A Viewer for PostScript Documents

Adam Ginsburg  
Aiken Computation Lab  
Harvard University  
Cambridge, MA 02138  
+1-617-495-2344  
ginsburg@eecs.harvard.edu

Joe Marks  
MERL  
201 Broadway  
Cambridge, MA 02139  
+1-617-621-7534  
marks@merl.com

Stuart Shieber  
Aiken Computation Lab  
Harvard University  
Cambridge, MA 02138  
+1-617-495-2344  
sieber@eecs.harvard.edu

ABSTRACT
We describe a PostScript viewer that provides navigation and annotation functionality similar to that of paper documents using simple unified user-interface techniques.

KEYWORDS: Document viewing, PostScript, annotation.

INTRODUCTION
Viewing a document on a computer screen is a poor substitute for reading a paper copy of the same document. This is due in large part to the physical limitations of computers and display monitors, but uninspired user-interface design has also been a contributing factor. Our goal in developing DeckView has been to design a PostScript viewer that affords some of the same kinds of functionality provided by paper documents. For instance, paper documents provide sequential paging; contextual location of a page; scanning (by riffling); direct access (using thumb indexes, bookmarks, and fingers); and annotation. Our approach, as manifested in the DeckView document viewer, has been to provide this functionality not by mimicking the devices that provide for it in paper documents (e.g., simulating the appearance of bookmarks or thumb indexes), but rather by designing simple unified user-interface techniques tailored specifically to on-screen documents.

NAVIGATION FEATURES
We provide the variety of navigational functions supported by paper documents — sequential paging, scanning, permanent and temporary marked direct access, contextual location — through a single novel device. Antialiased thumbnail images of the pages are placed in a “tray” (see Figure 1). By overlapping the thumbnails like a deck of cards, all of the document’s pages can be included in the tray, regardless of document length. Depending on the number of pages, more or less of each page will be visible, except for the current page, which is always fully visible. The position of the fully visible thumbnail locates the current page within the document.

Annotation icons are overlaid on top of the PostScript text and remain open.) Target page can be performed while the annotation dialog box is another annotation: it is placed by dragging the hyperlink icon. (Navigation to find the annotation label, annotation text, and hyperlink destination. Any annotation on the page in the page-display window. The dialog box appears (shown in Figure 1) contains fields for an annotation label, annotation text, and hyperlink destination. Any or all of these fields may be specified, allowing for a variety of different annotation types. The destination of a hyperlink is another annotation: it is placed by dragging the hyperlink icon from an open annotation dialog box to the desired location on the target page, or by dragging an existing annotation and dropping it on the hyperlink icon. (Navigation to find the target page can be performed while the annotation dialog box remains open.)

Annotation icons are overlaid on top of the PostScript text and

1 There are other ways in which on-line document viewing might be enhanced: the “Virtual Paper” project at DEC SRC [2] is motivated by the same general goals as ours, but its developers have concentrated on different aspects of the document-viewing task.

2 Note that this aspect of the interaction permits an arbitrary number of pages, regardless of the height of the tray. In contrast, scroll bars require a minimum of one pixel per item, and are difficult to use without much more resolution.
imagery in the page-display window. In Figure 1, a sample icon is shown to the bottom left of the figure in the page-display window. The reader can access the annotation text (if available) by clicking on the “Post-it” portion at the left end of the icon, and can follow a hyperlink (if instantiated) by clicking on the arrow portion at the right end of the icon. Double clicking on the icon label brings up the dialog box again, allowing further editing.

IMPLEMENTATION DETAILS

DeckView is implemented in C++ and Motif. The GNU Ghostscript interpreter is used to render document images. Documents must be formatted according to the Adobe Document Structuring Conventions (DSC) [1] to enable the navigation and annotation features, though non-DSC PostScript documents are supported with limited functionality.

Permanent-bookmark and annotation data are saved as comments in the PostScript document, so the integrity of the document is not compromised in any way by the addition of bookmarks and annotations, and the marked document can be shared as a single file.

Source code and binaries for DeckView are available under a free license at the following URL: “http://www.eecs.harvard.edu/~shieber/DeckView/”.

ACKNOWLEDGMENTS

Mary Farbood assisted with system testing. This material is based upon work supported in part by the National Science Foundation under Grant No. IRI-9350192.

REFERENCES

