{1}$$

To calculate the amount of area to allot to each icon, this factor is multiplied by the amount of area that would be given to the icon for a US-Letter document. The bottom row of figure 2 shows the result of sizing the document icons in this manner, which we call *log-area*. While this scaling algorithm seems preferable to the others, all five are available in the current UpLib system, and user-selectable via the configuration mechanism.

2.2 Document Icon Decoration

Enhancement of document icons used in task-oriented applications has been examined by Woodruff et. al., [13]. That application constructed custom document icons to display search engine results to users. Each icon was a rendering of the result Web page, and text items that matched the query terms were exaggerated. This differs from the use of icons in UpLib in several ways: the rendering was only one of many possible renderings of the same Web page, the user may never have encountered that page before, and the text labels exaggerated on the page were specific to that search. However, the general idea of using exaggerated text on document icons to improve recognizability seems a useful one.

Fig. 4. Document icons with labels.

UpLib document icons support colored text labelling, in a deliberately limited fashion. Users can add multiline colored labels to document icons using a single size of a single font (chosen to be unlikely to appear similar to fonts in common use), by defining the "document-icon-legend" metadata element. Figure 4 shows an example of this type of document icon. This is particularly useful for technical papers using small fonts, email messages, newspaper web pages, or any other document that follows a standardized layout so that all documents in that layout appear quite similar.

This approach can also be extended to non-text labels. The rightmost icon in figure 4 shows a mockup of an icon for an email message in an UpLib repository. A picture of the sender has been located in a "biff" library of email senders, and pasted into a whitespace area of the original page image. In addition, a label

has been automatically calculated for the mail message from information in the mail headers.

E-mail raises some questions about the appropriate granularity of documents. Is it a good idea to make each mail message a separate document, or is it more useful to compose a single document from all of the messages that make up a thread in an email discussion? If so, an appropriate document icon may consist of a graph of the thread tree, rather than a picture of the first sender. Similarly, a related series of photographs might be stored as a single document, with concomitant considerations for the document icon. A document similar to the WebBook proposed by Card et. al. [4] could be built from a set of Web pages traversed in a single browsing session; an icon for such a book might be a graph of the session, similar to the "Web behavior graphs" discussed in [3]. The Web caricatures work [14] developed a feature analysis of a Web page, then used that analysis to drive construction of an iconic representation for that page.

Other icon decoration possibilities are possible. The DocuWorks system [1] incorporates cartoon decorations related to the desktop metaphor. Multi-page documents are distinguished from single-page documents with a small binder clip cartoon in the upper left corner. Multiple documents "bound" together are distinguished from individual documents by the addition of the cartoon of a spiral binding on the left side of the document. These decorations typically act as active controls; for example, the binder clip has left and right arrows that allow you to page through the page thumbnails of the document, and clicking the clip itself will fan out the pages of the document.

3 Page Images

The UpLib document reading subsystem makes certain assumptions about the economics of the computing environment. It assumes that disk storage on the order of gigabytes is very cheap (though not free, due to the overhead cost of backups); that the average communications speed is relatively fast, at least 802.11b speeds; and that display screens are fairly tall, at least 1024 pixels in height. A tablet-PC, for example, in portrait mode has a screen that is at least 768 pixels wide and 1024 pixels high. These assumptions are partially due to UpLib's design for personal use: a personal library will have fewer documents than a community library, reducing storage requirements; the document repository will frequently reside on the machine the user is using it on, reducing communication overhead.

As a result of this calculation, the reading subsystem uses page images as the primary presentation form of the document (figure 5). These are anti-aliased reduced-resolution versions of the document's page images, sized to fit on the screen, and optimized for reading. In addition, a small thumbnail version of each page is generated, for use in document overviews. This thumbnail also contains an oversized page number for that page. This section discusses appropriate generation of these two types of document page images.

3.1 Page Thumbnails

Small page thumbnails are used to show an overview of the pages of the document. Examples of this usage are shown in figures 5 and 6. They allow a user to locate graphically distinctive pages, containing diagrams, maps, or photographs, easily. They can also be used to provide some context for the particular page the reader is on. They are sized to be significantly smaller than document icons. This allows more of them to be presented to the reader without scrolling. For most slideshows or technical papers, all of the document's page thumbnails can be presented without scrolling in a display such as that shown in figure 6.

Each page thumbnail is numbered on the left top of the image. In a frame-based display such as that shown in figure 5, this allows the frame to be dragged to the left, occluding the right side of the page thumbnail, but providing more space for the main page image, and still showing the page number on the page thumbnail.

3.2 Large Images

UpLib uses "large thumbnails" in its primary reading interface. These are anti-aliased versions of the original high-resolution page images, scaled to fit on a typical display. The actual display size is user-configurable; by default, each large thumbnail is constrained to fit in a 680x880 pixel rectangle. This allows display on the 768x1024 pixel screen of a tablet-PC in portrait mode. The image format is also important for readability. A text page scaled to size and stored as either PNG (compression level 8) or JPEG (quality 75%) will be about the same size, but the JPEG version will exhibit ghosting effects around the characters, making it harder to read.

Human response times must be considered for Web-based user interfaces. Typically, actions that complete within about 100 msec are seen as instantaneous, and users become impatient if actions do not complete within about 700-1000 msec (see [10], [9]). If we assume a transfer rate of about 2Mbps (a reasonable average for 802.11b), and a maximum allowable user delay of 700 msec, the page image should be no larger than about 170 KB. Allowing for decompression overhead, a top size of about 150 KB is a good target. Faster transfer speeds and user-side caching can be used to increase this limit.

Another concern is the relationship of the presented document size to the original document size. Our original implementation scaled each document to fit within a fixed 680x880 rectangle. This had the unforeseen effect of blowing up small documents, such as the cash register receipt shown in the first column of figure 2, to very large scales, so large that pixelization of the image impaired its readability. In addition, users had difficulty recognizing it for what it was, since they were used to the small size of the physical artifact. To counteract this effect, a sizing governor was used to limit the maximum size for a document. This governor assumes a display resolution of about 100 ppi, and scales documents for display at that resolution, if possible, or for a lower resolution if necessary. This means that the large thumbnail of a document 5cm on a side, scanned at

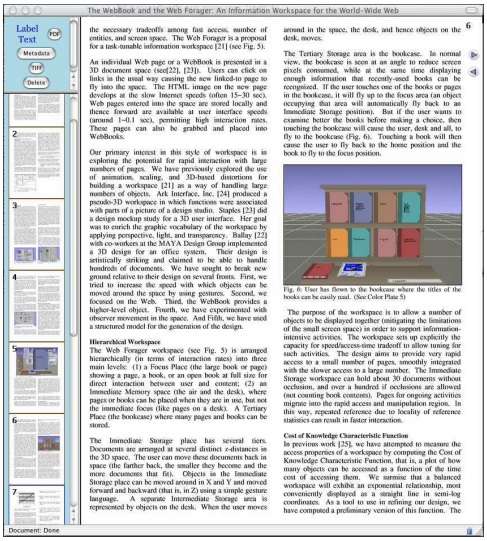

Fig. 5. An "open" document in an UpLib viewer. The small page icons on the left-hand side provide direct access to that page; when used on the 768 × 1024 pixel screen of a Tablet PC, they are partially occluded on the right side, but the page numbers are still visible.

Fig. 6. The small page thumbnails are used for overviews of a document, as shown here and in figure 5. The highlighted thumbnail shows that the current page is page 22. Gross graphical detail on other pages is discernible.

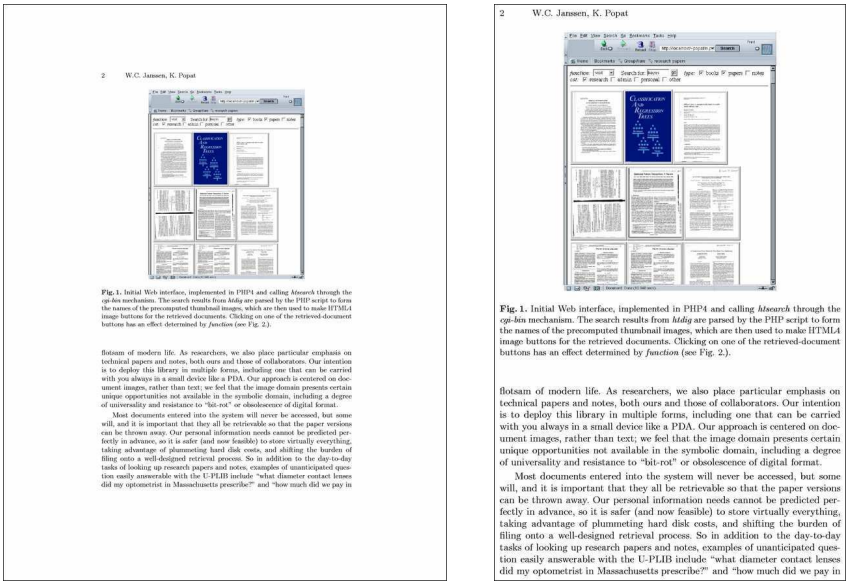

Fig. 7. A page from a paper, before and after whitespace cropping. Both versions are scaled to fit in a 680x880 pixel rectangle, but the one on the right is significantly more readable.

300dpi, would be only about 197 pixels on a side rather than 680 pixels; that is, it would be rendered at about its normal size on a 100 ppi display.

Larger documents are of course reduced more to fit in the bounding rectangle. For a letter-size document, there are about 80 ppi to work with. For 9-point text, this is about 10 pixels per line. With clean text, this is often good enough. However, since ppi is a ratio of total pixels to document size, it is possible to increase the apparent resolution not only by increasing the number of pixels, but also by decreasing the document size. We take the latter approach, by trimming excess background-color space around the text of the document. Figure 7 shows the results of this approach on a document page. It is important, when computing this type of cropping, to compute a single cropping box that will work across all the pages of the document, and apply this box to all pages uniformly. If the cropping box is instead determined and applied on a page-by-page basis, flipping from one page to the next may resize or recenter the text, causing minor perceptual dissonance which can impair reading.

4 Conclusion

With recent increases in storage capacity and network bandwidth, digital library systems based on page images, either scanned or generated, are becoming more popular. It is possible to optimize iconic representations of these page images for visual search and retrieval purposes, using the techniques outlined above. We

recommend the *log-area* algorithm for producing appropriately scaled document icons for selection from a set of documents with heterogeneous sizes. Additionally, scaled page images can be used for document reading on tablet-PC-sized display surfaces, if appropriate attention is paid to size issues. In particular, the technique of removing page borders to increase effective resolution seems to work quite well.

5 Acknowledgements

The UpLib system itself is the result of joint work with Kris Popat at PARC. Many of our colleagues at PARC have contributed generously to our work on this project, notably Eric Bier, Jeff Breidenbach, and Lance Good.

References

1. See http://www.fujixerox.co.jp/soft/docuworks/.
2. E. Bier, L. Good, K. Popat, and A. Newberger. A document corpus browser for in-depth reading. In *Proceedings of the Fourth ACM/IEEE Joint Conference on Digital Libraries (JCDL 2004)*, pages 87–96, June 2004.
3. S. K. Card, P. Pirolli, M. Van Der Wege, J. B. Morrison, R. W. Reeder, P. K. Schraedley, and J. Boshart. Information scent as a driver of web behavior graphs: results of a protocol analysis method for web usability. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 498–505. ACM Press, 2001.
4. S. K. Card, G. G. Robertson, and W. York. The WebBook and the Web Forager: An information workspace for the World-Wide Web. In *Proceedings of the Conference on Human Factors in Computing Systems CHI'96*, 1996.
5. W. C. Janssen. Collaborative extensions for the uplib system. In *Proceedings of the Fourth ACM/IEEE Joint Conference on Digital Libraries (JCDL 2004)*, pages 239–240, June 2004.
6. W. C. Janssen and K. Popat. UpLib: a universal personal digital library system. In *Proceedings of the 2003 ACM symposium on Document Engineering*, pages 234–242. ACM Press, November 2003.
7. A. Paivio and I. Begg. Pictures and words in visual search. *Memory & Cognition*, 2(3):515–521, 1974.
8. A. Rauber and D. Merkl. The SOMLib digital library system. In *Proceedings of the 3rd European Conference on Research and Advanced Technology for Digital Libraries*, number 1696 in Lecture Notes in Computer Science, pages 323–342, September 1999.
9. G. G. Robertson, S. K. Card, and J. D. Mackinlay. The Cognitive Co-Processor architecture for interactive user interfaces. In *Proceedings of the ACM Conference on User Interface Software and Technology (UIST '89)*, pages 10–18. ACM Press, 1989.
10. G. G. Robertson, S. K. Card, and J. D. Mackinlay. Information visualization using 3D interactive animation. *Communications of the ACM*, 36(4):57–71, April 1993.
11. R. Wilensky. Personal libraries: Collection management as a tool for lightweight personal and group document management. Technical Report SDSC TR-2001-9, San Diego Supercomputer Center, 9500 Gilman Drive – La Jolla, CA 92093-0505, 2001.

12. I. H. Witten, R. J. McNab, S. J. Boddie, and D. Bainbridge. Greenstone: A comprehensive open-source digital library software system. In *Proceedings of the Fifth ACM International Conference on Digital Libraries*, 2000.
13. A. Woodruff, A. Faulring, R. Rosenholtz, J. Morrsion, and P. Pirolli. Using thumbnails to search the web. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 198–205. ACM Press, 2001.
14. M. Wynblatt and D. Benson. Web page caricatures: Multimedia summaries for WWW documents. In *Proceedings of the IEEE International Conference on Multimedia Computing and Systems*, June 1998.