

Page Turning Design for 3D Electronic Books

Lichan Hong, Stuart K. Card, and Jock D. Mackinlay

Palo Alto Research Center

3333 Coyote Hill Road, Palo Alto, CA 94304

{hong, card, mackinlay}@parc.com

ABSTRACT

Taking the form of physical books, virtual 3D books can be used as basic components of e-book systems, information workspaces, and digital libraries. This paper describes the page turning design of 3Book, a 3D book that we recently developed based on our previous experiences with WebBook. Our design achieves interactive page turning by employing page textures of multiple resolutions at different stages of page turning, solving the scalability problem. By modeling all the faces of the book and synchronizing the movements of various portions of the book during page turning, our design helps to convey the impression of reading or viewing an actual physical book.

KEYWORDS: 3D book, 3D workspace, digital library, electronic publishing, visual design.

INTRODUCTION

In the past few years, several projects were initiated to digitize large amounts of books and documents [1,9]. To simulate the look and feel of a physical book in its electronic correspondent, the British Library has recently developed a system called “Turning the Pages” [2]. The Zinio Reader™ [14] and the FlipViewer™ [7] are two other examples displaying electronic pages in a 3D form simulating the appearance of a real book. All these efforts suggest the interest in adopting a 3D book metaphor that preserves the correspondence between the physical artifact and the electronic artifact. As a visually enhanced object, a book metaphor taps into the user’s familiarity with physical books. Another usage of 3D books is as design elements in 3D workspaces [10,12]. Furthermore, a book metaphor can be employed in 3D digital libraries to represent textbooks or conference proceedings [6].

One of the major technical challenges in creating 3D books is designing a mechanism for turning the pages. There are several constraints on a good mechanism. First, the page turning should be computable in real time, allowing the user

to examine different pages of the book interactively. Second, the mechanism should be scalable to handle books with large numbers of pages. Finally, the page turning must resemble a physical page turning. This does not necessarily mean that a physical model depicting the details of a turning page has to be employed. Rather, the page turning needs to be sufficiently natural that it supports the book metaphor.

WEBBOOK PAGE TURNING

As an example of a 3D book page turning mechanism, consider our work on WebBook [4] (see Fig. 1). WebBook is a 3D book of HTML pages developed for the Web Forager system. Fig. 2 illustrates the cross section of a WebBook open to a double page (e.g., the WebBook at the center of Fig. 1). The cross section is a 2D plane perpendicular to the book binding that connects the pages of the book together. Although the book is modeled in 3D, for discussion purposes, when appropriate in this paper we use simplified 2D cross sections.

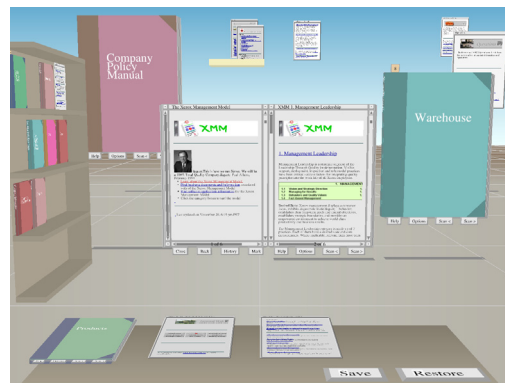


Figure 1. WebBook and Web Forager.

In Fig. 2, the left side of the book, which we refer to hereafter as *the left block*, consists of a thickness indicator AB, a scrollbar BC, a valley CDE, the left page EF, a valley FGH, a scrollbar HI, and a border IJ. Similarly, the right side of the book, which we refer to hereafter as *the right block*, consists of a border JK, a scrollbar KL, a valley LMN, the right page NO, a valley OPQ, a scrollbar QR, and a thickness indicator RS. AB and RS are used to depict approximately the numbers of pages in the left block and the right block, respectively. BC, HI, KL, and QR are used to perform certain predefined functionalities. CDE, FGH, IJK, LMN, and OPQ are used to reinforce the 3D effect by highlighting the transition between two contiguous regions. The

width of EF and NO is determined by the width of an individual page.

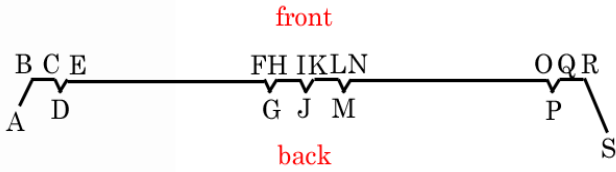


Figure 2. Cross section of a WebBook open to a double page (as seen from the bottom of the book).

When a page of the WebBook needs to be turned forwards, a turning block representing the page being turned is introduced at the beginning of the page turning and overlapped with the right block (see Fig. 3). The turning block, which consists of a top face and a bottom face, rotates counter-clockwise around point J of Fig. 3. Meanwhile, point J moves down at a constant speed until the turning block has rotated 90 degrees. Point J then travels back at the same constant speed along an opposite direction until the turning block has rotated 180 degrees. At this point, the turning block becomes invisible and the page turning terminates.

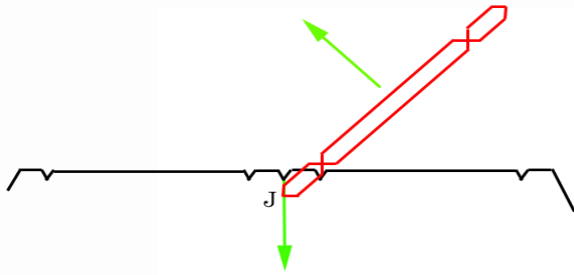


Figure 3. A turning block of a WebBook has rotated 45 degrees counter-clockwise.

The limitations of the WebBook’s page turning mechanism become apparent when we consider it against our criteria. First, in order to achieve interactive page turning, WebBook requires that all the page textures be pre-loaded and kept in main memory, restricting the number of pages that could be included in a book. According to our observation, as a book became larger than about 40 pages, the system would slow noticeably. Second, as shown in Fig. 3, the faces of the turning block stay flat throughout the page turning, reducing the affordance of the book metaphor. Finally, as can be seen from Fig. 2 and Fig. 3, WebBook looks reasonably realistic when it is viewed from the front side (i.e., its double page faces the user). In a 3D workspace where the virtual camera can be freely moved around and may see the back side of a WebBook, however, the book will not resemble a real book anymore.

Recently, taking a step further from what we had learned in creating WebBook, we developed a new 3D book called 3Book [3]. This paper focuses on describing how page turn-

ing is carried out in 3Book. In the next section, we present the page turning design in detail and discuss its major contributions.

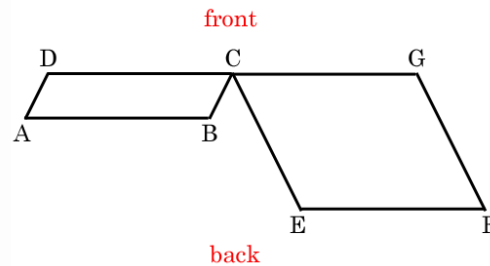
3BOOK PAGE TURNING

Static Mode

Fig. 4a shows a 3Book open to a double page, and Fig. 4b illustrates the cross section of the 3Book in Fig. 4a. As shown in Fig. 4b, the left block of the book is represented with a parallelogram ABCD, and the right block is represented with another parallelogram EFGC. ABCD and EFGC connect at point C. The width of a block (e.g., AB or EF) is determined by the width of an individual page, and the thickness of a block (e.g., the distance between AB and CD) is determined by the thickness of an individual page as well as the number of pages in the block. The slanting angle (e.g., DAB or EFG) is a predefined constant.



(a)



(b)

Figure 4. (a) A 3Book open to a double page. (b) Cross section of the 3Book in (a).

Dynamic Animation

In the following, we provide a sequence of snapshots to illustrate how page forward is carried out in 3Book. Page backward is accomplished in a symmetric way. Using Fig. 4b as our static book, suppose that due to some user interaction, the book needs to be forwarded 10 pages. Based on the thickness of an individual page, the system concludes that the portion represented by parallelogram JKGC of Fig. 5a needs to be turned from the right side of the book to the left side of the book.

Specifically, at the beginning of the page turning, the right block EFGC is split into two parallelograms EFIH and JKGC, where point H coincides with point J and point I coincides with point K. EFIH becomes the new right block

and JKCG becomes the turning block. In addition, two low-resolution textures corresponding to the top and bottom faces of the turning block are loaded *on the fly* from disk and applied. Likewise, a low-resolution texture corresponding to the top face of the new right block is loaded from disk and applied. Meanwhile, two separate threads are started to load the high-resolution textures corresponding to the top face of the new right block and the bottom face of the turning block, respectively.

Next, points J and H start to move towards the position where point C was located at the beginning of the animation. At the same time, JKGC starts to rotate 180 degrees counter-clockwise, around point J. Throughout the page turning, the left block and the turning block remain connected at point C, resulting in the left block being translated downwards. Similarly, the right block and the turning block remain connected at points J and H, resulting in the right block being translated upwards. Figs. 5b-5e shows four animation frames at which the turning block has rotated 45, 90, 135, and 180 degrees, respectively. Note that as the animation proceeds, the slanting angle of the turning block is gradually deformed (e.g., CJK changes from an obtuse angle in Fig. 5a into an acute angle in Fig. 5e), with the height of the turning block remaining unchanged. This is to ensure that after the turning block has rotated 180 degrees counter-clockwise (see Fig. 5e), BC will be aligned with CJ and AD will be aligned with GK.

At the end of the animation, JKGC, which is already on top of ABCD, is merged with ABCD to create a new left block ABJK (see Fig. 5f). In addition, the two threads started at the beginning of the page turning are checked to see if they are done. If not, the threads are allowed to continue their executions. When the high-resolution textures are available, they replace the low-resolution textures on the top faces KJ and JI, respectively. To this end, the book returns to the static mode, waiting for further actions from the user.

Discussion

At any instance, only a small number of pages are visible from a book. This tells us that it is unnecessary to maintain all the page textures in main memory. Reading textures of newly visible pages from disk on the fly enables us to handle an almost unlimited number of pages, solving the critical scalability problem. To alleviate the considerable latency to read a high-resolution texture off disk, we generated textures of multiple resolutions from scanned page images at a preprocessing stage [3]. Employing low-resolution textures at the beginning of the page turning reduces the time interval for the first frame to appear, creating a sense of rapid user responsiveness. Applying high-resolution textures at the end of the page turning ensures that the book pages are displayed at a high level of detail in the static mode. This strategy was inspired by the technique of mipmapping [13], although in mipmapping all the textures are required to have been loaded into the texture memory and the 3D renderer automatically decides which texture to apply.

Since we model all the faces of the book blocks in the static mode and during the page turning, 3Book will resemble a physical book no matter from which angle the camera looks at it. It is also worth noting that before and after the page turning, the top faces of the left block and the right block stay at the same location (see the dashed lines in Figs. 5a and 5e). Furthermore, modeling the book blocks as parallelograms brings yet another benefit. As can be seen from Fig. 4a, the fact that both sloping sides of the book are visible makes it easy to tell, roughly, how many pages are in the left block and the right block, respectively. The sloping sides of the book can also be used to attach other objects such as bookmarks.

Our design provides a general framework to accommodate various page deformation schemes. In a companion paper [8], we describe how we curled the turning pages with an

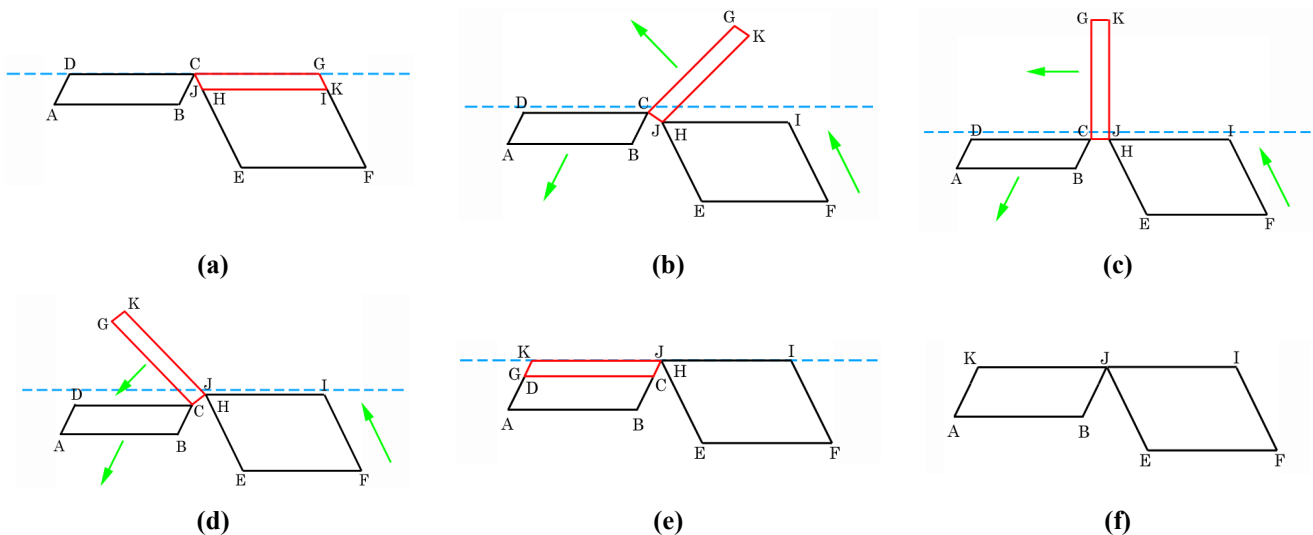


Figure 5. (a) At the beginning of forwarding 10 pages. (b)~(e) The turning block has rotated 45, 90, 135, and 180 degrees, respectively. (f) At the end of forwarding 10 pages.

imaginary cone to produce a realistic page turning effect. This deformation technique was implemented in 3Book by slightly extending the page turning mechanism described above. Fig. 6 shows one of the page turning frames. Obviously, other kinds of curved surfaces can also be used to model the faces of the book blocks.



Figure 6. Curling the turning pages with an imaginary cone to enhance the page turning effect.

Implementation

3Book was built on top of VTK 4.2 [11]. In order to integrate with other sensemaking components, we chose Java as our programming language and accessed the 3D rendering engine of VTK through Java Native Interface (JNI). The faces of the book blocks are modeled as polygonal meshes and their deformations are computed on the fly during the page turning. On a Dell Workstation 340 with nVidia GeForce4 TI 4600 graphics card, the first frame usually appears within 0.1 second and the animation speed reaches about 27 to 59 frames per second. Interactive page turning speeds of 5 to 20 frames per second have also been observed on our laptops and tablet PCs.

RELATED WORK

In the British Library's "Turning the Pages" project, photographs were taken at intermediate points during each page turning and stored in the system using Macromedia Director [5]. When a visitor turns a page, a corresponding sequence of page turning images is retrieved from the system and displayed. As a result, one version of the system consumes 304MB for only 20 book pages [5]. Obviously, the photography could be labor intensive and the number of book pages is limited.

Chu et al. [5] reported an ongoing work that aimed to simulate page turning using a mass-spring model. In their system, a book page was represented as a rectangular grid of particles connected with springs. Their implementation took from minutes to hours to calculate a single page turning, precluding real-time operation. As a result, they chose to pre-compute three page turnings under different conditions and saved the results on disk. Several implementation problems arose and were under investigation. It is not clear from the paper how they addressed the scalability issue and whether their technique can be extended to handle turning of multiple pages at a time.

CONCLUSIONS

For virtual 3D books, interactive page turning is particularly important to convey the impression of reading or viewing an actual physical book. In this paper we have presented a page turning mechanism developed for the 3Book system. Our design provides a general framework to simulate page turning under various circumstances. Additionally, it supports books of almost any size, including large medical texts or even encyclopedias containing several volumes. At the time of the Web Forager research it was barely possible to get even the smallest books to run on a PC. It is astonishing to us that large books can now be turned interactively on a laptop or a tablet PC.

The general idea of associating multiple textures with a page and dynamically deciding which texture to use applies beyond page turning. For example, a 3D workspace may contain many 3D books. Given the limited amount of texture memory in a graphics card, it is unlikely that all the visible pages of the 3D books can be displayed using high-resolution textures. Finding a scalable solution to allow as many 3D books as possible in the workspace is an interesting challenge that we plan to investigate in the future.

REFERENCES

1. Amazon (2004). Search Inside the Book. <http://www.amazon.com/exec/obidos/tg/browse/-/10197021/002-7290315-5435246>.
2. British Library (2004). Turning the Pages. <http://www.bl.uk/collections/treasures/about.html>.
3. Card, S. K., Hong, L., Mackinlay, J., and Chi, E. (2004). 3Book: A 3D Electronic Smart Book. *To appear in AVI'04*.
4. Card, S. K., Robertson, G. G., and York, W. (1996). The WebBook and the Web Forager: An Information Workspace for the World-Wide Web. *CHI'96*, 111-117.
5. Chu, Y.-C., Witten, I. H., Lobb, R., and Bainbridge, D. (2003). How to Turn the Page. *JCDL '03*, 186-188.
6. Cubaud, P., Stokowski, P., and Topol, A. (2002). Binding Browsing and Reading Activities in a 3D Digital Library. *JCDL '02*, 281-282.
7. E-Book Systems (2004). FlipViewer. <http://www.flipviewer.com/product/flipviewer/index.php>.
8. Hong, L., Card, S. K., and Chen, J. (2004). Deforming Pages of 3D Electronic Books. *Submitted for publication*.
9. Reddy, R. and St. Clair, G. (2002). The Million Book Project. Proposal to NSF. Pittsburgh, PA: Carnegie Mellon University.
10. Robertson, G., van Dantzich, M., Robbins, D., Czerwinski, M., Hinckley, K., Risdien, K., Thiel, D., and Gorokhovskiy, V. (2000). The Task Gallery: A 3D Window Manager. *CHI'00*, 494-501.
11. Schroeder, W., Martin, K., and Lorensen, B. (2003). *The Visualization Toolkit: An Object Oriented Approach to 3D Graphics, 3rd Ed.* Clifton Park, New York: Kitware, Inc.
12. Sun (2004). Project Looking Glass. http://www.sun.com/software/looking_glass.
13. L. Williams (1983). "Pyramidal Parametrics". *SIGGRAPH'83*, 1-11.
14. Zinio (2004). Zinio Reader. <http://www.zinio.com>.