Two-Pass Image and Volume Rotation

Baoquan Chen
Department of Computer Science and Engineering
University of Minnesota at Twin Cities
Minneapolis, MN 55455
baoquan@cs.umn.edu

Arie Kaufman
Center for Visual Computing and Computer Science Department
SUNY at Stony Brook
Stony Brook, NY 11794-4400
ari@cs.sunysb.edu

Abstract

We present a novel two-pass approach for both 2D image and 3D volume rotation. Each pass is a pseudo shear. However, it has the similar regularity to a pure shear in that a beam remains rigid. Furthermore, the 3D pseudo shear guarantees that beams within one major axis slice remain in the same directional plane after the shearing. These properties make it feasible to implement the pseudo shears on a multi-pipelined hardware or a massively parallel machine. Compared with the existing decompositions, ours offer a minimum number of shears to realize an arbitrary 3D rotation. Our decomposition also preserves the image/volume quality by guaranteeing no minification for the first pass shear.

1 INTRODUCTION

Many applications require interactive or even real time manipulation of rasterized data — 2D images or 3D volumes. Among various affine transformations, rotation is considered as most important and expensive. Three-dimensional volume transformation plays a key role in volume modeling and manipulation, registration of multiple volumes, as well as volume rendering. In a few volume renderers implemented on parallel distributed memory machines [6, 14, 18], a volume is first rotated to be aligned to the image grid and then orthographically projected and composited to obtain the final image. However, rotation of large data, especially 3D volume data, is very expensive. One immediate challenge is the memory access bandwidth, because rotation requires global communication and could cause contention while writing data back to the distributed memory modules. This can be the bottleneck of the approach. To address this issue, it is usually desirable to decompose the rotation transformation into a sequence of lower dimensional transformations which are much simpler to perform. Shear transformations, capitalizing on nearest neighbor connections, lend themselves to a feasible multi-pipelined hardware or parallel implementation. Any hardware with a barrel shifter can be potentially used to perform efficient shear transformation. Utilizing the neighboring connection, an array of barrel shifter is able to shift an entire beam of voxels by several units in one shift cycle [3].

There have been a number of decompositions so far. Most of the decompositions on 3D rotation are straightforward extensions from 2D decompositions into shears. There are essentially two kinds of shears used for decompositions: a pure shear or a pseudo shear. In a pure shear, a row of image/volume (also called a *beam* thereafter) is simply shifted. After the shearing, the area/volume of the image/volume stays the same. In a pseudo shear, a row of image/volume is stretched (or shrunk) when it is shifted. Therefore, after the shearing, the image/volume is either magnified or minified.

We first introduce decompositions using pure shears. A three-shear decomposition of a 2D image rotation was introduced independently by Paeth [11] and Tanaka et al. [15]. A straightforward extension of this method to 3D was proposed by Schroeder and Salem [14], also by Danielsson and Hammerin [4]. From the initially obtained nine shear decomposition sequence, they managed to merge two neighboring shears into a single shear, resulting in an eight-shear decomposition. Schroeder and Salem [4] have implemented the eight-pass rotation on CM-2. The first attempt of directly performing decomposition on 3D rotation was taken by Wittenbrink and Somani [19] and recently by Toffoli and Quick [16]. In their decompositions, three shears are needed and each shear is a *general shear* operation — first sliding (*shearing*) volume slices (a volume plane perpendicular to a major axis) along one another and then sliding beams within each slice along one another. Wittenbrink and Somani [18] have implemented the three-pass algorithm on Maspar MP-1 parallel machine. Most recently, the authors have presented in another paper [2] a group of decomposition methods for 3D volume rotation. By defining different 2D shears, such as 2D beam-shear, slice-shear, and beam-slice-shear, the decompositions of four-pass, three-pass algorithms have been derived, using pure shear as the basic transformation. In this paper, we will use *pseudo shear* to conduct the decomposition and further reduce the number of shears to two, and furthermore the pseudo shears we use have the most important property of regularity as pure shears.

Using pseudo shear, Catmull and Smith [1] have proposed a two-pass algorithm to realize a general affine transformation. To serve as a comparison for our new decompositions, we describe their decomposition in the following. First, given the rotation angle α, the rotation matrix $R(\alpha)$ rotates a point (x, y) to (x', y') as:

$$\begin{bmatrix} x' & y' \end{bmatrix} = \begin{bmatrix} x & y \end{bmatrix} \begin{bmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{bmatrix}. \qquad (1)$$

Catmull and Smith decompose the rotation matrix $R(\alpha)$ into two sequences:

$$\begin{aligned} R(\alpha) &= \begin{bmatrix} \cos\alpha & 0 \\ \sin\alpha & 1 \end{bmatrix} \begin{bmatrix} 1 & -\tan\alpha \\ 0 & \sec\alpha \end{bmatrix} \\ &= \begin{bmatrix} 1 & -\sin\alpha \\ 0 & \cos\alpha \end{bmatrix} \begin{bmatrix} \sec\alpha & 0 \\ \tan\alpha & 1 \end{bmatrix}. \end{aligned} \qquad (2)$$

Each pass of a sequence works on a separate raster direction, where each row (column) of an image is both sheared and scaled. This shear/scale makes the sampling more complex even though it is constrained within the beam. Another severe problem is that memory access becomes irregular because of this scaling, which prevents it from an efficient parallel implementation. In addition, this scaling may cause a situation called *bottleneck* problem, where a beam is first shrunk and then magnified such that the original beam can not be recovered. Later, Hanrahan [8] generalized the two-pass image transformation method to a three-pass algorithm for volume affine transformation, where all three passes separately work on three raster directions of the volume. This generalization is significant. However, it inherits the same bottleneck problem. To

solve this problem, re-ordering has to be done so that magnification always precedes minification. When it comes to 3D, this becomes non-trivial because there are many cases to permute (36 as pointed out by Hanrahan) to determine the appropriate order. In this paper, we introduce a novel two-pass decomposition on both image and volume rotation using a newly defined pseudo shear. The pseudo shear that we use shifts a beam in 2D/3D space but does not scale it, therefore, bears similar regularity of a pure shear. In addition, our two-pass decomposition can guarantee no shear doing minification comes before the second shear so that we can preserve the image/volume quality.

In the remainder of this paper, we will present our two-pass decomposition on 2D image rotation (Sec. 2) and then on 3D volume rotation (Sec. 3). We then introduce a hardware implementation (Sec. 4) for our pseudo shear, followed by some implementation results (Sec. 5) and concluding remarks (Sec. 6).

2 TWO-PASS IMAGE ROTATION

Here we propose a new decomposition of the 2D image rotation. Our decomposition is a two-pass approach. Each pass is a pseudo shear, namely X-pseudo shear or Y-pseudo shear. An X-pseudo shear is defined in Equation 3, which is illustrated in Figure 1. The dotted line square is sheared to the thick solid line parallelogram position. There, each horizontal row of the image slides with each other, and in the mean time, it is scaled in the vertical direction; but it is not scaled in the horizontal direction. This makes it feasible for parallel implementation because an image row can be accessed from the memory in parallel. Similarly, we can define a Y-pseudo shear. As all shears in our decompositions are pseudo shear, we will simply call X-pseudo shear as X shear and Y-pseudo shear as Y shear. Equation 4 gives out two sequences of decomposition featuring two orders, X shear $\rightarrow Y$ shear and Y shear $\rightarrow X$ shear. A favorable property of this decomposition is that the first pass guarantees a magnification, because the scaling factor, $\sec\alpha$, for the first shear is *always* greater than one for non-zero degree rotation. This avoids bottleneck problem automatically.

$$SH_{X_{beam}} = \begin{bmatrix} 1 & 0 \\ a & b \end{bmatrix} \tag{3}$$

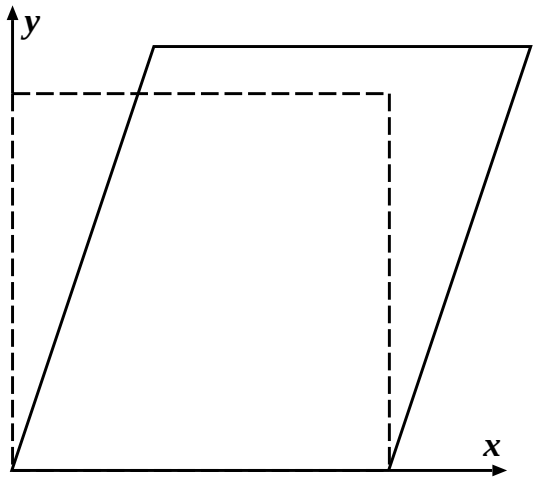


Figure 1: *2D X-beam shear*

$$\begin{aligned} R(\alpha) &= \begin{bmatrix} 1 & 0 \\ \tan\alpha & \sec\alpha \end{bmatrix} \begin{bmatrix} \cos\alpha & -\sin\alpha \\ 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} \sec\alpha & -\tan\alpha \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 \\ \sin\alpha & \cos\alpha \end{bmatrix} \end{aligned} \tag{4}$$

3 TWO-PASS VOLUME ROTATION

The property of our 2D pseudo shear is that one directional beam is kept uniform, which makes it feasible for parallel implementation because an image row can be accessed from the memory in parallel. We strive to preserve this property in 3D pseudo shear when designing the decomposition for 3D rotation. To make the shear even more regular, we further impose another constrain for a 3D shear: there is at least one directional plane so that after the shearing, all beams in that plane stay at the same plane. This maintains the regularity of the shear, which has potential for more efficient parallel implementation. In addition, we aim to deliver the least number of pass to achieve volume rotation. This requires that the designed shear be as powerful as possible within the above constrains.

We first define an X-beam-Y-slice shear, which means an X-beam is sheared within its Y-slice while the whole Y-slice is sheared along with other Y-slices. The transformation matrix of it is

$$SH_{X_{beam}Y_{slice}} = \begin{bmatrix} 1 & 0 & 0 \\ a & b & c \\ c & 0 & e \end{bmatrix}. \tag{5}$$

This shear is illustrated in 3D in Figure 2 using a perspective view, where the dotted line box is sheared to the thick solid line box position. The shaded slice of the dotted line box is shifted to the position of the shaded slice of the thick solid line box. Within the slice, X beams are sheared similarly as what is illustrated in Figure 1.

Similarly, we can define the other five 3D shears: X-beam-Z-slice shear, Y-beam-X-slice shear, Y-beam-Z-slice shear, Z-beam-X-slice shear and Z-beam-Y-slice shear. A favorable property is that the transpose of a pseudo shear is still a pseudo shear by our definition.

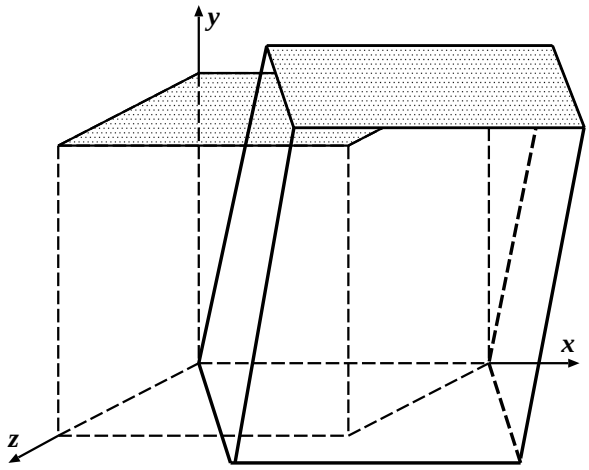


Figure 2: *X-beam-Y-slice shear.*

Now that we have defined 3D shears, we start to design the decomposition of 3D rotation by using these shears. A 3D rotation matrix can be expressed as the concatenation of three major axis

rotations, $R_x(\phi)$, $R_y(\theta)$ and $R_z(\alpha)$. A different order of this concatenation results in different 3D rotation. Without losing generality, we choose $R = R_x(\phi)R_y(\theta)R_z(\alpha)$ as our underlying 3D rotation matrix.

Let us first design a decomposition using the X-beam-Z-slice shear as the first pass. This shear has a lower triangular matrix, which is much similar to the 'general shear' used in Wittenbrink and Somani [19]'s and Toffoli and Quick [16]'s decomposition, except that there are scaling factors along the diagonal line of the matrix. Once the first shear is determined, we then derive the second shear. A first thing to know is that consecutive shears of the same type produce a conforming shear. For example, for two X-beam-Z-slice shears:

$$\begin{aligned} &= \begin{bmatrix} 1 & 0 & 0 \\ a & b & 0 \\ c & d & e \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ a' & b' & 0 \\ c' & d' & e' \end{bmatrix} \\ &= \begin{bmatrix} 1 & 0 & 0 \\ a + a'b & bb' & 0 \\ c' + ad' + ce' & bd' + de' & ee' \end{bmatrix}. \end{aligned} \tag{6}$$

In this respect, the second shear matrix has to be 'complementary' to the first shear matrix, which means for the second shear it has to be an upper triangular matrix. This makes it a Z-beam-X-slice shear. Therefore, we get the following decomposition:

$$R = \begin{bmatrix} 1 & 0 & 0 \\ a & b & 0 \\ c & d & e \end{bmatrix} \begin{bmatrix} f & g & h \\ 0 & 1 & i \\ 0 & 0 & 1 \end{bmatrix}. \tag{7}$$

This matrix equation implies nine trigonometric equations with nine variables, $a, b, c, d, e, f, g, h, i$. In solving these equations we obtain:

$$\begin{aligned} a &= \frac{\sin\phi\sin\theta\cos\alpha - \cos\phi\sin\alpha}{\cos\theta\cos\alpha} \\ b &= \frac{\cos\phi}{\cos\alpha} \\ c &= \frac{\cos\phi\sin\theta\cos\alpha + \sin\phi\sin\alpha}{\cos\theta\cos\alpha} \\ d &= -\frac{\sin\phi}{\cos\alpha} \\ e &= \frac{1}{\cos\theta\cos\phi} \\ f &= \cos\theta\cos\alpha \\ g &= \cos\theta\sin\alpha \\ h &= -\sin\theta \\ i &= \frac{-\cos\phi\sin\theta\sin\alpha + \sin\phi\cos\alpha}{\cos\phi\cos\theta}. \end{aligned} \tag{8}$$

Because a decomposition sequence can be solely determined by the first shear, we call the above sequence as X-beam-Z-slice sequence. Similarly, we can design the other five decompositions using each of the other five shear as the first pass. For each of the shear sequences, we compute the product of the consecutive shear matrices and make it equal to the target 3D rotation matrix to solve for variables.

Since the scaling operation embeds in the shearing transformation, we have to address the same bottleneck problem as in Catmull and Smith'method by solving for suitable shearing order. However, our decompositions allow us to have a very quick solution to it. As we can see from the above decomposition Eq. 7, b and e are two scaling factors for Y and Z respectively for the first matrix. From Eq. 8, e is guaranteed to be greater than one, indicating a magnification in Z. However, the scaling factor b depends on the two rotation angles: ϕ and α. Here we discuss only the situation when $0 \leq \phi, \alpha < \pi$. The discussion on the situation when $\pi \leq \phi, \alpha < 2\pi$ is similar. When $\phi < \alpha$, b is greater than one, therefore the volume is magnified in Y; otherwise, the volume is minified in Y. One approach is to permute different decomposition sequences and choose the one which is bottleneck-free. This is not too bad as only six sequences need to be evaluated, while in [8], 36 sequences have to be considered. However, our decomposition offers us a even simpler solution for some rotation angles. Let us now look at another decomposition sequence: Z-beam-X-slice sequence.

$$R = \begin{bmatrix} a & b & c \\ 0 & d & e \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ f & 1 & 0 \\ g & h & i \end{bmatrix} \tag{9}$$

where,

$$\begin{aligned} a &= \frac{1}{\cos\theta\cos\alpha} \\ b &= -\frac{\sin\phi\sin\theta\cos\alpha - \cos\phi\sin\alpha}{\cos\phi\cos\theta} \\ c &= -\frac{\sin\theta}{\cos\phi\cos\theta} \\ d &= \frac{\cos\alpha}{\cos\phi} \\ e &= \frac{\sin\phi}{\cos\phi} \\ f &= -\tan\alpha \\ g &= \cos\phi\sin\theta\cos\alpha + \sin\phi\sin\alpha \\ h &= \cos\phi\sin\theta\sin\alpha - \sin\phi\cos\alpha \\ i &= \cos\phi\cos\theta. \end{aligned} \tag{10}$$

As we can see, a and d are two scaling factors for X and Y respectively for the first shear, where a is guaranteed to be greater than one. However, scale factor d depends on the two rotation angles: ϕ and α. Opposite to the first sequence, when $\phi > \alpha$, d is greater than one; otherwise, it is less than one. Therefore, once we have these two sequences, theoretically we can choose one from them for any input rotation angles so that magnification comes first, hence bottleneck free. One problem occurs when either ϕ or α is close to $\frac{\pi}{2}$, the magnification factors become very large. When this is the case, we need to permute through other four decomposition sequences and find the most appropriate sequence to use based on the input rotation angles. The other four decomposition sequences are given in Appendix A.

4 SHEARING ON THE CUBE ARCHITECTURE

Pure shear transformations have been implemented on several massively parallel distributed memory machines, such as the Connection Machine (CM-200) [14], Maspar MP-1 [17, 18, 19], and CAM-

8 [16]. Because of the semi-regularity of our pseudo shear, it can also be very efficiently implemented on these machines.

Here we describe an application and its hardware design to perform efficient shearing on the Cube architecture [13]. Cube is a multi-pipelined special-purpose hardware design for real-time volume rendering, which has been developed at the State University of New York at Stony Brook and later adopted by Mitsubishi Electric for its VolumePro board [12]. In the current Cube/VolumePro architecture, only a single volume is rendered at a time. However, multiple overlapping volumes are common in the real world scene. Consider the scenery where smoke rises up through a cloud, or a radiation beam penetrates through a human organ. When objects occupy the same space, colors from each object must be separately classified and shaded prior to being modulated [10]. Because the Cube design features a slice-by-slice processing order, slices from different overlapping volumes have to be interlaced for a correct rendering. Because the slice is determined by the storage order in memory, it is critical to align the overlapping volumes so that their memory storage reflects their physical positions. This involves a rotation transformation. Straightforward hardware implementation of volume rotation is very expensive [5, 7]. Rotation requires global communication and could cause memory contention while writing data back to the distributed memory modules. However, as the shear transformation capitalizes on the nearest neighbor connections, it lends itself to an extremely feasible multi-pipelined hardware implementation. Because our design is based on the existing Cube architecture, we desire to take full advantage of the existing Cube design to save hardware. There are two major issues to address when performing a shear transformation on Cube: (1) the parallel access of a beam and (2) the interpolation. Let us consider the memory access first. In Cube design, we use a distributed skewed volume buffer [9]. A voxel with space coordinates (x, y, z) is mapped onto the k-th memory module out of n modules by:

$$k = (x + y + z) \bmod n \qquad (0 \leq k, x, y, z \leq n - 1) \qquad (11)$$

The data is distributed and skewed across the volume memory modules. By providing direct connections from each of the n Cube processing unit to its dedicated volume memory module, this 3D skewed organization of the n^3 voxels enables conflict-free access to any directional beam of n voxels. Because the shear transformations we have proposed guarantee a rigid translation of beams in at least one major axis, therefore, after the beams are shifted (translated), they can still be written into the memory conflict free. We utilize the neighboring connections between the Cube processing units to shift a beam across the Cube processing units, much like a barrel shifter (cf. [3]).

Cube/VolumePro also features a slice-by-slice processing order and has on-chip slice buffers to cache voxel slices for interpolation. Our pseudo shear also supports slice processing order. We can utilize this on-chip slice buffers to take advantage of the memory access coherence to lower the bandwidth while employing the trilinear interpolation unit of Cube to perform interpolation.

5 IMPLEMENTATION AND RESULTS

We first demonstrate the image rotation using our two-pass algorithm. Figure 3a is the original image; Figure 3b is the intermediate image after the first shear, and Figure 3c is the result after the second shear and is the final result.

For volume rotation, we use a Gazebo volume data for demonstration. The original volume has the resolution of $67 \times 127 \times 67$, with each voxel value ranging from 0 to 255. Volumes are rendered using ray-casting. The original volume is shown in Figure 4a. Fig. 4b and Fig. 4c show two results of X-beam-Y-slice sequence achieving the 3D rotation of $\phi = 30^\circ, \theta = 30^\circ$, and $\alpha = 45^\circ$. After the two consecutive shears, the volume has resolution of $253 \times 232 \times 114$ and $155 \times 159 \times 154$, respectively. As can be seen from the intermediate volume resolution, the volume is enlarged at the first step.

To evaluate the quality of different approaches, we first rotate an original volume with $R = R_z(-70^\circ)R_y(-45^\circ)R_x(-10^\circ)$ using the straightforward single pass method. Then we rotate the volume with $R = R_x(10^\circ)R_y(45^\circ)R_z(70^\circ)$ using different decomposition sequences. The results are depicted in Fig. 5. In the first implementation, we use the Z-beam-X-slice sequence, which minifies the volume first; while in the second we use X-beam-Z-slice sequence, which magnifies first. In the last implementation, we use a straightforward single pass to rotate the volume back. The difference volume between the twice rotated volume and the original is then calculated and volume rendered. The transfer function used to render the difference volumes is a linear ramp between 5 and 50 changing from zero to full opacity. Fig. 5a shows that Z-beam-X-slice decomposition produces the largest error, while the single pass rotation creates the least amount of error as can be seen from Fig. 5c. As shown in Fig. 5b, X-beam-Z-slice decomposition produces much less error than the Z-beam-X-slice decomposition and is very close to the single pass implementation.

6 CONCLUSIONS AND SUMMARY

We have presented two-pass decomposition methods for both 2D image and 3D volume rotation. The obvious advantage of our method is that (1) it requires the least number of shears to perform an arbitrary volume rotation. Compared with Catmull and Smith's method, (2) our decomposition has the regularity property of the pure shear in that it guarantees parallel access of a rigid beam. Therefore, it is feasible for implementation on data parallel computer architecture or multi-pipelined architecture. (3) It is more efficient to address the bottleneck problem. We permute at most only six decomposition sequences, which is much less than other methods, e.g., at least 36 situations have to be permuted in Hanrahan's method. (4) Compared to two-pass shear in [2], our shear supports a slice-by-slice processing order. This allows to take advantage of the data access coherence to lower the memory access bandwidth.

We have further shown a straightforward and efficient implementation of the shear transformation on the Cube architecture, as shearing capitalizes on the neighboring connections between the Cube processing units. In each shear pass, an entire beam can be accessed and processed in parallel.

One disadvantage of our method is the processing volume size. Because the first shear is always magnification, therefore, we have to process a larger volume than the original one. One candidate solution is to keep track of the bounding box of the original volume so that for each intermediate shear, we don't have to work on the whole volume.

7 ACKNOWLEDGMENTS

We wish to thank Alvy Ray Smith for his inspiration on this work and pointing out the likelihood of two-pass approach for 3D rotation. This work has been supported by Office of Naval Research grant N00014011034. The first author would also like to acknowledge Grant-in-Aid of Research, Artistry and Scholarship from the Office of the Vice President for Research and Dean of the Graduate School of the University of Minnesota.

References

[1] E. Catmull and A. R. Smith. 3-D transformations of images in scanline order. *Computer Graphics (SIGGRAPH '80 Proceedings)*, 14(3):279–285, July 1980.

[2] B. Chen and A. Kaufman. 3D volume rotation using shear transformation. *Graphical Models*, 62:308–322, 2000.

[3] D. Cohen and R. Bakalash. The conveyor: an interconnection device for parallel volumetric transformations. *Advances in Graphics Hardware VI*, pages 77–85, 1992.

[4] P. Danielsson and M. Hammerin. High-accuracy rotation of images. *CVGIP: Graphical Models and Image Processing*, 54(4):340–344, July 1992.

[5] M. Doggett. An array based design for real-time volume rendering. *10th Eurographics Workshop on Graphics Hardware*, pages 93–101, Aug. 1995.

[6] R. A. Drebin, L. Carpenter, and P. Hanrahan. Volume rendering. *Computer Graphics (SIGGRAPH '88 Proceedings)*, 22:65–74, Aug. 1988.

[7] T. Günther, C. Poliwoda, C. Reinhard, J. Hesser, R. Männer, H.-P. Meinzer, and H.-J. Baur. VIRIM: A massively parallel processor for real-time volume visualization in medicine. *The 9th Eurographics Hardware Workshop*, pages 103–108, Sept. 1994.

[8] P. Hanrahan. Three-pass affine transforms for volume rendering. *Computer Graphics (San Diego Workshop on Volume Visualization)*, 24(5):71–78, Nov. 1990.

[9] A. Kaufman and R. Bakalash. Memory and processing architecture for 3D voxel-based imagery. *IEEE Computer Graphics & Applications*, 8(6):10–23, Nov. 1988. Also in Japanese, *Nikkei Computer Graphics*, 3, No. 30, March 1989, pp. 148–160.

[10] A. Kaufman, F. Dachille, B. Chen, I. Bitter, K. Kreeger, N. Zhang, and Q. Tang. Real-time volume rendering. *International Journal of Imaging Systems and Technology*, 2000.

[11] A. W. Paeth. A fast algorithm for general raster rotation. In *Proceedings of Graphics Interface '86*, pages 77–81, May 1986.

[12] H. Pfister, J. Hardenbergh, J. Knittel, H. Lauer, and L. Seiler. The VolumePro real-time ray-casting system. *Proceedings of SIGGRAPH 1999*, Aug. 1999.

[13] H. Pfister and A. Kaufman. Cube-4: A Scalable Architecture for Real-Time Volume Rendering. *Proceedings of 1996 Symposium on Volume Visualization*, pages 47–54, Oct. 1996.

[14] P. Schröder and J. B. Salem. Fast rotation of volume data on parallel architectures. *IEEE Visualization '91 Proceedings*, pages 50–57, 1991.

[15] A. Tanaka, M. Kameyama, S. Kazama, and O. Watanabe. A rotation method for raster image using skew transformation. *Proc IEEE Conf on Computer Vision and Pattern Recognition*, pages 272–277, June 1986.

[16] T. Toffoli and J. Quick. Three-dimensional rotations by three shears. *Graphical models and image processing: GMIP*, 59(2):89–95, Mar. 1997.

[17] G. Vezina, P. A. Fletcher, and P. K. Robertson. Volume rendering on the maspar MP-1. *1992 Workshop on Volume Visualization*, pages 3–8, 1992.

[18] C. M. Wittenbrink and A. K. Somani. 2D and 3D optimal parallel image warping. In *Seventh International Parallel Processing Symposium*, pages 331–337. ACM, Apr. 1993.

[19] C. M. Wittenbrink and A. K. Somani. Permutation warping for data parallel volume rendering. In *ACM SIGGRAPH Symposium on Parallel Rendering*, pages 57–60, Nov. 1993.

A Other 3D Decomposition Sequences

1. Y-beam-X-slice sequence

$$R = \begin{bmatrix} a & b & c \\ 0 & 1 & 0 \\ 0 & d & e \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ f & g & h \\ i & 0 & 1 \end{bmatrix}. \tag{12}$$

where,

$$\begin{aligned} a &= \frac{1}{\cos\theta\cos\alpha} \\ b &= \frac{\cos\theta\sin\alpha}{\sin\phi\sin\theta\sin\alpha+\cos\phi\cos\alpha} \\ c &= -\frac{\cos\phi\sin\theta\cos\alpha+\sin\phi\sin\alpha}{\sin\phi\sin\theta\sin\alpha+\cos\phi\cos\alpha} \\ d &= \frac{\cos\phi\sin\theta\sin\alpha-\sin\phi\cos\alpha}{\sin\phi\sin\theta\sin\alpha+\cos\phi\cos\alpha} \\ e &= \frac{\cos\theta\cos\alpha}{\sin\phi\sin\theta\sin\alpha+\cos\phi\cos\alpha} \\ f &= \sin\phi\sin\theta\cos\alpha-\cos\phi\sin\alpha \\ g &= \sin\phi\sin\theta\sin\alpha+\cos\phi\cos\alpha \\ h &= \sin\phi\cos\theta \\ i &= \frac{\sin\theta}{\cos\theta\cos\alpha} \end{aligned} \tag{13}$$

2. Y-beam-Z-slice sequence

$$R = \begin{bmatrix} a & b & 0 \\ 0 & 1 & 0 \\ c & d & e \end{bmatrix} \begin{bmatrix} 1 & 0 & i \\ f & g & h \\ 0 & 0 & 1 \end{bmatrix}. \tag{14}$$

where,

$$\begin{aligned} a &= \frac{\cos\phi\cos\theta}{\sin\phi\sin\theta\sin\alpha+\cos\phi\cos\alpha} \\ b &= \frac{\cos\theta\sin\alpha}{\sin\phi\sin\theta\sin\alpha+\cos\phi\cos\alpha} \\ c &= \frac{\sin\theta}{\sin\phi\sin\theta\sin\alpha+\cos\phi\cos\alpha} \\ d &= \frac{\cos\phi\sin\theta\sin\alpha-\sin\phi\cos\alpha}{\sin\phi\sin\theta\sin\alpha+\cos\phi\cos\alpha} \\ e &= \frac{1}{\cos\phi\cos\theta} \\ f &= \sin\phi\sin\theta\cos\alpha-\cos\phi\sin\alpha \\ g &= \sin\phi\sin\theta\sin\alpha+\cos\phi\cos\alpha \\ h &= \sin\phi\cos\theta \\ i &= -\frac{\cos\phi\sin\theta\cos\alpha+\sin\phi\sin\alpha}{\cos\phi\cos\theta} \end{aligned} \tag{15}$$

3. Z-beam-Y-slice sequence

$$R = \begin{bmatrix} a & 0 & b \\ c & d & e \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & f & 0 \\ 0 & 1 & 0 \\ g & h & i \end{bmatrix}. \tag{16}$$

where,

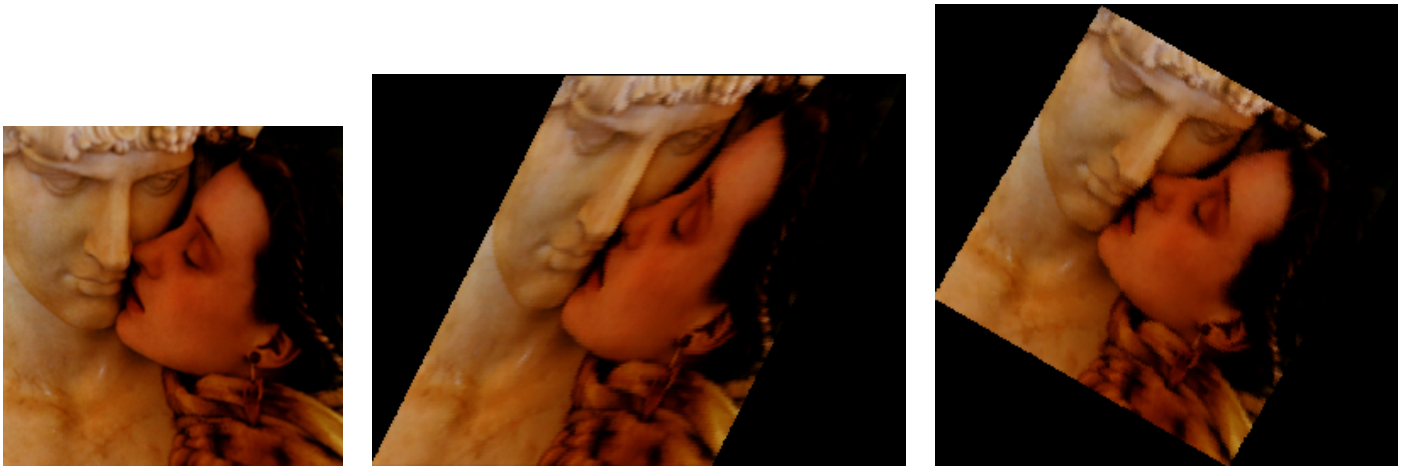
$$\begin{aligned} a &= \frac{\sin\phi\sin\theta\sin\alpha+\cos\phi\cos\alpha}{\cos\phi\cos\theta} \\ b &= -\frac{\sin\theta}{\cos\phi\cos\theta} \\ c &= -\frac{\sin\alpha}{\cos\phi} \\ d &= \frac{1}{\sin\phi\sin\theta\sin\alpha+\cos\phi\cos\alpha} \\ e &= \frac{\sin\phi}{\cos\phi} \\ f &= \frac{\cos\phi\sin\alpha-\sin\phi\sin\theta\cos\alpha}{\sin\phi\sin\theta\sin\alpha+\cos\phi\cos\alpha} \\ g &= \cos\phi\sin\theta\cos\alpha+\sin\phi\sin\alpha \\ h &= \cos\phi\sin\theta\sin\alpha-\sin\phi\cos\alpha \\ i &= \cos\phi\cos\theta \end{aligned} \tag{17}$$

4. X-beam-Y-slice sequence

$$R = \begin{bmatrix} 1 & 0 & 0 \\ a & b & c \\ d & 0 & e \end{bmatrix} \begin{bmatrix} f & g & h \\ 0 & 1 & 0 \\ 0 & i & 1 \end{bmatrix}. \tag{18}$$

where,

$$\begin{aligned} a &= \frac{\sin\phi\sin\theta\cos\alpha-\cos\phi\sin\alpha}{\cos\theta\cos\alpha} \\ b &= \frac{1}{\sin\phi\sin\theta\sin\alpha+\cos\phi\cos\alpha} \\ c &= \frac{\sin\phi\cos\alpha-\cos\phi\sin\theta\sin\alpha}{\cos\theta\cos\alpha} \\ d &= \frac{\cos\phi\sin\theta\cos\alpha+\sin\phi\sin\alpha}{\cos\theta\cos\alpha} \\ e &= \frac{\sin\phi\sin\theta\sin\alpha+\cos\phi\cos\alpha}{\cos\theta\cos\alpha} \\ f &= \cos\theta\cos\alpha \\ g &= \cos\theta\sin\alpha \\ h &= -\sin\theta \\ i &= -\frac{\sin\phi\cos\theta}{\sin\phi\sin\theta\sin\alpha+\cos\phi\cos\alpha} \end{aligned} \tag{19}$$



(a) *Original Image* (b) *First Shear* (c) *Second Shear*

Figure 3: *Two-pass shear achieves 2D rotation ($\alpha = 30^\circ$)*

(a) *Original gazebo ($67 \times 127 \times 67$)* (b) *First shear ($253 \times 232 \times 114$)* (c) *Second shear ($155 \times 159 \times 154$)*

Figure 4: *Two-pass shear achieves 3D arbitrary rotation ($\phi = 30^\circ$, $\theta = 30^\circ$, and $\alpha = 45^\circ$)*

(a) Z-beam-X-slice sequence (b) X-beam-Z-slice sequence (c) Single pass rotation

Figure 5: *Volume rendering of the difference volumes for the various decomposition sequences. The original Gazebo volume is first rotated by $R = R_z(-70^\circ)R_y(-45^\circ)R_x(-10^\circ)$ using the straightforward single pass method. Then, it is rotated by $R = R_x(10^\circ)R_y(45^\circ)R_z(70^\circ)$ using the various methods. Finally, the difference volume between thus twice rotated volume and the original volume is calculated and volume rendered. The transfer function used is a linear ramp between 5 and 50 changing from zero to full opacity (maximum voxel value 255).*