

Pages Without Paper

by

Christopher Martin Schmandt

Submitted in Partial Fulfillment

of the Requirements for the

Degree of Bachelor of Science

at the

Massachusetts Institute of Technology

December, 1978

Signature of Author .....  
Department of Electrical Engineering and Computer  
Science

Certified by .....  
Nicholas Negroponte, Associate Professor of Computer Graphics,  
Thesis Supervisor

Accepted by.....  
Chairman, Departmental Committee on Theses



## Pages Without Paper

Christopher Martin Schmandt

submitted to the Department of Electrical Engineering and Computer Science on Dec. 8, 1978, in partial fulfillment of the requirements for the degree of Bachelor of Science.

### Abstract

This thesis describes research on size, resolution, and processing speed for high quality fonts on a mini-computer driven raster scan display. Such a font is then used in a book-like display which formats text from ASCII files into pages of characters. Page turning animation is done in real time under the finger-tip control of the reader. The book display is a component of a Spatial Data Management System developed at the Architecture Machine Group.

Thesis supervisor: Nicholas Negroponte \_\_\_\_\_

Title: Associate Professor of Computer Graphics

the appropriate direction on a touch-sensitive joy-pad. A "page" actually consists of two pages of characters, one occupying the low order two bits (of the 8 bit per pixel display image) and the other occupying the next two bits. By changing the color matrix, display can be switched from one page to another.

The page shadows are a template stored in the high-order four bits of the display image. By modifying the color matrix changes mentioned above, various segments of the template are made to carry the shadows back and forth across the page. As the page flips, the page being exposed is visible through the shadows.

As soon as the current page is

PAGES WITHOUT PAPER

Christopher Schmandt

## TABLE OF CONTENTS

List of Illustrations	4
Acknowledgements	6
The Data Environment	7
The Display Processor	10
Font Development	14
Text and the Page Flipper	19
Future Directions	37
Bibliography	41

## LIST OF ILLUSTRATIONS

The palette, a method of graphic display of the color matrix.	12
A sample page of text.	18
A page flipping sequence. The top page turns from the bottom.	22
The page flip continues. The shadow moves across the page.	23
The page flip continues, revealing the hidden page underneath.	24
The page flip continues	25
The page flip finishes. The page underneath is fully revealed.	26
The palette showing the color matrix set to show the low page only.	28
The palette showing the color matrix set to show the high page only.	30
The shadow contour template, with color enhancement.	31
A sequence showing the color changes during a page flipping sequence. The palette shows the color matrix settings during the page turn.	33
The palette during the flip from high to low page, showing the shadow areas.	34

The palette showing the color matrix for a nearly completed page flip.	35
The color matrix at the completion of the page flip.	36

## ACKNOWLEDGEMENTS

I would like to thank a number of people who have helped provide the stimulating research atmosphere at the Architecture Machine:

Nicholas Negroponte, by thesis supervisor.

Richard Bolt, my faculty U.R.O.P. advisor for four terms here, and originator of many of the ideas embedded in this project.

William Donelson, for many useful suggestions through the whole work, as well as for teaching me the majority of what I have learned about the computing system here.

Wendy Richmond, of M.I.T. Press, for help with the font design and suggestions on page composition.

The other members of the Architecture Machine Group, especially members of the SDMS project, for encouragement, assistance, and a general pattern of investigative cooperation.



Research on size, resolution, and processing speed for high quality fonts on a mini-computer driven raster scan display is described. Such a font is then used in a book-like display which formats text from ASCII files into pages of characters. Page turning animation is done in real time under the finger-tip control of the reader. The book display is a component of a Spatial Data Management System developed at the Architecture Machine Group.

#### THE DATA ENVIRONMENT

The Spatial Data Management System (SDMS) is a multi-media interactive data management and display system centered around the concept of spatial orientation of data. In much the same way as we often refer to stored information by where it is ("on the top shelf to the right of my desk, near a large red book"), SDMS creates a data space, and an environment in which a user navigates within that space to store or retrieve information.

SDMS runs on a growing mini-computer network, utilizing several display processors and video monitors, an eight-channel sound processor, and an MCA optical video-disc. The system is driven from a comfortable, instrumented arm chair. Control inputs are taken from two joy-sticks mounted in the arms of the chair, a padded Summagraphics tablet and stylus, and a clear touch-sensitive surface

mounted over the screen of a Tektronix color monitor located within easy reach from the chair.

The spatial environment is created within the "Media Room", an 18 feet wide by 11 feet high and deep room. In the center of the room is the control chair, with two small monitors, one at each side. The walls are of visually neutral grey sound-absorbent padding, with the eight speakers of the octophonic sound system in the corners. Immediately in front of the control chair is a 6 by 8 feet screen, rear projected by a General Electric Light Valve. The Light Valve, on which video images as well as text are displayed, is driven from a frame buffer associated with an Interdata Model 85 mini-computer (see below). The two 12 inch diagonal ancillary displays are driven by a Ramtek frame buffer under control of an Interdata Model 7/32.

The user of the system navigates spatially through the data space much like flying an airplane. By pushing on the joy-sticks mounted in the chair, the image on the large screen translates in two dimensions, as well as zooming. As images on the initially viewed top level are zoomed into, they lead to or are replaced by lower levels of data, with a number of types and modes of interaction.

Data types come in many, and an expanding variety, of styles in SDMS. Many are color or black and white photographs, digitized and stored on magnetic disc. Others are

color slides or video recorded frame by frame on the video-disc. Sound is another data type, both as sound-sync with respect to a pre-recorded film or video-tape, or as recorded lectures, verbal annotations, etc. A process may itself be a data type, e.g. a display of a pocket calculator on the touch-sensitive display, which performs arithmetic operations when the "buttons" are touched. Text in various forms is another data type, either pre-recorded in ASCII on disc, or, potentially, dynamically, as from a wire service.

The primary hinderance to full use of text as a data type has been the lack of a high quality (i.e. eminently readable and "visually pleasing") font display system. What follows is description of the development and operation of such a system, suited for display of text (e.g. a book, previously entered by keyboard), page by page, on the Light Valve, under gesture control from the chair. The text display process is initiated by zooming into a photograph of the book on the top level, under control of the joy-sticks. On entry into the data "port", an auxilliary key map, on the touch-sensitive display, shows a picture of a book, with chapter headings at the side. The viewer enters a chapter by touching its name, and a page of the appropriate text is shown on the Light Valve. The reader then turns the pages by finger motions on small touch-

sensitive pads in the arms of the chair. The process is terminated by pulling back on the joy-sticks, returning to flight mode, and popping back up the the higher level of data space from which the book was entered.

#### THE DISPLAY PROCESSOR

The display processor of interest in this aspect of SDMS is an Interdata Model 85, referred to as "the 85". It is a micro-codeable machine, with a programmable control store of 1K 32-bit instructions. The 218K-byte frame buffer is manipulated by a micro-code program known as the Pixel Window. An important asset of the Pixel Window is that it allows the frame buffer to be interpreted at varying numbers of bits per pixel element. A subset, under software control, of the picture memory is selected for input to the special video display hardware, which interprets it in either RGB or YIQ color space. All accesses to the picture memory are done by mapping 4K sections of the 64K of addressable memory into the extended picture memory, keeping access times small.

The color video output is usually generated by indirection through a table, the color matrix. The color matrix allows 256 colors, each of 16 bits (YIQ) or 15 bits (RGB). Thus a pixel element of value "4" in the picture memory will produce a displayed pixel of the color which

is defined by entry 4 of the color matrix. For convenience, the color matrix is often displayed as a "palette", in which colors are displayed as a 16 by 16 grid of rectangles. Then, the color of value 4 could be seen as the color of rectangle 4 (row major) of the palette for inspection (see figure 1).

The video output may be viewed directly as an RGB signal, or sent onto a "video bus" via an NTSC encoder, and sent to any of a number of color monitors.

A further relevant feature for possible use in this project is the hardware's ability to do scaling of an image. Scaling is accomplished by repeat counts associated with both the X and Y screen directions. A scale of (2, 3) for example would repeat each pixel twice during a horizontal scan (doubling the apparent size), and repeat each raster three times successively during a vertical sweep.

In addition to enabling fast hardware zoom, this feature allows a smaller part of the frame buffer to be mapped onto a larger display area. If such a feature can be exploited, it frees picture memory (e.g. for multiple buffering), and simultaneously speeds up (relative to the viewer) the time to change or reload a display image, as fewer memory accesses are needed to the picture memory by the micro-code.

The image is defined under program control by a

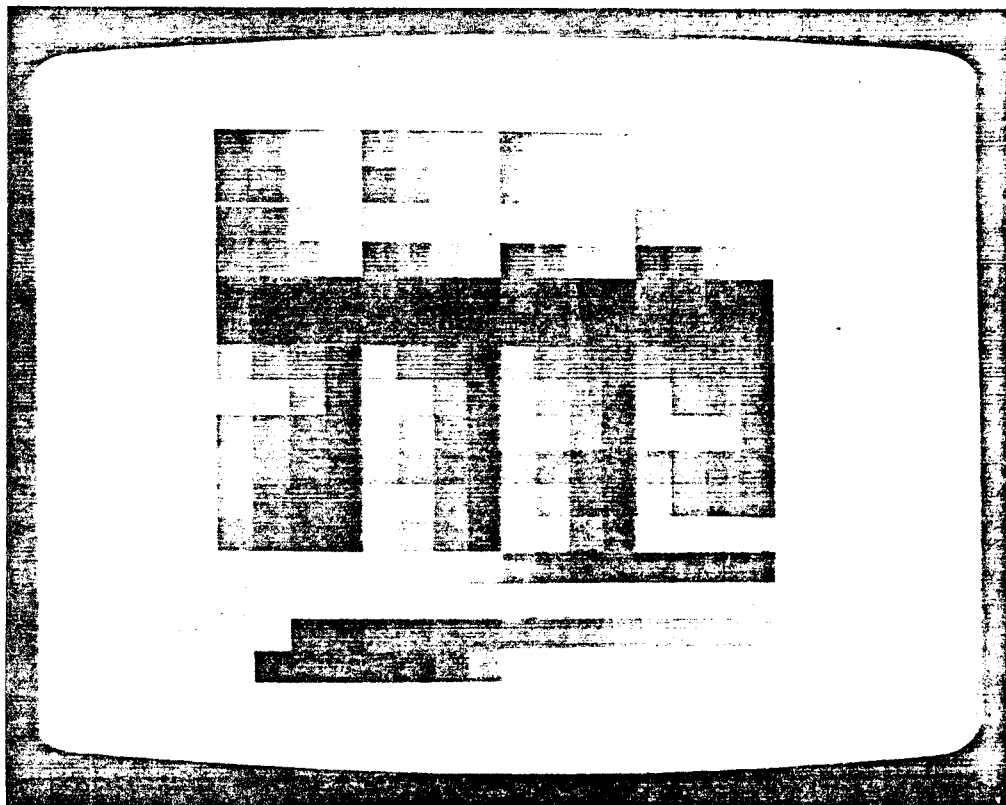


Figure 1. The palette, a convenient visual display of the color matrix. The 256 squares show the colors of each of the 256 color matrix values. The lower left square maps to color 0 (i.e., it appears with the color in entry 0 to the color matrix). The entries increase across the row to 15, then up to the beginning of the next row at 16. Entry 255 is in the upper right.

Picture Control Block (PCB), a structure which contains width, height, scale, bits per pixel, etc. In addition to the display PCB, there are two other PCB's defined in the micro-code, one for writes to the picture memory, and one for reads. These smaller PCB's also allow the width, height, bits per pixel, masks (for read and write), and starting page of extended memory to be manipulated under program control. The three PCB's allow great flexibility, namely independent interpretation of the picture memory for each of read, write, and display functions.

The heart of software control over the contents of the frame buffer is the Pixel Window. Besides setting and reading the PCB's, the micro-code supports routines to read or write between the picture memory and a user defined array, draw a rectangle of variable parameters, or move one part of the picture (by copy) to another. These operations are performed under functions, specified as a parameter on the calls; the functions take the old and new pixel values as operands. The operations range from simple replacement to arithmetic or logical functions such as add, min, and OR, performed under the appropriate masks. It is worth noting for clarity that the read and write operations use their respective PCB's, which may map different memory pages, have different masks, and

operate on different numbers of bits per pixel. In particular, the move function reads via the read PCB, and writes back out using the write PCB.

#### FONT DEVELOPMENT

Initial research efforts were aimed at determining several parameters for the font display system, particularly character size, scale, and number of grey-levels. Two somewhat mutually exclusive goals were specified. First, the font had to lead to high quality, easy to read text, especially for display on the Light Valve under SDMS. Second, it was desired to fit as much text as possible on the screen at one time, with speed of update being of concern as the text composition from ASCII files must take place in real time. This leads to a trade-off, for as the information content per character (number of pixels times bits per pixel) increases, so does readability, but also memory space requirements and time necessary for update.

The initial font models were obtained by magnetic tape from the Artificial Intelligence Laboratory's Xerographic printer font set. These fonts came at one bit per pixel, and the one selected had a 16 by 20 pixel field. Early efforts involved displaying a subset of these characters under varying conditions, and judging the results as they appeared under several display systems: the 85's own video output, an NTSC encoded



signal, and projected on the large screen by the Light Valve.

First attempts involved displaying characters at one bit per pixel, i.e. black on white, at original scale. Although they are quite readable at that scale, scintillation detracts severely from image quality. This is due largely to one pixel wide horizontal character elements flashing under interlace, but even wider (up to about ten pixels) high contrast boundaries show this effect. In addition, the NTSC encoded signal tends to produce annoying chroma blurs along any such high-frequency areas. The problem is accentuated by the Light Valve, which, probably due to ringing in the green video amplifier, generates green and red boundary lines at the top and bottom of horizontal lines.

The next attempts involved low-pass filtering, or de-jagging in graphics terminology, and at the same time size reduction. The characters were reduced by a factor or two in both dimensions, and two grey levels introduced by defining the new pixel value of a reduced two by two pixel area to be the sum of the four values (with a ceiling at three).

This character could then be displayed at scale two to restore the original size. If text could be displayed at scale two, it could be moved around the memory at four

times the previous rate; the change from one to two bits per pixel decreases space requirements, but only marginally (100 ns. per pixel) affects access time. The resulting characters are readable, reduce scintillation considerably, but are unfortunately ugly, reminiscent of fonts used in early efforts at machine recognition of characters. This made them unsuitable for use.

Returning to the original 16 by 20 one bit font, blurring was introduced by a low-pass filter of the form:

1	1	1
1	3	1
1	1	1

By then manipulating the color matrix to adjust the intermediate levels of grey, excellent results can be achieved on any of the display systems. Contrast can be set to virtually reduce scintillation, the characters are clean, and are quite readable at scale one. In fact, the 16 by 20 font appears out of proportion, so it was squeezed horizontally to twelve pixels by an algorithm that reduces three contiguous vertical pixel bands to two. As the result is well balanced, it is speculated that the Xerographic pixels are rectangular instead of square as on the 85.

The use of the Light Valve as the primary display for text on SDMS poses some further limitations. The image degrades rapidly with distance from the center of the screen past a certain radius, mostly in luminence. As a result, the "page" for text display is limited to a 400 by 400 pixel region at the center (out of a possible 512 by 400 display image). Thus a page of text allows a maximum of 700 characters, more than double that of prior attempts (see figure 2). In practice, white space reduces this number to 400 to 500 typically. In fact, this font has been displayed via NTSC at up to 40 characters per line, retaining a high image quality.

Experimentation with inter-character and inter-line spacing led to a one-dimensional kerning table. Although a two-dimensional table is desirable, to allow spacing to be a function of the preceeding and following characters, memory space constraints ruled it out. However, the quality of a composed line of text remains acceptable.

The actual character set, font master, and kerning table were produced on a font editor written for the purpose. The font is made, a character at a time, with input taken from a keyboard and Summagraphics table. The character may be drawn one pixel at a time on a large scale grid; simultaneously the actual size character

### ASCII TEXT DISPLAY

This is a page of ASCII text displayed by the new 85 font system. The font master is stored "off screen" starting at page 0 of the extended memory. As the ASCII text is parsed, each character is "moved", via the microcode, to the appropriate spot in the currently visible space.

The fonts are two bits per pixel. Associated with each font is a location table giving its coordinates on the invisible read-image. Also with the font is a one dimensional kerning table. The kerning table is stored in program memory during the formatting phase of display.

The page is flipped by scratching

Figure 2. A page of text. This and the next five photographs show the page turning animation. A second, pre-composed, page appears underneath.

appears on the screen as well as a box showing the current color being carried by the pen. The character can also be hand drawn in continuous curves, a mode in which the pen is used as a brush. In this mode, a variable width brush becomes attached to the pen, and the character is drawn smoothly over the grid. On command, this smooth image is reduced to discrete pixels, whose grey values are determined by that percentage of the area of the corresponding grid block which has been drawn in. This latter mode has proven very useful for designing new fonts, with the point by point mode an accessory used to touch up.

For convenience, the blur and squeeze functions are included as commands within the editing environment. The characters are written, as produced, to the font "master", a 448 by 60 pixel region in low picture memory. At this time, a two dimensional location table is generated, with an entry for each character giving its x and y coordinates in the master image. At the end of an editing session, the font master, along with the kerning and location tables, are written to disc in several formats.

#### TEXT AND THE PAGE FLIPPER

The textual data to be displayed is in ASCII format, e.g. an ASCII file on disc created by any of

several mundane text editors. The file is displayed in a book format, with twenty lines on a square or rectangular "page". Text is usually black on white, but may be initialized to give arbitrary ink and paper color combinations. The background color on the display which surrounds the page may be used as a color code associated with the content.

This image is displayed on the Light Valve in front of the instrumented chair. Adjacent to the chair, on the touch-sensitive ancillary monitor, appears a book figure, with the appropriate title and chapter headings. Touching a chapter heading on the display opens the page on the large screen to the first page of text of the chapter indicated. Several levels of touch indexing may be substituted for simple chapter headings.

From this point, the 1 1/2 inch square touch sensitive pads in the arms of the chair become active as control devices. By making a flipping motion on the pad with his finger, the viewer causes the pages of text to turn; a right-to-left motion across the pad flips forward, and left-to-right flips backwards.

The flipping itself is an animation designed to somewhat resemble pages turning in a book; as the top page is turned aside, the following page is revealed, in shadow, underneath. The shadow boundary moves across the page, until it reveals the page underneath in

entirety. The effect is the "top" page being peeled away from the upper right hand corner (see figures 2 through 7). A reverse flip runs the same animation backwards.

As soon as the page flips, the next successive page is invisibly formatted in the same image, replacing the page just read. Although a backwards turn at this point now involves some delay due to formatting time, the flipper is ready for an immediate turn to the expected forward page. Several variations to this algorithm will be discussed later.

The "font master" is loaded from disc at the beginning of a session into an unused segment of low picture memory, at two bits per pixel resolution. The actual color in which the characters will appear is set independently in the color matrix.

A page is composed by parsing the ASCII input (hyphenation is not used), looking up the (x, y) coordinates of the needed letters in the location table, and moving, via the micro-code, a copy of the letter to the appropriate (x, y) position on the display page. The x position is then incremented by the value in the kerning table for that character, to give the starting position for the following character.

The display image, which coincides with the write

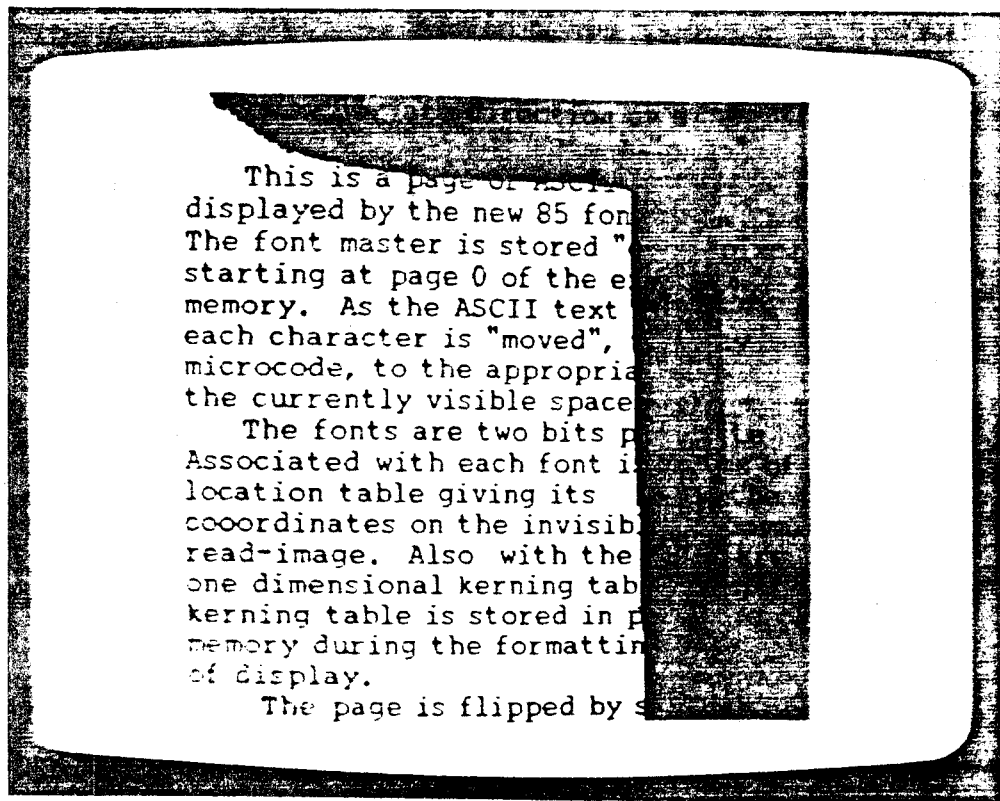


Figure 3. The page begins to turn in response to a finger gesture. The second page appears in shadow.



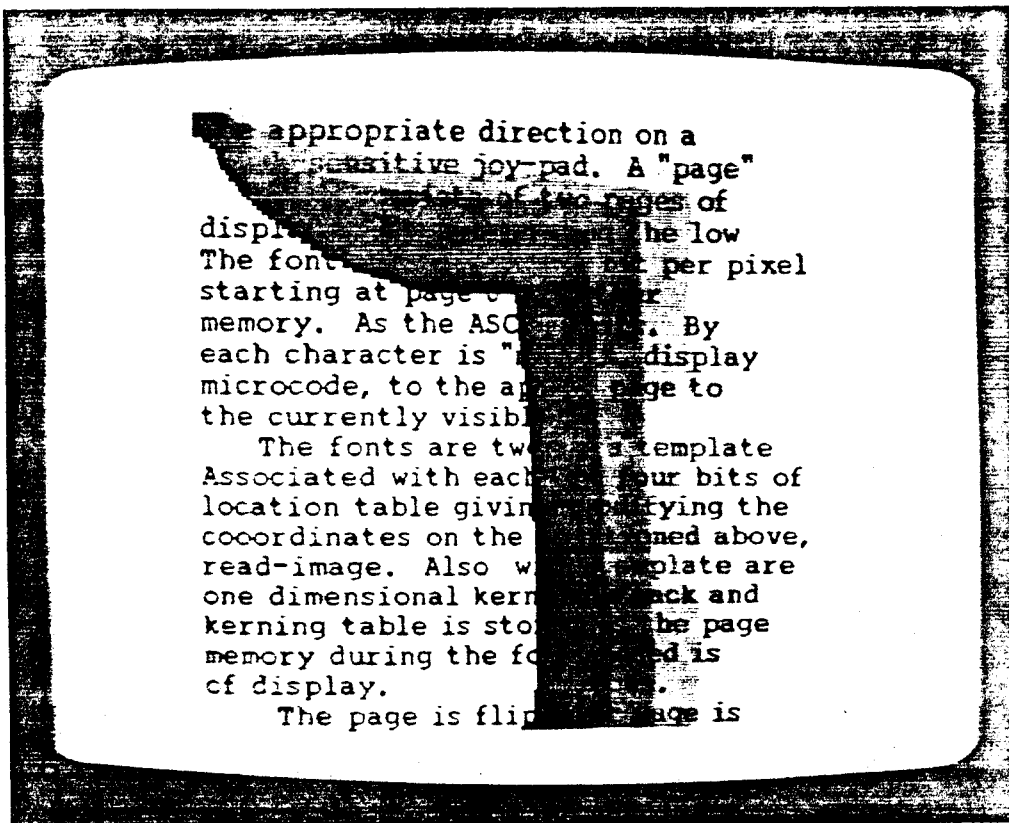


Figure 4. The shadow moves across the page, as the top page is peeled off.

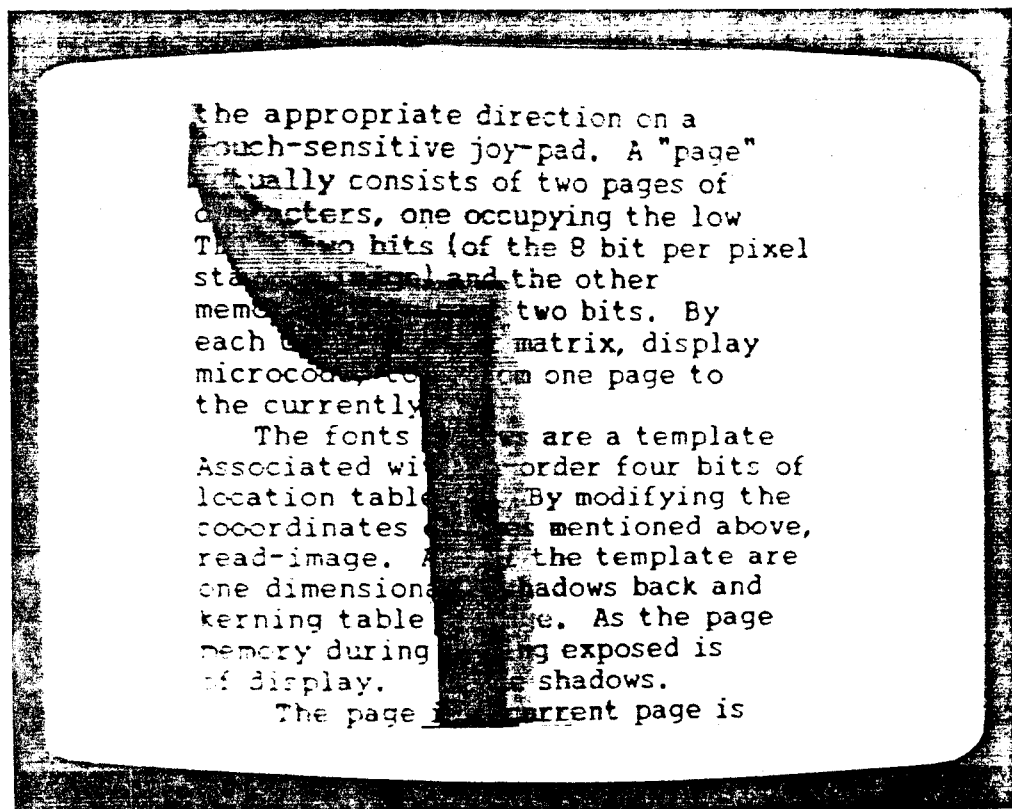
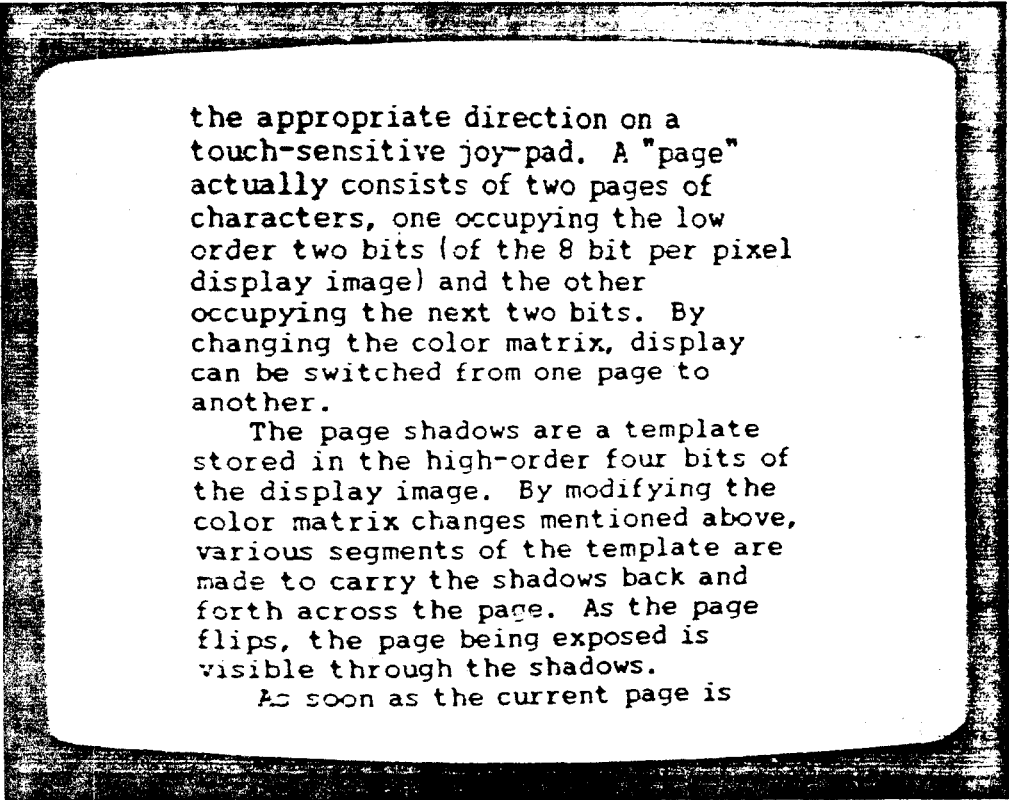


Figure 5. The shadow continues to move across, revealing more of the lower page.

the appropriate direction on a touch-sensitive joy-pad. A "page" actually consists of two pages of characters, one occupying the low order two bits (of the 8 bit per pixel display image) and the other occupying the next two bits. By changing the color matrix, display may be switched from one page to the other.

The page shadows are a template. As the high-order four bits of the display image. By modifying the color matrix changes mentioned above, read elements of the template are one carry the shadows back and kernel across the page. As the page memory page being exposed is of data through the shadows. as the current page is

Figure 6. The page turn nearly completed. This whole process has taken one to two seconds, set to be slower on the larger displays to keep constant linear motion across the screen.



the appropriate direction on a touch-sensitive joy-pad. A "page" actually consists of two pages of characters, one occupying the low order two bits (of the 8 bit per pixel display image) and the other occupying the next two bits. By changing the color matrix, display can be switched from one page to another.

The page shadows are a template stored in the high-order four bits of the display image. By modifying the color matrix changes mentioned above, various segments of the template are made to carry the shadows back and forth across the page. As the page flips, the page being exposed is visible through the shadows.

As soon as the current page is

Figure 7. The new page, completely revealed.

image in the above mentioned move, is eight bits per pixel. The independence of the read and write PCB's allows easy, dynamic conversion from the two bit font master image to the eight bit display and write image.

Of these eight bits, the low order two bits correspond to one page of text, and the next two to a second page. The high-order four bits are reserved for the shadow contours (see below). Which of these two pages is displayed can be set by changing the pattern of repetition of grey values in the color matrix. If the color matrix is written with repetitions of a pattern (white, light grey, dark grey, black) that repeats every four entries, starting from zero, the least significant two bits of the image determine the color of that point (see figure 8). Setting the color matrix this way is conceptually the same as displaying the contents of the display image modulo four.

Note that at this point, since display is modulo four, changes in the high order six bits of a pixel will not affect its color. This way, while showing a low order page, the high page can be written into the same memory invisibly.

To switch to the high order page, the color matrix is re-written with a pattern repeating in multiples of sixteen, i.e. (white for four entries, light grey for

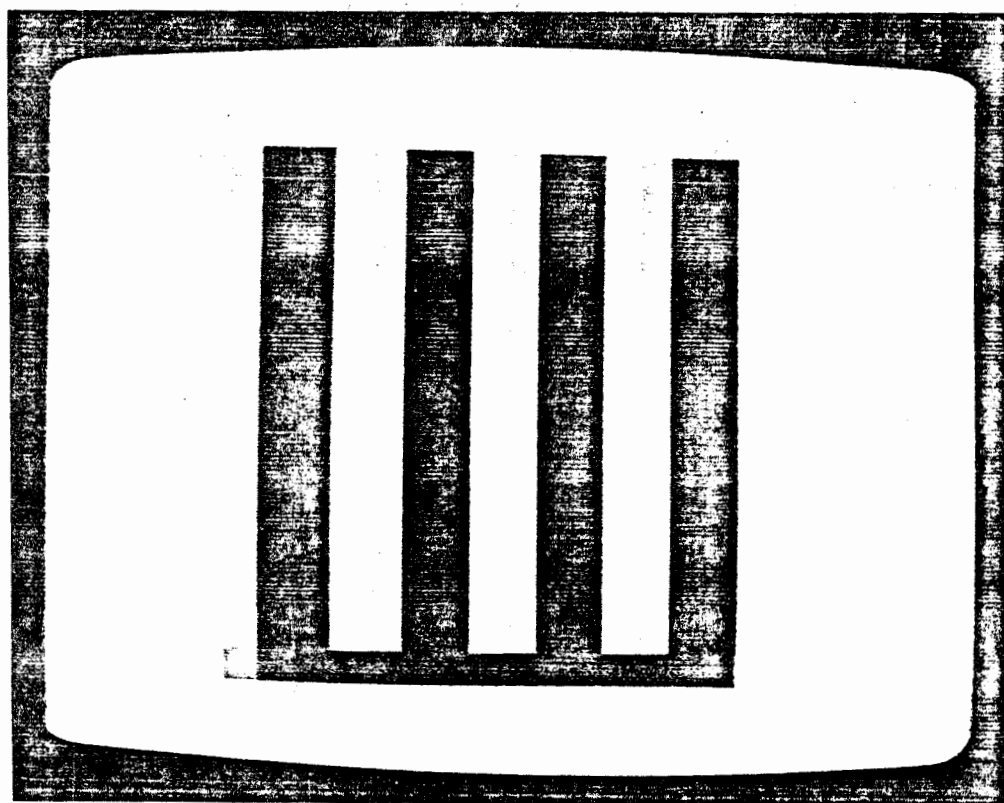


Figure 8. The palette showing the color matrix for display of the low order page. Note that the darker regions actually consist of two shades of grey. At this point, the two least significant bits of a pixel determine the color it is displayed with.

four entries, dark grey for four entries, and black for four entries) (see figure 9). With such an arrangement, only the next two bits determine the pixel color, and changes in the low two bits won't be visible. This allows the low page to be written to the image while the high page is being displayed. Since color matrix changes are very fast on the 85 (the entire matrix can be updated during a single vertical retrace), the change can be accomplished smoothly.

As the PCB's in the micro-code include read and write masks, functions are available which allow the two bit master copy of a character to be moved into either page slot of text in real time. The two bit character can be moved into bits zero and one or bits two and three without affecting any other bits of the eight bit pixel. Likewise, a high or low page of text may be erased by drawing a rectangle over the whole page, under the function of a bit-wise AND, using hexadecimal "f3" or "fc" respectively.

At the beginning of a session, the high four bits of the page image are loaded from disc with the template for page shadows. The template consists of regions drawn with successive multiples of 16, which fill the high order four bits of each pixel, thus dividing the page into separately colorable blocks (see figure 10).

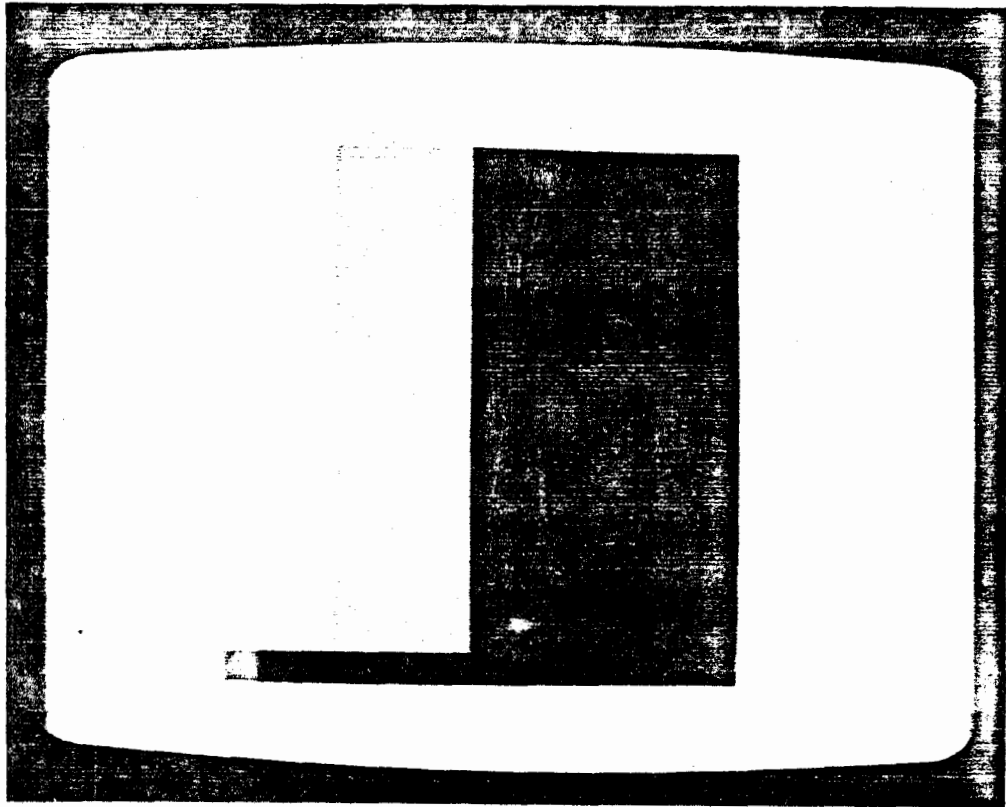


Figure 9. The palette showing the color matrix for display of the high order page. Since the pattern repeats such that four successive pixel values are the same colors, the two least significant bits no longer influence the color with which the pixel is displayed.



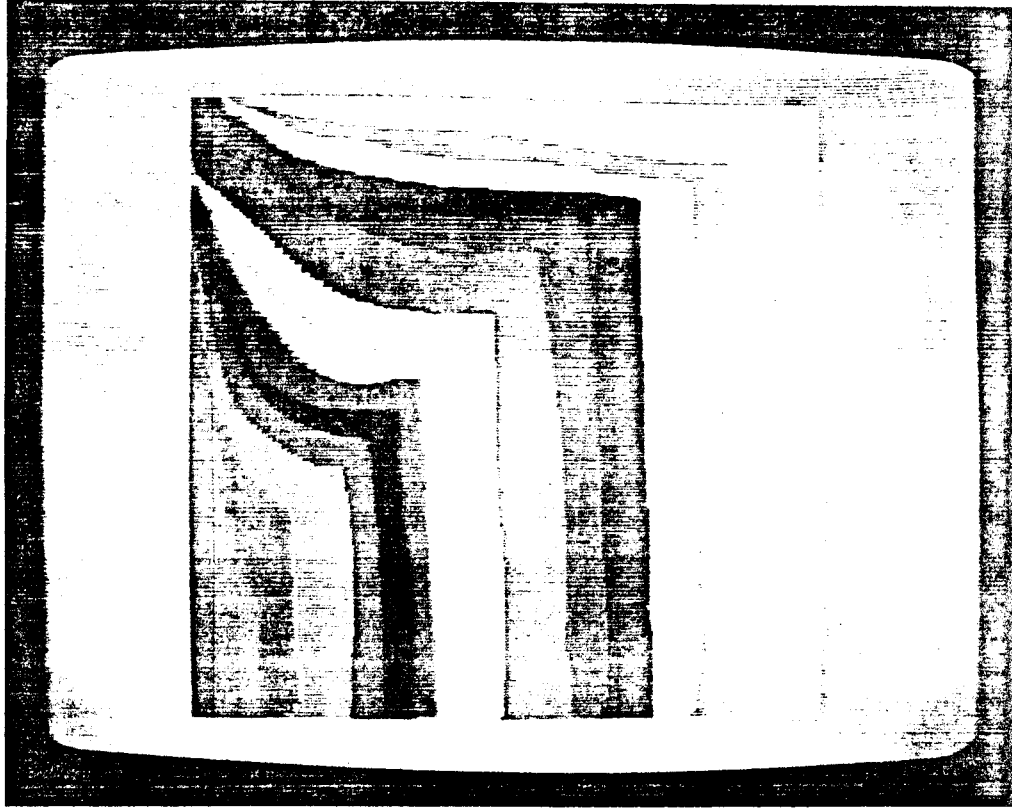


Figure 10. A sample shadow template. The grey scale has been added to aid display for photography; under normal use the contours are visible only as a shadow moves across the page. The templates may be varied for interest, from one book to another.

The normal page display color matrices (figures 8 and 9) render the shadow areas invisible, as they are all the same color.

The regions produced by the template are used to move shadows across the page. For example, to move forward (i.e. right to left), the procedure runs as follows. First, change the color matrix values from entries 16 to 31 only from the repeat by four pattern to the repeat by sixteen pattern (note that the right-most contour has value 16, i.e. the region from color zero to color 15 isn't used). Suddenly, the rightmost section of the page shows the high order page (note that the rightmost area is that with "16", i.e. "0001" in the high order four bits of the template, and thus the low order four bits of that region select colors 16 to 31). This visible area expands as we do the same operation to matrix entries 32 to 47, and on to successive 16 entry sections. Figures 11 through 14 illustrate this complete process.

Shadow is added to the colorings by using a dark grey instead of white in the new color matrix repeat by 16 pattern. This grey propagates across ("up" to color matrix as represented by the palette photographs). In the meantime, light greys are introduced from the right (bottom), leading eventually to white, which marks the

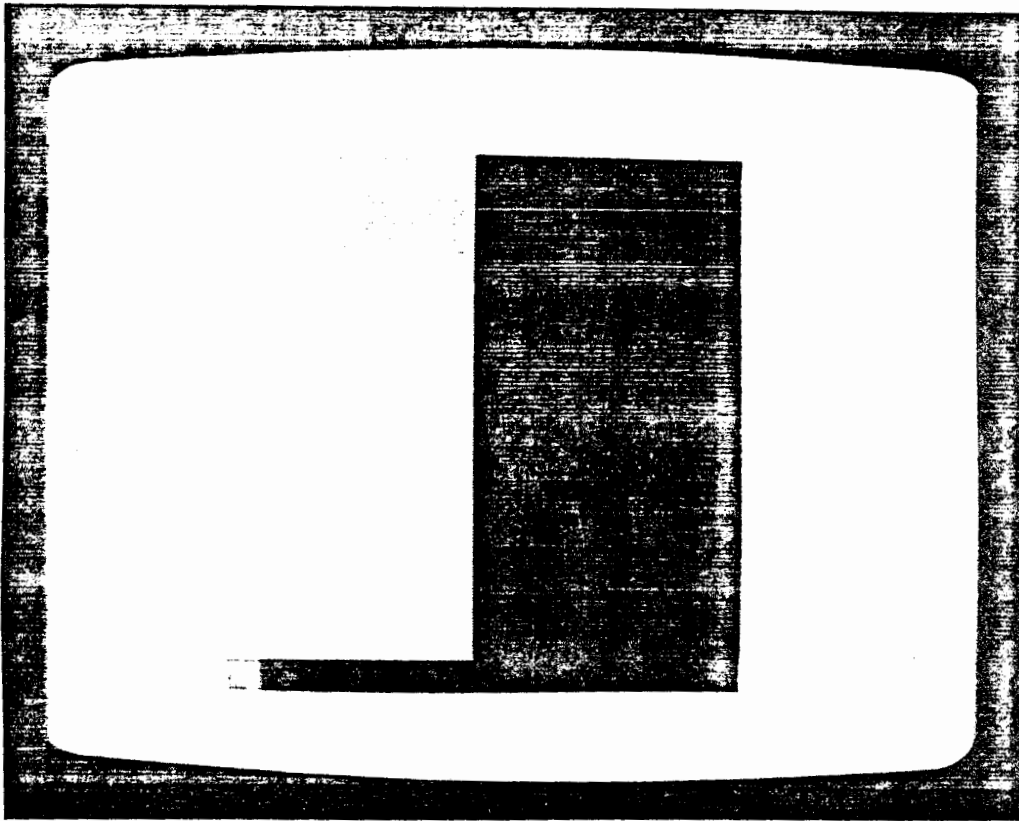


Figure 11. This and the next three photographs show the color changes that take place during a page flip. At the point in time shown here, the high order page is displayed in entirety.

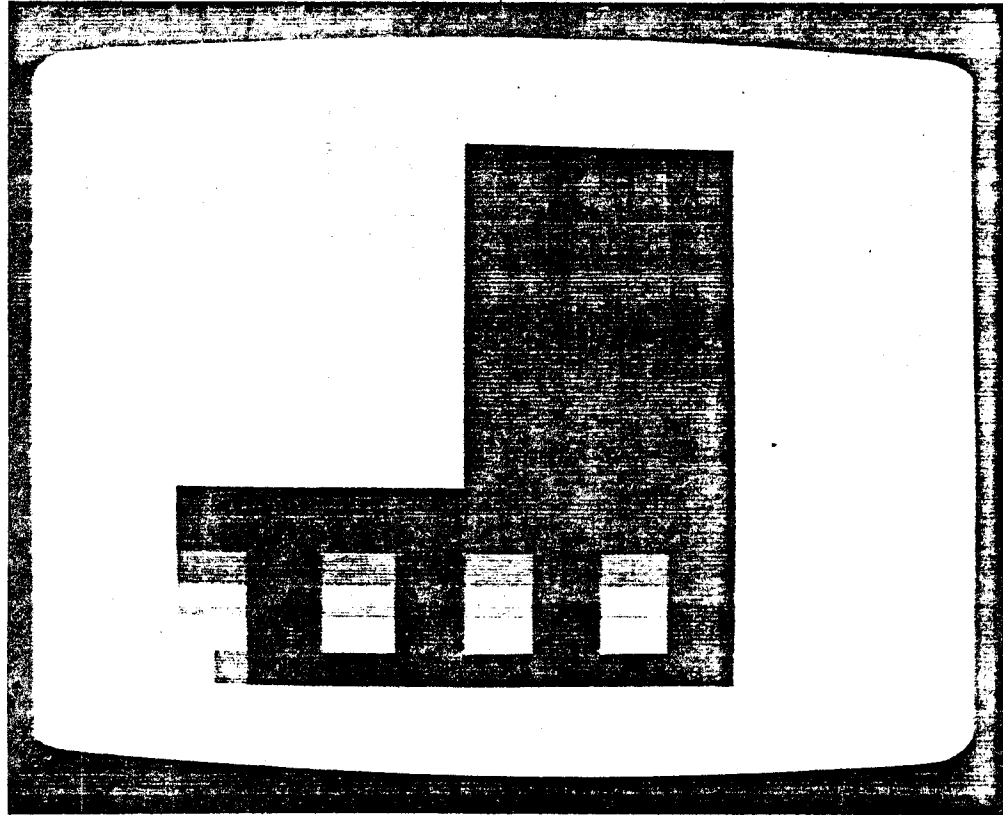


Figure 12. At this point, the next page, which is a low order page, is visible at the upper right side of the page, indicated by the repeat-by-four pattern at the bottom of the color matrix. The dark band in the lower half of the color matrix is the shadow. Above (to the left, in the page display), the high order page is visible. Below it, the low order page is coming in from the right side of the page. This would correspond to a page display as seen by the reader similar to that of figure 5.

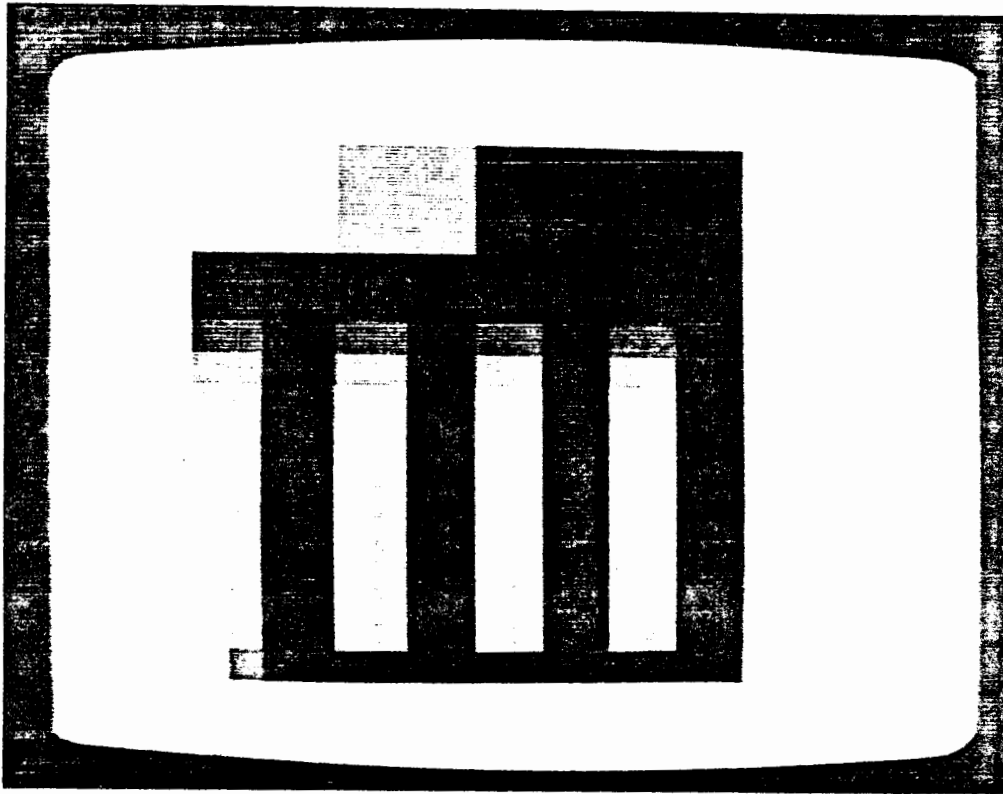


Figure 13. Now the shadow has progressed much of the way across the page ("up" the color matrix). This corresponds to a display similar to a display image resembling figure 6.

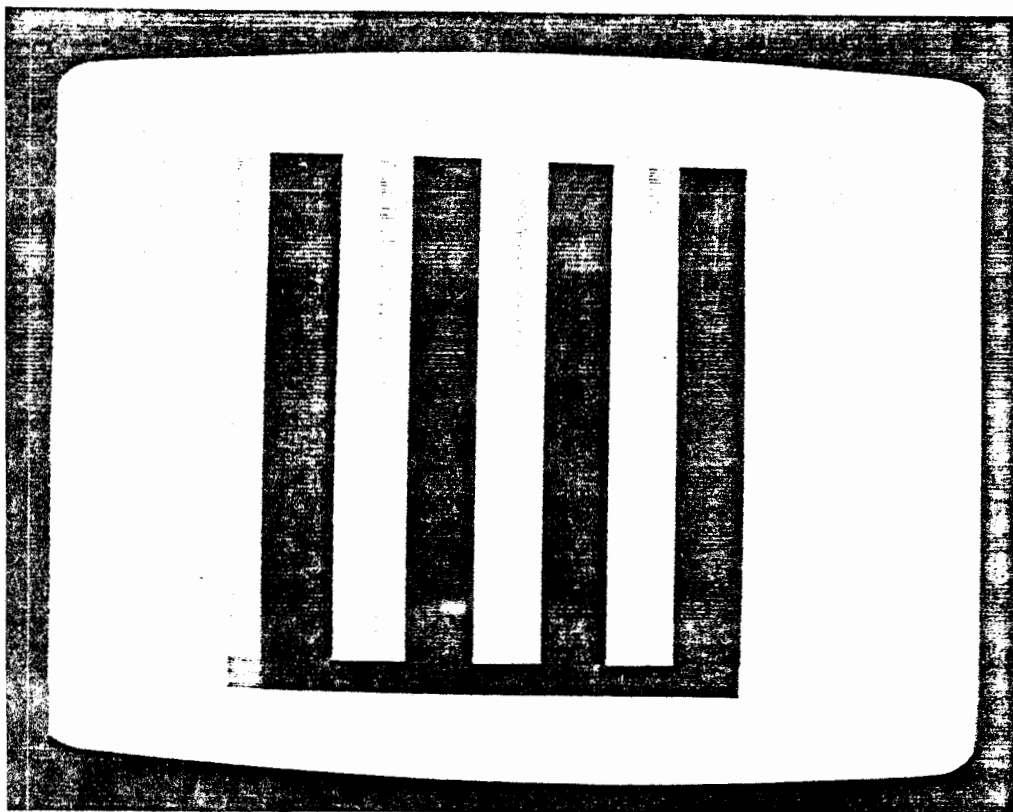


Figure 14. The shadow has swept across the page, revealing the page underneath completely.

end of the shadowed region. The visual effect produced by this shadow sweep is that of the top page being lifted and turned ("peeled" is descriptive), with the bottom page being revealed under the moving shadow of the top. Reversing this process produces the effect of pages being turned backward in a book, covering the current page.

#### FUTURE DIRECTIONS

This is a rather open ended project at the moment, as high quality text is a new element in SDMS and paves the way for further research in the use of text as a data type. The first considerations will relate to integrating the text display process into SDMS as a system, particularly under the currently being completed new network of the laboratory's processors running in the multi-processing environment of the Magic Six operating system.

The work described here has been done using the 85 in stand-alone mode, and, in such a configuration, it takes 1.5 to 2 seconds between page flips to prepare the next page. If the reader flips the pad with his finger in this interval, a wait is necessary. Parallel processing should reduce that time significantly; the central machine could easily do more than just initiate a text perusal process on the 85, and instead do some

concurrent processing. One obvious division of labor is for the central machine to watch the control chair, and on cue do the parsing, sending over (character, x position on the page, y position) triples into a page buffer. The 85 would be reduced to the role of a sophisticated display processor, with resulting time saving. The actual benefits to be gained from this configuration will depend mainly on the speed of inter-processor communication relative to the time needed to move characters around in the 85's picture memory.

Other uses of parallel processing in the network are possible. One suggestion is to have the idle sound processor play the sound of paper rustling as the page turns. The auxiliary Ramtek display could be used as a medium for annotation of the 85 driven page of text, which has no room left in the color matrix to support the extra bit needed for transparent inks. The two video signals could be mixed, and annotations written into the Ramtek, so they would appear as overlays. Video mixing under computer control will allow the processors to make the annotations disappear when the page turns, most likely as a rapid fade out.

As to the content of the page, the two bits used for a page of text could also be used for charts or



diagrams loaded from picture sections of disc. This would involve some modification of existing disc-to-picture memory routines, to use the high order page. Likewise, in a somewhat slower process, grey scale reductions of photographs could be loaded, which then gain color resolution after the page has been turned.

Other work will involve the flipping gesture recognition algorithms, possibly making them more idiosyncratic. Several modes are possible. First, it should be noted that in normal reading one turns pages of a book backwards rarely, and usually only shortly after turning forward, to pick up a lost train of thought. This suggests having the flipper pause some number of seconds after turning a page, watching for a backward turn, before continuing and replacing the previous page with the page next to appear.

The opposite extreme is skimming mode, in which pages aren't being perused but rather scanned quickly. In this mode, pages would be built up from right to left, and the process would continue watching the pads for a second flip, which could interrupt the first flip already in process, and start moving yet another page in. Since the two modes tend to be mutually exclusive, the challenge lies in making an intelligent process, able to determine which type the user is or which mode of reading she is doing at the moment, and draw on the

appropriate algorithm.

Even further off are dynamic text data bases. Text could be taken from a wire service, for instance, and a background system would continue to update a "news" data base. On initiation of a "read news" process, some data base map would indicate areas (e.g. topics, geographic regions, names) about which current news is ready to be displayed.

## BIBLIOGRAPHY

Artificial Intelligence Laboratory, M.I.T., XGP Font Catalog, Working Paper Number 72, May 24, 1974.

Bolt, R. A. Spatial Data-Management - Interim Report, M.I.T., Nov. 1977.

Bolt, R. A. Spatial Data-Management, manuscript in progress.

Donelson, William C. Spatial Management of Information, Computer Graphics, Quarterly Report of SIGGRAPH-ACM, Vol. 12, No. 3, Aug., 1978.

Mathews, M. V., Lochbaum, Carol, and Moss, Judith A., Three Fonts of Computer Drawn Letters, The Journal of Typographic Research, Vol. 1, No. 4, Oct. 1967.

Mergher, H. W., and Vargo, P. M. One Approach to Computer Assisted Letter Design, The Journal of Typographic Research, Vol. 2, No. 4, Oct. 1968.

Ruder, Emil, Typography, a Manual of Design, Fritz Brunner-Lienhart.

Shurtloff, Donald Relative Legibility of Leroy and Lincoln/MITRE Fonts on Television, The Journal of Typographic Research, Vol. 3, No. 1, Jan. 1969.

