An Experimental System for Creating and Presenting Interactive Graphical Documents

STEVEN FEINER, SANDOR NAGY, and ANDRIES VAN DAM
Brown University

An experimental system is described for the design, development, and presentation of computer-based documents that combine pictures and text on a high-resolution color raster display. Such documents can be used, for example, for maintenance and repair tasks, videotex databases, or computer-aided instruction.

Documents are directed graphs whose nodes we refer to as *pages*, in analogy to the pages of a paper book. A page includes a set of simultaneously displayed pictures, actions (procedures and processes), and indexing information. Pages may be nested arbitrarily deeply in *chapters* that serve much the same organizing function as those of conventional books.

The system is comprised of separate programs for laying out and drawing pictures, for graphically specifying the contents of pages, chapters, and their interconnections, and for displaying the document for user interaction.

Examples are given from a prototype maintenance and repair manual in which emphasis was placed on designing actions that allow simple real-time animation and assist in finding one's way around the document.

Categories and Subject Descriptors: H.4.2 [**Information Systems Applications**]: Types of Systems; I.3.6 [**Computer Graphics**]: Methodology and Techniques—*interaction techniques*; I.7.2 [**Text Processing**]: Document Preparation; K.3.1 [**Computers and Education**]: Computer Uses in Education—*computer-assisted instruction (CAI)*

General Terms: Algorithms, Human Factors

Additional Key Words and Phrases: maintenance and repair, pictorial information systems

1. INTRODUCTION

Paper documents are inadequate for creating, storing, and accessing much of today's information. Their limitations and the problems they create are well exemplified in documentation for maintenance and repair tasks. Current maintenance and repair manuals are bulky where portability is essential. They are difficult to keep up-to-date where timeliness is crucial. They provide static, predominantly textual displays where dynamic, pictorial presentations of complex information are becoming increasingly necessary. The same technology that is

An earlier version of this paper appeared in *Computer Graphics*, vol. 15, no. 3, 1981, ACM.
This work was supported in part by the Office of Naval Research under Contract N00014-78-C-0396 and the National Science Foundation under Grant INT-7302268-A03.
Authors' present addresses: S. Feiner and A. van Dam, Department of Computer Science, Box 1910, Brown University, Providence, RI 02912; S. Nagy, Computer and Automation Institute, Hungarian Academy of Sciences, Budapest, Hungary.

responsible for today's information explosion, however, can also provide the ability to manage it. We believe that well-crafted computer-based documents can retain the useful aspects of paper documents while providing sophisticated information retrieval facilities and adaptive presentation capabilities beyond those of paper.

We are developing a database model and support software for research in computer-based documents. Our *documents* are directed graphs whose nodes we refer to as *pages*, in analogy to the pages of a conventional book.

A page includes color pictures containing both text and illustrations, actions (procedures and processes) triggered when the page is accessed or when pickable picture elements in it are selected, and indexing information, such as keywords, that may be used to satisfy retrieval requests. Thus each page is not merely a passive frame containing text and graphics, but has associated actions that may, for example, run animations, access new pages, or even run other programs. The directed-graph structure arises from actions associated with a page that cause other pages to be accessed when they are invoked. The set of actions that may be invoked from a document is conceptually part of the document itself—different documents may have different sets of actions. Pages may be placed in arbitrarily deeply nested *chapters* that serve much the same organizing function as those of paper books. The document itself is treated as a chapter containing those highest level chapters and pages.

The choice of a directed-graph structure is an outgrowth of our earlier work on the Hypertext Editing System [3] and its successor, FRESS (File Retrieval and Editing System) [19], both of which are text processing systems with powerful information structuring and retrieval capabilities. FRESS, like NLS [5], has cross-reference facilities that allow any part of a document to reference or actually access any other part of it or another document. It has been used in large document preparation [16], including the preparation of a richly interconnected, multiauthor "communal text" [4]. Our experience with FRESS and the success of network databases in applications such as computer-aided instruction (e.g., PLATO [1]) have convinced us of the need for such generality. In systems such as FRESS and PLATO, however, authors often make paper sketches of network diagrams and then translate the sketched structures into appropriate commands. The author cannot view or manipulate the structure graphically. We wanted to construct a graphical authoring interface of the kind used to display tree-structured databases in the database management system SDMS [2, 9], but with provisions for easily manipulating rich directed-graph structure. It was important to us that the natural way of expressing graph structure—as a directed graph—become the actual means by which our authors structured their documents.

Our system is comprised of a set of programs all of which use common graphics support software that drives a high-resolution color display. Each program corresponds to a phase in the nonsequential process of creating, modifying, and presenting a document.

Picture layout is the creation of individual pictures.

Document layout consists of page and chapter layout. Page layout involves combining pictures (often rough sketches that serve as "place holders") on a page

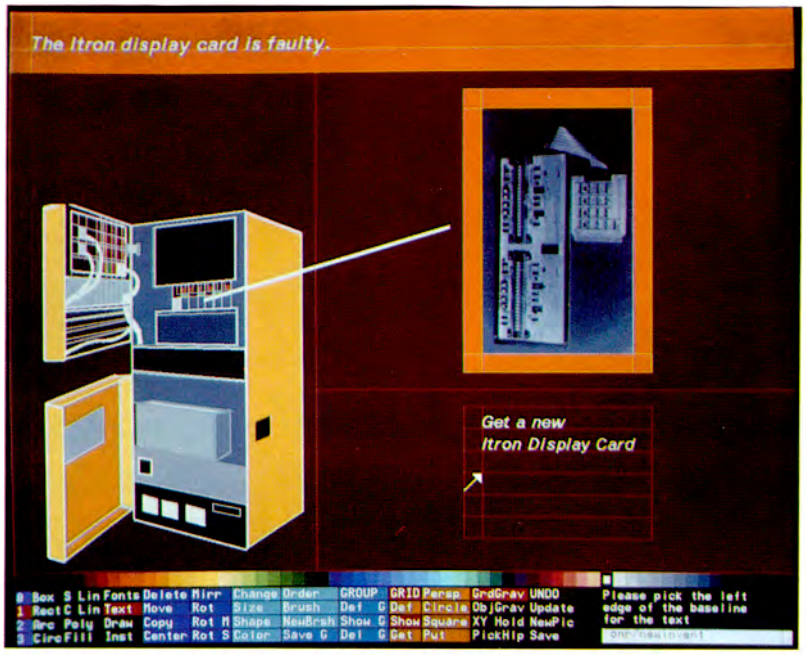

Fig. 1. The picture layout system. Here a set of grids is being used to lay out a picture. The picture of the machine at the left is a scaled-down instance of a separate picture; the circuit board at the right is a bit-mapped image.

and associating actions with both the page and its components. Chapter layout is the creation of chapter hierarchy and the imposition of high-level interconnection structure.

Document presentation is the interaction of the user with the document. User-selected actions are executed for the current page, and the document is traversed as actions access new pages.

We assume that the creation and modification of the database, ranging from isolated pictures to a completed document, are accomplished by skilled professionals: designers, illustrators, authors. On the other hand, the complexity of the pages and actions that are designed will vary from document to document according to the interests and abilities of a particular document's intended users. To avoid confusion, we refer to the users of the picture layout, document layout, and document presentation systems as designers, authors, and readers, respectively.

2. PICTURE LAYOUT

This menu-driven program allows a designer to draw and position line and solid geometric graphic primitives, bitmaps, and text (Figure 1).

The top section of the screen is the picture space. The bottom section of the screen contains a palette of pickable color chips and several rows of color-coded

function buttons. A small area is reserved for prompts and messages. Objects are drawn or positioned by using a data tablet and stylus to select buttons and points in the picture space. When alphanumeric input is required (e.g., for text), a simple screen editor is used.

In our current system we have allocated six out of the available eight frame buffer planes to the picture itself, allowing 64 user-selected colors. The remaining two planes are used for the menu, system-controlled highlighting, and grids (described below). Each picture has its own color palette. By convention, however, all pictures within a single document have identical palettes to allow an arbitrary set of pictures to be displayed on the same page in their correct colors.

2.1 Input Primitives

The designer can draw straight and freehand lines in a variety of widths and styles, as well as arcs, circles, and ellipses. Solid rectangles, general polygons, and an area-fill facility are provided. Existing pictures may be instanced with arbitrary scaling and rotation, and bitmap images may be included (typically scanned in with a video camera). Single lines of text may be set in a variety of fonts and point sizes. In addition, a simple typesetting capability allows the designer to format TROFF [11] files in a designated rectangular section of the picture. All objects possess "gravity" that can be used to position them precisely in relation to each other. When object gravity is enabled by the designer, it constrains any point selected with the data tablet within a small tolerance of an object's perimeter to lie on the perimeter itself. Some objects have gravity at additional places, such as the baseline of a line of text.

2.2 Editing Functions

Any object may be deleted, moved, copied, mirrored, centered between other objects, rotated, altered in size with or without regard to aspect ratio, and recolored. Certain objects may be modified in other ways. For example, the font or text string of a piece of text may be altered. Objects can be grouped by specifying a polygonal extent, allowing all enclosed objects to be edited with a single operation. An undo stack is maintained for reverting any modifications made during a session.

2.3 Grids

A grid system allows the designer to create grids that can be used both for layout placement and as an aid in perspective drawing. One kind of grid is a rectangle that may be divided independently along its length and width in proportions or absolute units (e.g., inches, picas, points) specified by the designer. For example, a horizontal specification of

$$2 : 0.5\ \text{in} : 1$$

may be used to divide a rectangular grid into two columns, the left one twice as wide as the right, separated by a 0.5-inch gutter. Repetition factors allow regularly ruled grids to be easily specified. Other grids are circular (divided by proportionally spaced radii and nested concentric circles) and perspective (trapezoids

specified by vanishing point). Any number of possibly overlapping grids may be used simultaneously.

The lines and intersections of grids possess gravity that may be turned on or off independently of the gravity possessed by objects. Although grids are not part of the picture being created, they may be operated on by the same object manipulation commands used for editing picture elements. The set of grids is named, saved, and retrieved separately from the picture, so that a library of standard grids may be maintained. Grids are displayed in their own plane as an overlay whose colors are a function of the colors of the objects they intersect to improve legibility—on top of dark colors grid lines are lighter, on top of light colors they are darker.

3. DOCUMENT LAYOUT

It was important to us that pictures created with the picture layout system be usable (for slide shows, displays for other programs, etc.) without the rest of the document system. We therefore decided to isolate from the picture layout system those parts of the document system that dealt with the directed-graph interconnections and chapter structure. Another goal of the project was to impose as little design methodology as possible on the system's users so that neither a top-down nor a bottom-up document building strategy was enforced. As a consequence, the document layout system allows the author to alternate freely between working on a chapter and working on a page. The author may even interconnect empty pages (which are created as they are specified) or, conversely, specify the contents of a page which has yet to be linked into the document. Pictorial representations of pages and chapters are displayed in an editing window. The author's graphical editing commands result in changes to the document structure, which is maintained using an experimental relational database system [12]. As with the picture layout system, any editing mistakes may be easily "undone."

3.1 Page Layout

The page layout facilities allow the author to position on a page one or more pictures, containing illustrations and text, made with the picture layout system (Figure 2). Picture elements or rectangular areas may be designated as *buttons* that will invoke some specified action when touched by the reader. The author couples actions to buttons by menu picking.

As a convenient shorthand, the author may indicate that whenever some picture is instanced in any other picture, it is to be considered a button with a specified default action. This feature helps reduce the tedium that would otherwise be associated with making standardized buttons on a page. If many pages of a document are to contain the same set of buttons, a *button board* picture may be created containing an instance of a special picture for each button desired. This one button board picture need only be included in a page to obtain all of its buttons. As any object, including a picture, may be made into a button and buttons may be placed anywhere on a page, page design can be determined entirely by designers and authors.

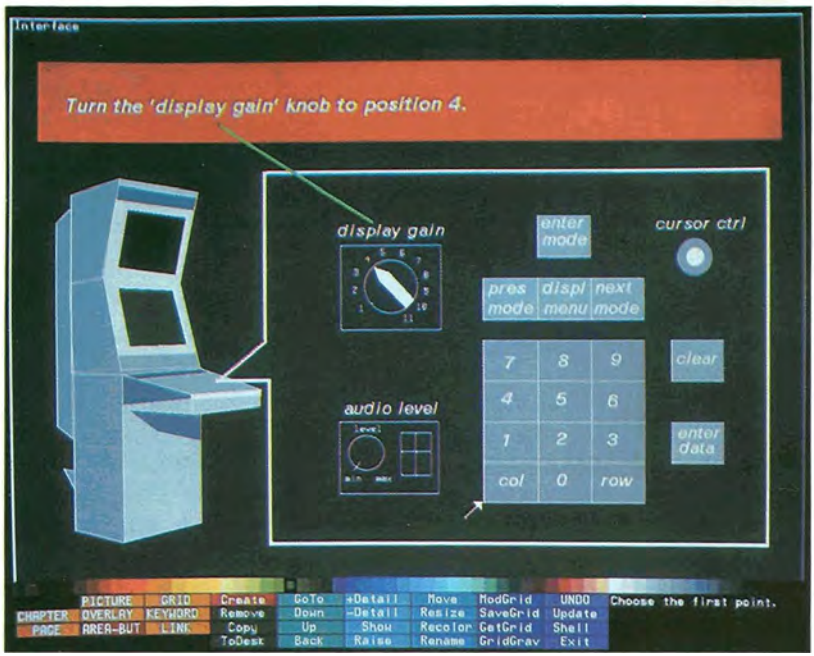

Fig. 2. The page-editing facilities of the document layout system. A page is being composed from already existing pictures created using the picture layout system. The author is making a button by specifying its position on the screen. The pictures will then be dimmed and a menu of available actions overlaid in white.

Actions may be attached to the page itself; they are executed when the page is accessed. (A similar facility is exploited by ZOG [13] and SDMS [2].) Keywords may be associated with pages for keyword retrieval. Overlay pictures, whose colors are automatically derived by interpolating between an overlay color and the colors of regular pictures, may be specified. Editing facilities allow pictures to be redefined, buttons to be associated with new objects, etc.

3.2 Chapter Layout

The chapter layout facilities enable the author to manipulate a chapter's structure (Figure 3). The editing window displays a single chapter that may contain an arbitrary number of subchapters and pages, all shown as named solid rectangles. The color of each rectangle is specified by the author. Each subchapter's rectangle recursively contains the rectangles for those pages and chapters nested inside it. Each page's rectangle contains miniature instances of its pictures. The document is itself considered to be a chapter enclosing its highest level chapters and pages. The author can set the level of nesting to be displayed, to avoid being overwhelmed with unwanted detail. In addition, the amount of information hiding done may be fine-tuned by selecting any unelaborated chapter or page and asking that its internals be displayed, or by turning off the detail of a chapter or page.

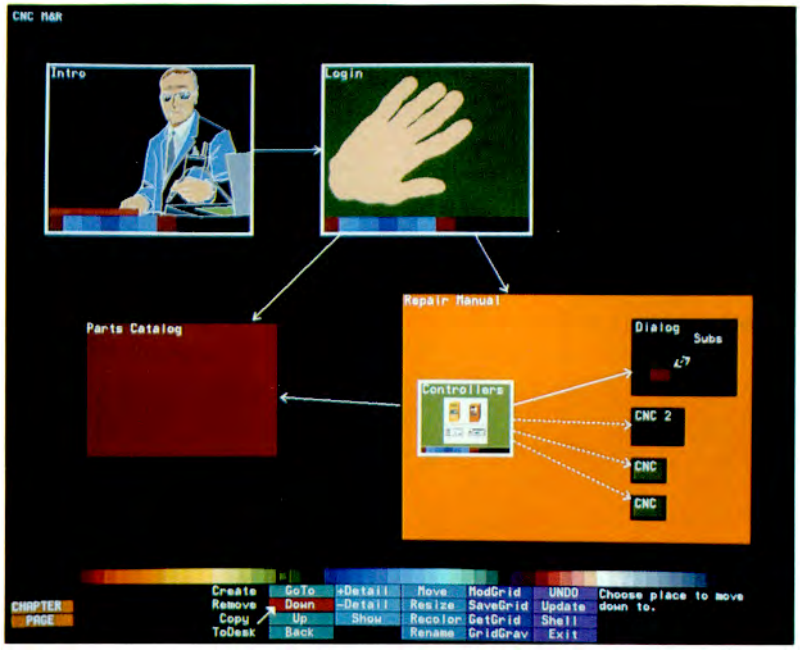

Fig. 3. The chapter-editing facilities of the document layout system. A small set of black-bordered chapters and white-bordered pages is shown nested inside their parent chapter which takes up the top part of the screen above the buttons and palette. Dashed arrows are fuzzy links that indicate that there will be a way (as yet unspecified) of getting from the source to the destination of the arrow; solid arrows are fully specified links.

The size and location of chapters and pages in the display are determined by the author, who positions rectangles in much the same way as the designer does with the picture layout system. The author selects two points on the screen with the data tablet (perhaps within a nested subchapter), and a page or chapter, as appropriate, is created. Chapter and page rectangles may not overlap, however, a restriction imposed to minimize the confusion of "hidden" pages and chapters. The page layout facilities may also be used to edit any of the displayed pages in place. Positioning is aided by the same grid facilities used by the picture layout system.

The author may traverse the document's structure by designating a chapter or page in the window, the current chapter's parent, or any arbitrary chapter or page, which is expanded to fill the window for further editing.

3.3 Links

The directed-graph structure of a document is created by connecting its pages together with links. A link has a source and a destination, and indicates that there is some action that the reader can execute in the source page that will access the destination page. The link is created when the author makes a button

or specifies some page-associated action that accesses a page. It appears as a solid arrow whose tail emanates from its source rectangle and whose head terminates at its destination rectangle. If there is also a link in the opposite direction, then the arrow is bidirectional, with arrowheads at both ends.

As an aid in designing a document, the author may create "fuzzy links" that serve as high-level "placeholders" for actual links to be made later. Fuzzy links may be made by pointing to source and destination rectangles, each of which may be a chapter or page. A fuzzy link between two pages indicates that some as yet unspecified action in the source page will access the destination page. When the source or destination is a chapter, this means that the chapter contains a page (perhaps several levels down) that will be the actual source or destination.

In defining a fuzzy link the author does not have to specify what object is to be made a button or exactly what action is to be associated with it (hence our use of the term "fuzzy"). Both source and destination might even be completely empty at the time the fuzzy link is created. When the author later edits a chapter or page associated with a fuzzy link, he or she will be prompted to resolve it by specifying an actual button and action. Fuzzy links are drawn as dashed arrows. When the fuzzy link is resolved, its arrow is solidified. A consistency check may be run to point out unreachable pages or fuzzy links that have yet to be resolved.

Since any page may potentially be connected to any other page, a confusion of arrows could result. Therefore arrows are drawn only between those pages and chapters that share the same immediate parent chapter. Chapters inherit the links of their progeny. Consequently, when two linked pages are in different chapters, some arrow (possibly at a higher level) in the document display will reflect the linkage. The author can also request more information about a link or cause the links associated with a page or chapter to be highlighted.

3.4 Windows

Although the preceding discussion refers to a single editing window, the document layout system actually allows multiple windows, each of which can contain the pictorial representation of a selected chapter or page (Figure 4). Windows have a constant aspect ratio and may be scaled and positioned arbitrarily on the screen, though they may not overlap. The author may traverse the structure of the document independently in each window. In addition, links may be made between pages and chapters displayed in different windows. The multiple window display makes it possible for the author to connect two chapters or pages at different levels in widely separated parts of the document, or to work in different chapters simultaneously. A change (such as deleting a page) made to a window is reflected in all windows that display the altered object, assuring consistency between windows. The visibility of features such as a page's pictures or a chapter's subchapters is determined on a per-window basis, however. Thus, each window provides a user-specified view of some part of the database. Typically, one window is used to maintain an overview of the document, while other windows show chapters and pages of interest.

Although all operations may be performed in the multiple window display, the author may elect to concentrate on any single window, which can be automatically

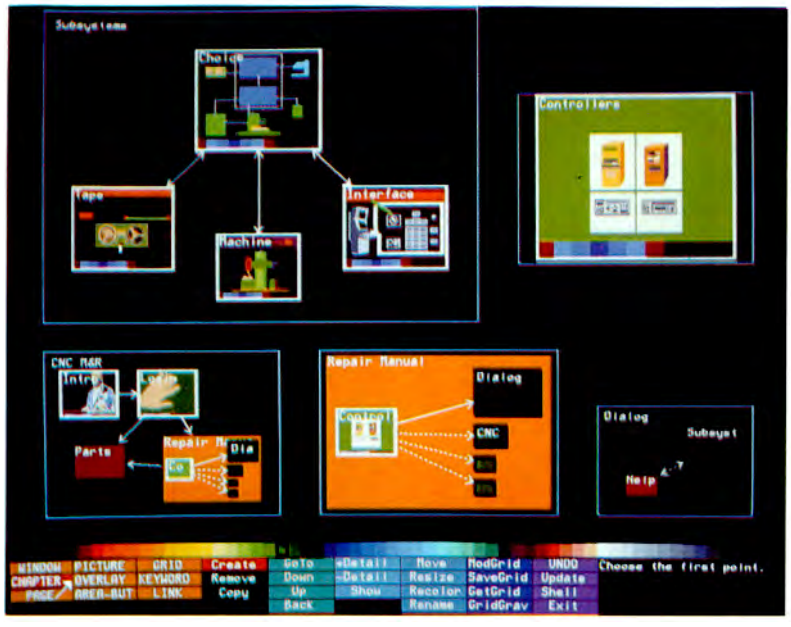

Fig. 4. A multiple window display in the document layout system. Each window contains a chapter or page. The author may traverse the document's structure independently in each window and make links between chapters and pages in different windows. Windows may be created, resized, and removed as needed.

enlarged to become the only window on the screen. The multiple window configuration is then saved and the author may return to it when desired. The window configuration is also maintained between sessions.

4. DOCUMENT PRESENTATION

The document's reader traverses the database, usually by touching buttons that cause new pages to be accessed. The display program traverses the directed graph using the following simple algorithm:

```
access starting page
loop
   wait for reader interaction
   execute associated action
endloop
```

The actions included in a document are logically a part of the document itself and not a part of the display system. Different documents can provide capabilities as varied as their intended audiences. In our testbed system we have sidestepped the issue of allowing the nonprogrammer users of the document layout system to create their own actions. We have instead opted for actions that are coded to order by application programmers and provided as "black boxes" to authors, who need to have no understanding of their internals.

Actions have access to the modeling and database systems. Thus, actions may be created that automatically fashion an entire page, such as the "timeline" display discussed below.

5. SAMPLE DOCUMENTS

To aid in developing our document presentation system, we have been designing prototype computer-based repair manuals that could deliver the technical documentation and assistance needed to maintain and repair complicated equipment.

We envision a portable maintenance aid for the 1990s in the form of an attaché-case-sized device containing a high-resolution color screen for the display of pictures and text, a touch-sensitive overlay, voice input/output, powerful computers, and a mass storage device [8]. Sophisticated repair and test equipment would be built into (or directly interfaced to) the system. Thus the technician's role would be minimized, whenever possible, to mechanically attaching test leads to equipment and operating automated repair equipment. The system would decide what needed to be done through a combination of predetermined, proceduralized troubleshooting sequences and a generalized reasoning component, developed with artificial intelligence techniques, that understood the device being serviced. In addition, a rich set of browsing facilities would allow an advanced user to access any part of the document freely.

A prototype document for servicing computerized numerical control equipment was developed first. This application represents well in microcosm the large, complex databases that we want to handle. Sample pages are shown in Figures 5 and 6. In this first document, no attempt was made to implement actions that would embody actual real-time decision making or interface with external equipment. Instead, we concentrated on developing a small set of standard actions based on data tablet interaction alone to provide a simple, easy-to-use interface. These actions are among those being used in other documents currently being designed.

5.1 Basic Actions

5.1.1 *New Page.* The current page is replaced with the named page. *New page* is typically attached to picture elements that are to be made buttons that invoke other pages. It is the "glue" most often used to connect the database.

5.1.2 *Back Page.* A specified number of pages is popped from the stack of pages accessed during the session and the current page is replaced with that last popped.

5.1.3 *Animation.* A named program in a simple diagrammatic animation language [6] is interpretively executed. The modest speed with which our display can be written prohibits the use of full animation. We therefore rely on color table manipulation [14], panning and zooming about the frame buffer, and the limited animation that can be accomplished by erasing and redrawing individual objects. The color table techniques used include cycling selected portions of the color table and smoothly interpolating a color between other selected colors. Object manipulation is accomplished by scaling, translating, and rotating named objects in pictures.

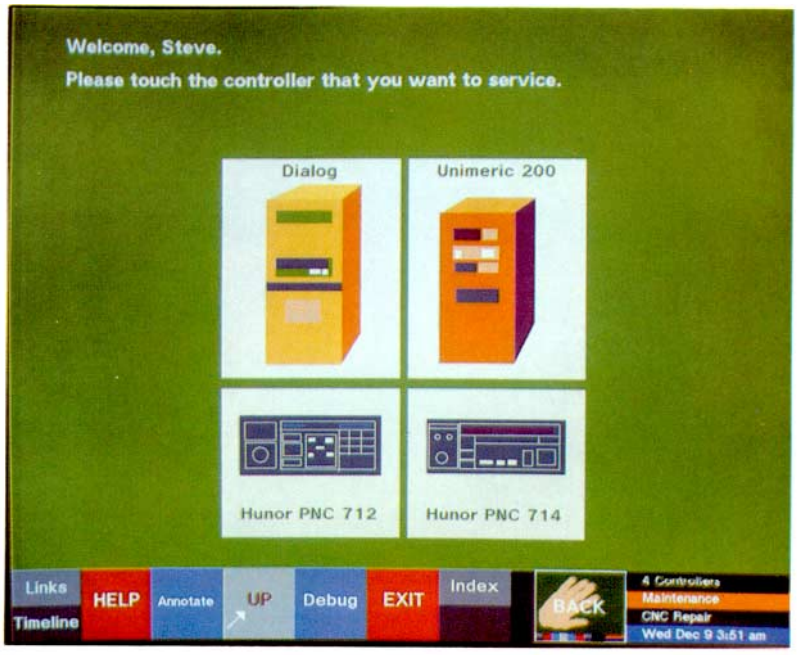

Fig. 5. A page from a document currently under development. The reader's selection of one of the four numerical controllers shown determines the page to be displayed next. All buttons (the controllers and the named rectangles along the bottom) were specified by the page's author. At the lower right, an automatically generated picture shows a miniature of the previous page (which serves as a back button), the names of the page, parent chapter and document, and the time.

5.1.4 *Overlays.* A set of pictures is displayed in its own separate plane as a tinted overlay. Its colors are derived by interpolating between the overlay color and the colors of the picture beneath, providing an effect similar to that of a colored transparent overlay.

5.1.5 *Annotation.* Each page is associated with a unique annotation picture. This action displays the annotation picture (if it exists) and creates a *draw* button. Touching draw creates the annotation picture (if it did not exist) and allows the reader to annotate by drawing with the stylus. The annotation picture is displayed in its own plane and is, in essence, a writable overlay.

5.2 Actions that Display Document Structure

The reader might easily become lost if presented with a computer-based document rich in connective tissue and deprived of the physical feedback of handling a book. We are trying to design a set of actions that will help the reader maintain (or regain) a sense of where she or he is.

5.2.1 *Folio.* A folio provides each page with a standard display of the page name, chapter name, document name, and time at which the page was accessed

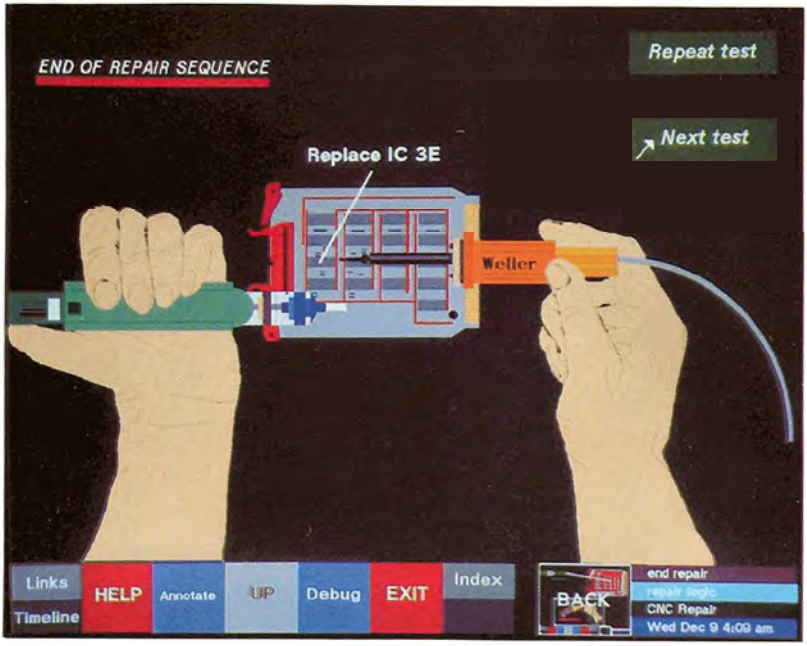

Fig. 6. A later page from the same document. In preceding pages, the fault was isolated to a specific component which the reader has been instructed to replace.

(Figure 5). Each name appears on a field drawn in the color selected in the document layout system for the page, chapter, or document. A miniature of the previously accessed page is automatically created and is labeled as a *back page* button that when touched makes the previous page current. The *folio* action is typically associated with a page's access by the author and produces its display from information contained in the database.

5.2.2 *Timeline.* This action is associated with a special *timeline* page that is empty except for the action. When the page is accessed, the action is executed and completely constructs a set of pictures and buttons on the fly, comprising a pictorial transcript of the current session (Figure 7). The most recently viewed pages are inspected and classified by parent chapter. One horizontal band is drawn for each chapter represented and is colored in the chapter's color. Miniatures of the pages are drawn, each nested in its parent chapter's band, starting with the oldest on the left, with the time at which each was accessed appearing below it. The miniature pages are made into buttons that when touched transport the reader back to the selected page. Buttons permit the reader to move forward and backward along the timeline to examine miniatures of pages viewed earlier.

5.2.3 *Index.* Keywords picked from the list of keywords attached to pages by the document layout program are used to generate an alphabetized display of

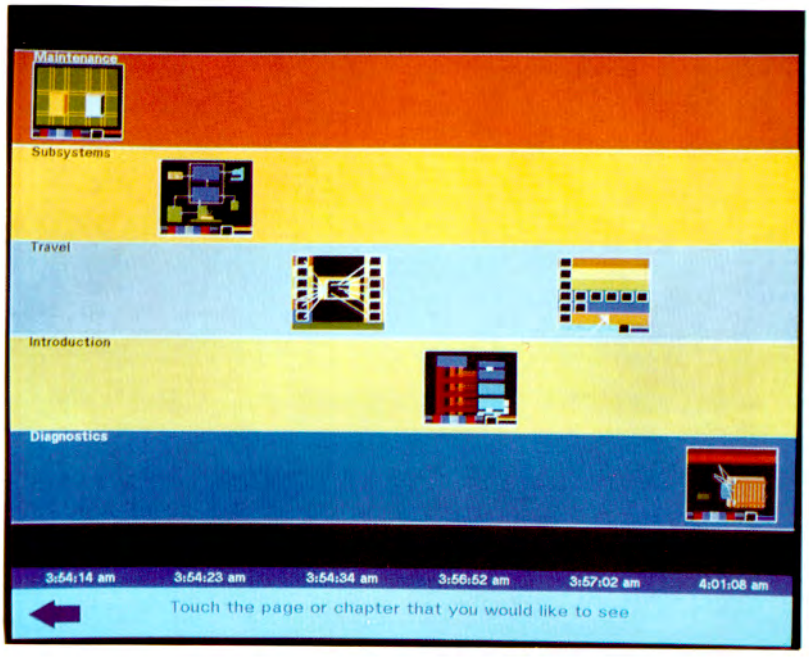

Fig. 7. The timeline page. Miniatures of the most recently viewed pages appear in chronological order. Each is nested in a colored band associated with its parent chapter. Touching a page takes the reader back to that page. The arrow at the bottom scrolls the display so the reader may examine pages that were seen earlier in the session.

miniatures of those pages they match. Each page miniature of this pictorial index is a button that accesses its page (Figure 8).

5.2.4 *Neighbors.* This action creates a display in which a selected page is shown in miniature at the center surrounded by its possible graph predecessors and successors (Figure 9). Those pages that can access the center page are shown in miniature at the left with arrows leading from their associated buttons to the center page; those that can be accessed by the center page are shown at the right with arrows leading to each of them from the center page's buttons. If there are more connected pages than can fit on either side of the display, small up and down arrow buttons appear at the bottom of the display. These may be used to scroll independently through the set of pages on either side.

6. IMPLEMENTATION

Our testbed document system is implemented under UNIX™[1] on a time-shared DEC VAX 11/780. The graphics display is a Ramtek 9400 graphics system with a 1280 × 1024 × 8 frame buffer, affording high image resolution. However, the

[1] UNIX is a trademark of Bell Laboratories.

Fig. 8. The index page. All pages associated with a user-selected keyword are displayed. Colored bands, representing the pages' parent chapters, are alphabetically ordered by chapter name. They contain alphabetized page minatures, each of which accesses its page when touched. If there are more pages than will fit on the screen, then arrows appear at the bottom to allow the reader to scroll over the remainder of the index entry.

belief (inspired by the work of Xerox PARC's Learning Research Group [10]) that more powerful, less expensive hardware can eventually be fit into a package about the size of a book has been a fundamental motivating force behind the project.

Figure 10 shows the interaction of the picture layout, document layout, and document presentation systems with the database system, modeling system, and graphics package that they use.

6.1 Graphics Package and Modeling System

All document programs use a modeling and library system that builds and maintains a picture as a hierarchical data structure. Pictures are displayed through calls to a graphics package [18] based on the proposed CORE graphics standard [7]. Thus, instead of conceptualizing (and storing) each picture as a monolithic bit map, as do many raster graphics "paint" systems [15], our modeling system employs an underlying logical structure in which a picture is broken into components such as lines, polygons, and text, much like typical vector graphics "sketchpad" systems [17]. Although this approach does not offer the same facility

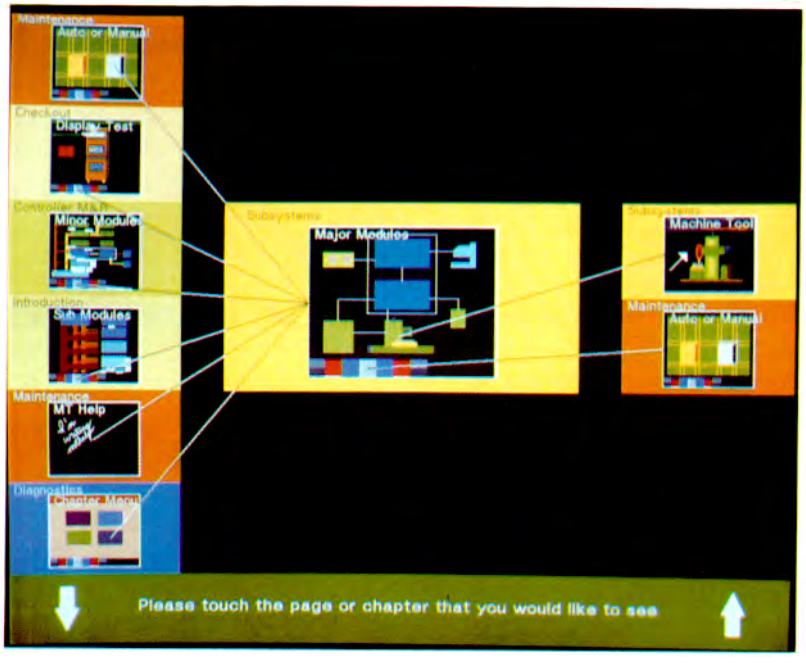

Fig. 9. The neighbors page. The center page is surrounded by miniatures of its predecessors (on the left) and successors (on the right), each of which is nested in a box representing its chapter. Arrows emanate from the center of buttons on the center page and its predecessors and terminate on the pages they can access. Touching a page makes it the current page. Arrows at the bottom will scroll either column to show neighbors that would not fit on the display.

in creating continuous tone pictures as a paint system, it provides several important advantages:

Smoothly scaled instances. Different pictures may make use of smoothly scaled multiple instances of common subpictures. This feature is used extensively, both to decrease the drawing time needed to make complex pictures via instancing and to make possible automatic on-the-fly generation of displays that include different-sized instances of previously created pictures, such as the timeline discussed above.

Greatly decreased storage requirements. We had an immediate need for writable storage for a large number (hundreds to thousands) of pictures. Though future mass storage technologies such as optical digital recording will make inexpensive, fast multiple-gigabyte storage of data a reality, current videodisk storage was not acceptable because of both its read-only nature and its incompatibility with non-NTSC equipment such as our own. On our system a single unencoded full-screen-sized bitmap would require 1.25 mbyte of storage, as compared to the thousands of bytes needed for typical pictures with the present

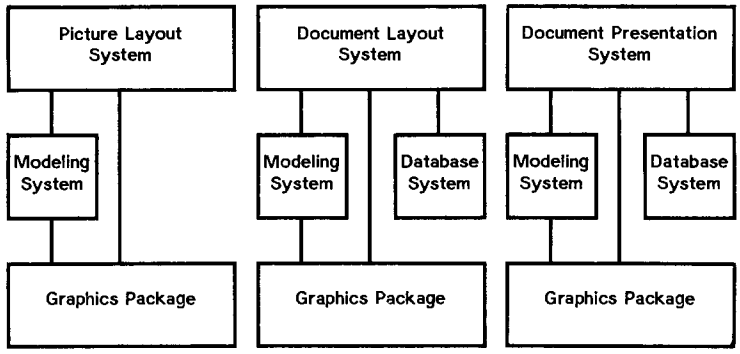

Fig. 10. System implementation. The picture layout, document layout, and document presentation systems use a modeling system and a graphics package. In addition, the document layout and document presentation systems invoke a relational database system that maintains document structure.

approach. One of our prototype documents containing over 300 pictures is smaller than 1.25 mbyte.

Object-oriented picking facilities. The logical structure allows data tablet selection of individual logical entities as well as arbitrary rectangular areas. The object can change location or size without changing identity and can thus remain pickable.

Ease of manipulating individual elements. The ability to name and access individually each of a picture's constituent elements makes it possible to alter attributes such as color, scale, translation, and rotation for animation effects.

In addition to the storage constraints mentioned above, we are governed by severe time constraints on picture refresh. This is an interactive system, designed to be used by a presumably impatient reader. Consequently, no anti-aliasing has been attempted so far, a shortcoming mitigated somewhat by the high resolution of the display. Typical display generation times range from 7 seconds (Figure 5) to 21 seconds (Figure 6), still much too slow for comfortable use. The bottleneck here seems to be drawing time, which is four times that of a comparable 640 × 512 medium-resolution system. For example, the graphics system takes 2 seconds to draw a filled rectangle the size of the screen. Some objects, such as circles, require more time per pixel to scan convert. Other objects which the graphics system does not support directly, such as filled polygons, must first be scan-converted to vectors by the graphics package on the host. More intelligent frame buffers with faster processors and closer host coupling are needed to make systems that build complex pictures on-the-fly feasible.

6.2 Relational Database System

The structure of a document is maintained by a relational database system [12] which runs as a separate process. Each document is represented by a set of relations that include information on pictures, buttons, keywords, page and chapter appearance and nesting, and links. Most of the database operations that

are performed by the document layout or presentation system are invariant between sessions and require only parameter substitution. These retrieval and insertion requests are precompiled by the software and referenced by name by the document layout and presentation systems, allowing increased efficiency.

Consider, for example, what happens when the reader uses the data tablet to select a point on the page. A database retrieval request is issued for a button in the current page whose area contains the selected point or which is associated with the named picked object (if any). The document presentation system executes the button's action if one is found. If it is a new page action, then the database system is requested to compare the designated page's pictures and their positions with those of the current page so that only those pictures not in the new page will be erased and only those pictures not in the old page will be drawn. The name, position, and scale of such pictures are returned and the document presentation system calls the appropriate modeling system routines to draw or erase the pictures by name. The names of any page-associated actions are retrieved next and executed by the document presentation system.

All reader interactions are recorded in a history relation and the status of each page (including, for example, whether an overlay was being displayed) is saved when it is replaced. This makes possible the *back page* action that allows the reader to travel back in time by restoring the status of a previous page.

7. CONCLUSIONS

An experimental system for creating and presenting richly connected computer-based documents has been implemented. In recognition of the difficulty of creating and maintaining complex databases, we have emphasized building powerful graphical authoring tools. By making all actions logically a part of the database rather than of the display program, and by implementing a general directed-graph data structure, we have made possible a variety of documents suited to widely different purposes and audiences.

Our current research is directed toward two areas: automated authoring and 3D modeling.

Automated authoring. It is more difficult to create material that takes advantage of the capabilities of computer-based media than it is to design its paper equivalent. In our current system, all page and chapter layout is the responsibility of the author. Even with sophisticated graphical assists, however, authoring pictorial documents is a demanding and time-consuming task.

Work has begun that will result in a formalization of the knowledge needed to design pages that are more general than, for example, the highly regular "neighbors" display discussed above. Under consideration are pages such as those that provide instructions for removing a particular board from its chassis. Such pages may contain pictures of the board, open cabinet, module extractor, and perhaps even the repair person demonstrating a procedure, combined with textual instructions, explanations, and warnings. Domain-specific knowledge about the design rules for various kinds of generic display formats needed in a document and the objects to be displayed will be combined with information about the contents of a particular page to determine its layout automatically. The desired page would

be classified according to its format and then composed by selecting pictures and bitmap photographs from a library of existing artwork. These would be combined with simple text strings or more complex verbatim text provided by the author, set in fonts selected in accordance with the generic format (style sheet) for the page. Both text and graphics would be scaled and positioned on the page, and appropriate actions would be associated with picture elements. Of particular interest are those situations in which a design rule may have to be stretched or broken to handle a specific page. The long-term goal is to investigate the automatic creation of multiple page structures (sequences) and eventually of entire documents generated a page at a time in response to user interaction.

3D modeling. Each different view of an object depicted in our documents is currently created "from scratch" as a picture or bitmap photograph. By deriving pictures from a 3D model of an object we will be able to create on demand the particular views needed for a page (though not in real time at first because of slow scan-conversion algorithms). This solid modeling capability will allow us to display and interact with 3D projections such as parts explosions. Eventually, we would like to have these parts explosions animated by the automated authoring component.

ACKNOWLEDGMENTS

Past and present members of the Brown University Computer Graphics Group whose participation is gratefully acknowledged are Kurt Fleischer, Joseph Pato, Randy Pausch, Will Poole, Joel Reiser, Adam Seidman, David Salesin, Charlie Tompkins, Barry Trent, Mark Vickers, and Gerry Weil. Imre Kovacs provided creative direction of prototype documents and high-level design of the picture layout system's user interface.

REFERENCES

1. Bitzer, D., and Johnson, R. PLATO: A computer-based system used in the engineering of education. *Proc. IEEE 59,* 6 (June 1971), 960–968.
2. Bolt, R. Spatial Data-Management. Architecture Machine Group, Massachusetts Institute of Technology, Cambridge, Mass. 1979.
3. Carmody, S., Gross, W., Nelson, T., Rice, D., and van Dam, A. A hypertext editing system for the /360. In *Pertinent Concepts in Computer Graphics*, M. Faiman and J. Nievergelt, eds. University of Illinois Press, 1969, pp. 291–330.
4. Catano, J. Poetry and computers: Experimenting with the communal text. *Comput. Hum. 13,* (1979), 269–275.
5. Engelbart, D., Watson, R., and Norton, J. The augmented knowledge workshop. In *Proceedings of AFIPS National Computer Conference*, vol. 42, AFIPS Press, Arlington, Va., 1973, pp. 9–21.
6. Feiner, S. Animal: A Diagrammatic Animation Language. Computer Graphics Group, Brown Univ., Providence, R.I., 1981.
7. GSPC. Status report of the Graphic Standards Planning Committee. *Computer Gr. 13,* 3 (Aug. 1979).
8. Gurwitz, R., Feiner, S., Fleming, R., and van Dam, A. Future Technical Documentation Delivery Systems for Naval Maintenance and Repair. ONR Rep., Computer Graphics Group, Brown Univ., Providence, R.I., 1979.
9. Herot, C., Carling, R., Friedell, M., and Kramlich, D. A prototype spatial data management system. *Computer Gr. 14,* 3 (July 1980), 63–70.

10. KAY, A. Microelectronics and the personal computer. *Scientific American 237,* 3 (Sept. 1977), 230–244.
11. OSSANNA, J. NROFF/TROFF User's Manual. Computing Science Tech. Rep. 54, Bell Laboratories, Murray Hill, N.J., 1976.
12. REISS, S. Eris Reference Manual. Dep. of Computer Science, Brown Univ., Providence, R.I., 1981.
13. ROBERTSON, G., MCCRACKEN, D., AND NEWELL, A. The ZOG Approach to Man-Machine Communication. Tech. Rep. CMU-CS-97-148, Dep. of Computer Science, Carnegie-Mellon Univ., Pittsburgh, Pa., Oct. 1979.
14. SHOUP, R. Color table animation. *Computer Gr. 13,* 2 (Aug. 1979), 8–13.
15. SMITH, A. Paint. Tech. Memo 7, Computer Graphics Laboratory, New York Institute of Technology, Old Westbury, N.Y., July 1978.
16. STRANDBERG, J., CHOMSKY, C., SCHOLES, R., AND VAN DAM, A. An Experiment in Computer-Based Education Using Hypertext. Div. of Applied Mathematics and Dep. of English, Brown Univ., Providence, R.I., June 1976.
17. SUTHERLAND, I. SKETCHPAD: A man-machine graphical communication system. *SJCC 1963,* Spartan Books, Baltimore, Md., 1963.
18. TRENT, B. SGP User's Manual. Computer Graphics Group, Brown Univ., Providence, R.I., 1981.
19. VAN DAM, A., AND RICE, D. On-line text editing: A survey. *ACM Computing Surveys 3,* 3 (Sept. 1971), 93–114.

Received August 1981; revised October 1981; accepted November 1981