An Efficient Perspective Projection Using VolumePro™

Sukhyun Lim and Byeong-Seok Shin

Inha University, Department of Computer Science and Engineering
253 Yonghyeon-Dong, Nam-Gu, Inchon, 402-751, Korea
g2011498@inhavision.inha.ac.kr, bsshin@inha.ac.kr

Abstract. VolumePro is a real-time volume rendering hardware for consumer PCs. It cannot be used for the applications requiring perspective projection such as virtual endoscopy since it provides only orthographic projection. Several methods have been proposed to approximate perspective projection by decomposing a volume into slabs and applying successive orthographic projection to them. However it takes a lot of time since entire region of each slab should be processed, even though some of the region does not contribute to final image. In this paper, we propose an efficient perspective projection method that exploits several subvolumes with cropping feature of VolumePro. It reduces the rendering time in comparison to slab-based method without image quality deterioration since it processes only the parts contained in the view frustum.

1 Introduction

Volume rendering is a visualization method of displaying volumetric data as a two-dimensional image [1]. However it is hard to achieve interactive speed since it requires large amount of computation. For this reason VolumePro hardware was released in 1997 by Misitubishi Electric [2]. It provides real-time rendering on standard PC platform. One drawback of VolumePro is that it does not produce perspective projection images. Although an algorithm to simulate perspective projection using parallelly projected slabs in VolumePro API was presented [3,4], it takes long time since some of the entire region that does not belong to the view frustum should be processed. In this paper, we propose an efficient method to approximate perspective projection using subvolume feature in VolumePro. VolumePro can subdivide a volume data into subvolumes with a size less than or equal to the original volume. Our method renders subvolumes located in view frustum instead of the entire volume.

Direct volume rendering produces an image directly from the volumetric data [5]. Several optimization methods are devised for reduce rendering time, and they can be classified into software-based and hardware-based approaches. The software accelerated techniques usually require additional storage and preprocessing [6,7]. Hardware accelerated techniques achieve interactive speed on a specified workstation, but it is difficult to incorporate those techniques into a standard PC platform [8,9,10].

M. Bubak et al. (Eds.): ICCS 2004, LNCS 3037, pp. 396–403, 2004.

VolumePro is a hardware implementation of raycasting using shear-warp factorization. It provides real-time rendering (up to 30 fps) with compositing, classification and shading. One drawback of this hardware is that it does not produce perspective projection images. For this reason, some methods to approximate perspective projection from several parallel projected slabs are presented. These methods generate a series of slabs, which are parts of the volume data in-between two cutting planes orthogonal to the viewing direction. Then intermediate images for those slabs are generated, scaled and clipped with regard to FOV(field of view) angle. All of these images are blended to make a final image. This method is efficient when the thickness of slab is properly determined. However processing time may increase since the outside region of view frustum should be rendered for each slab.

In section 2, we briefly review the slab-based method. In section 3, we explain our method that uses subvolume in detail and describe advantages of subvolume method over previously-proposed slab-based method. Also we describe the analysis of our algorithm in comparison to conventional method. Experimental results are presented in section 4. Finally we conclude our work in the last section.

2 Projected Slab-Based Algorithm

VolumePro provides two features to generate partial volumes : a slab and a subvolume. A slab is a part of volume data defined in between two orthogonal planes and a subvolume is a partial volume against entire volume. Figure 1 shows an example of a slab and a subvolume respectively.

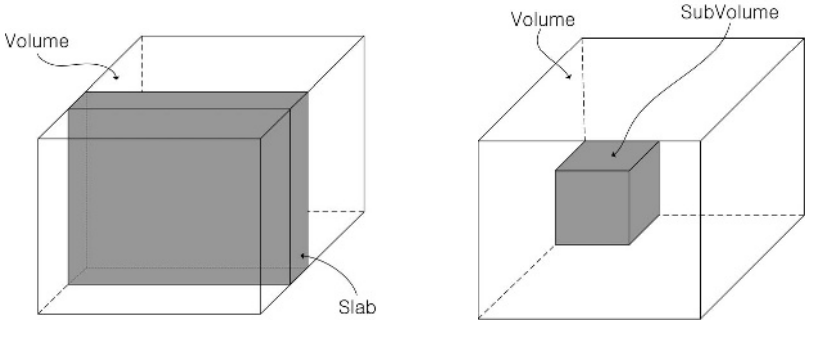


Fig. 1. An example of a slab (left) and a subvolume (right)

Figure 2 depicts the rendering process using slabs for approximating perspective projection. At first, we divide an entire volume into consecutive slabs according to viewing condition (figure 2(a)). We have to specify the plane normal, distance from camera position and thickness to define a slab as shown in figure 3.

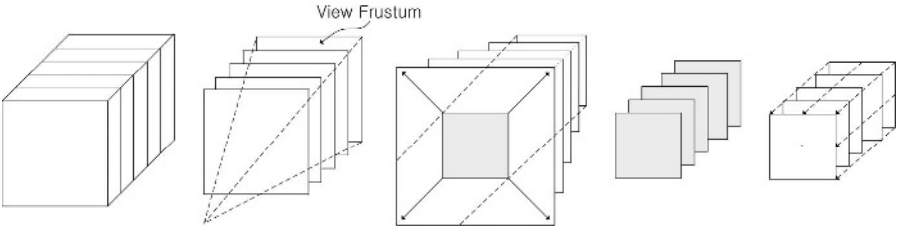


Fig. 2. Process of slab-based rendering for approximating perspective projection : (a) subdivide a volume into several slabs (b) make intermediate images for all slabs (c) normalize intermediate images according to distance from view point (d) clip those images against parallel view volume (e) blend the images to generate final image

Create(double *A, B, C, D*, double *Thickness*);
(*A, B, C*) is a normal vector
D is a distance from origin
Thickness is the distance between the two parallel planes

Fig. 3. VolumePro API used to set a slab

Next, intermediate images are generated by orthographic projection in each slab (figure 2(b)). The images are normalized (scaled) considering the distance from camera to corresponding slab (figure 2(c)). The normalized images are clipped against parallel view volume (figure 2(d)) and blended them to make final image using texture blend function in graphics hardware (figure 2(e)).

Figure 4 shows how to operate the slab-based projection algorithm. Let a thickness of *n*-th slab be Z_n and a distance from camera position to the first slab be D_0 .

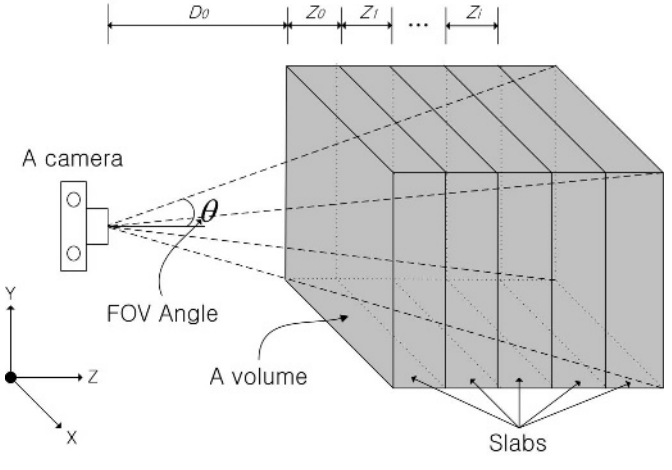


Fig. 4. Slab-based rendering using VolumePro API

3 Projected Subvolume-Based Algorithm

It is possible to set the VOI (volume of interest) region using subvolume feature in VolumePro. Figure 5 illustrates the subvolumes contained in current view frustum where Z_n is a thickness of *n*-th subvolume.

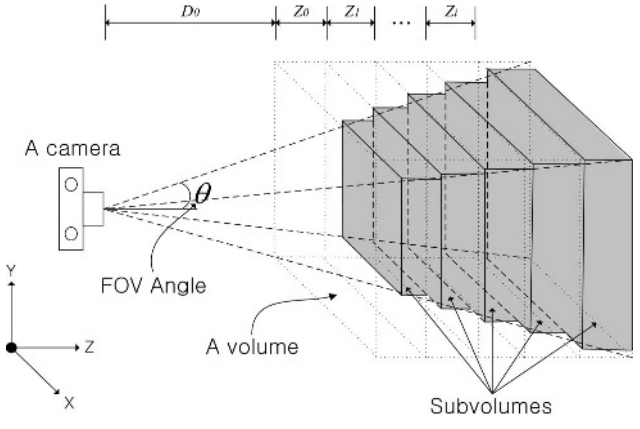
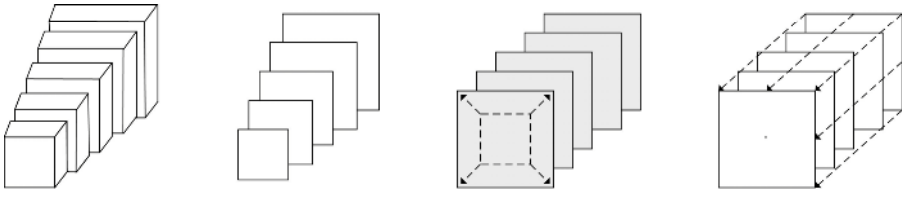


Fig. 5. Subvolume-based rendering using VOI feature in VolumePro

Gray boxes indicate regions of which the voxels are processed and transparent regions are discarded from rendering in figure 5. Therefore rendering time is much shorter than that of slab-based method and the memory for storing intermediate image can be reduced.



(a) subvolumes (b) intermediate images (c) normalization (d) blending

Fig. 6. Process of subvolume-based rendering : (a) subdivide a volume into several subvolumes (b) make intermediate images for all subvolumes (c) normalize intermediate images to fit into parallel view volume (d) blend the images to generate final image

Figure 6 shows the rendering steps using subvolume feature. After deciding the camera position and orientation, we make several subvolumes contained in view frustum (figure 6(a)). We can define a subvolume using VLICrop() API of VolumePro API as shown in figure 7.

VLICrop(double *Xmin, Xmax, Ymin, Ymax, Zmin, Zmax*);
(*Xmin, Xmax*), (*Ymin, Ymax*), (*Zmin, Zmax*) are the minimum and maximum values of a subvolume in *x*-, *y*- and *z*-coordinates

Fig. 7. VolumePro API used to set a subvolume

To specify the minimum and maximum values in principal axes, the distance from camera to *i*-th subvolume (D_i), width and height of the subvolume ($Width_i$ and $Height_i$) should be calculated as shown in Equation (1). ΔZ_i means a thickness of *i*-th subvolume.

$$\begin{aligned} D_i &= D_0 + \sum_{n=0}^{i} \Delta Z_n \\ Width_i &= 2\tan\theta(D_{i-1} + \frac{\Delta Z_i}{2}) \\ Height_i &= 2\tan\gamma(D_{i-1} + \frac{\Delta Z_i}{2}) \end{aligned} \tag{1}$$

In second step, intermediate images are generated by rendering of subvolumes (figure 6(b)) as shown in Figure 8.

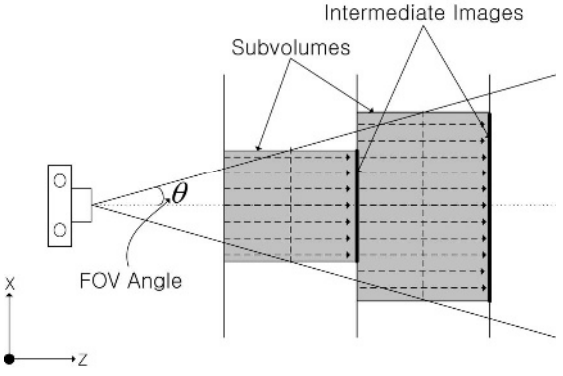


Fig. 8. How to generate intermediate image for each subvolume

Next, the intermediate image is normalized according to distance from view point (figure 6(c)). While slab-based method should clip intermediate images against parallel view volume after scaling them, all the pixels in an intermediate image are contributed to final image in our subvolume-based method, so clipping is not necessary. Final image is generated by blend function in graphics hardware just as in slab-based method (figure 6(d)).

When we use slab-based method, rendering time is composed of five components as follows :

$$T_{slab} = t_{sg}^{sl} + t_{im}^{sl} + t_{no}^{sl} + t_{cl}^{sl} + t_{bl}^{sl} . \tag{2}$$

where t_{sg}^{sl}, t_{im}^{sl}, t_{no}^{sl}, t_{cl}^{sl}, and t_{bl}^{sl} are the time for slab generation, intermediate image generation, normalization, clipping time, and blending.
In subvolume-based method, rendering time can be defined as follows :

$$T_{sub} = t_{sg}^{sv} + t_{im}^{sv} + t_{no}^{sv} + t_{bl}^{sv} . \tag{3}$$

where t_{sg}^{sv}, t_{im}^{sv}, t_{no}^{sv}, and t_{bl}^{sv} are the time for subvolume generation, intermediate image generation, normalization, and blending.

Slab generation cost is almost the same as subvolume generation cost since two functions are performed on VolumePro hardware ($t_{sg}^{sl} \bullet t_{sg}^{sv}$). Intermediate image generation time of slab-based method is longer than that of subvolume-based method since intermediate image using slab-based method is generated not for partial volume but for entire volume ($t_{im}^{sl} > t_{im}^{sv}$). Normalization cost of slab-based method is more expensive than that of subvolume-based method since normalization step in slab-based method generates image generated for entire volume ($t_{no}^{sl} > t_{no}^{sv}$). Blending time of slab and subvolume method is the same ($t_{bl}^{sl} = t_{bl}^{sv}$). Consequentially, in general, total rendering time of subvolume-based method is less costly than that of slab-based method ($T_{slab} > T_{sub}$).

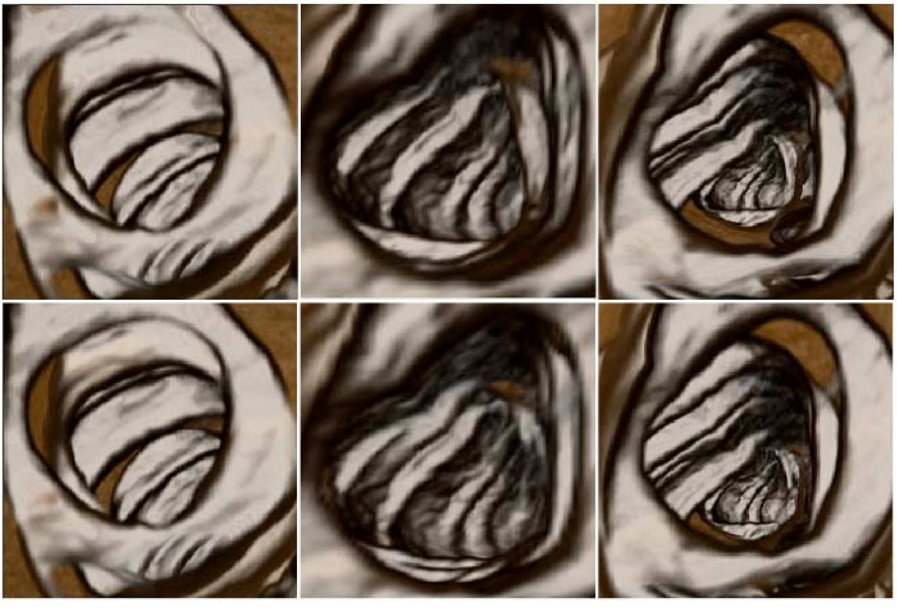


Fig. 9. Comparison of the quality of images rendered by the slab-based method (upper row) and our method (bottom row) in several positions in human colon.

4 Experimental Results

We compare the rendering time and image quality of the conventional slab-based method and our method. Both methods were implemented on a 1.7GHz Pentium IV PC with 1GB of main memory. We use VolumePro 500 model with 256MB voxel memory. The volume data used for the experiment is a clinical volume obtained by scanning a human abdomen with resolution $256 \times 256 \times 256$.

Figure 9 shows a comparison of the quality of image produced by both methods. The thickness of a slab and a subvolume (ΔZ_i) is 32 voxels, and FOV angles (θ and γ) are 30 degrees. Image resolution is 400×400 and it performs two-times super-sampling in the *z*-direction. Comparing the images generated by two methods, it is not easy to distinguish image quality.

Table 1 denotes the rendering time to get a final image in both methods. Sub-volume-based method is faster than slab-based approach. When we approximated perspective projection by using slabs or subvolumes, the smaller the thickness of slab or subvolume, the more realistic results we can get. However the rendering time is inversely proportional to the thickness. Table 1 shows that the processing time of subvolume-based method is shorter than that of slab-based method in any cases. However the difference of rendering time in-between two method becomes smaller as the thickness decreases. Therefore we have to choose appropriate value of thickness. According to experimental results, the thickness of 20~90 voxels is a proper selection to approximated perspective projection.

Table 1. Comparison of rendering time (FOV angle = 20°).

Thickness(voxels) / Method	256	120	90	70	60	40
A. Slab-based (msec)	117	156	167	197	234	313
B. Subvolume-based (msec)	47	78	117	153	198	280
B/A (%)	**59.9**	**50.1**	**29.8**	**22.3**	**15.4**	**10.5**

Thickness(voxels) / Method	30	20	10	5	2	1
A. Slab-based (msec)	392	553	1063	2086	5128	10124
B. Subvolume-based (msec)	353	514	1020	2042	5081	10063
B/A (%)	**10.0**	**7.2**	**4.0**	**2.1**	**0.9**	**0.6**

5 Conclusion

Since VolumePro hardware provides only orthographic projection, we cannot apply it to applications demanding perspective projection. Although some approaches to approximate perspective projection using parallelly projected slabs have been presented, it takes a lot of time since the entire region that does not belong to view frustum should be processed. In this paper we present an efficient method to approximate perspective projection using subvolume. To approximate perspective projection, we make several subvolumes with cropping feature of VolumePro. We conclude that our method is faster than slab-based method when we set the thickness as 20~90 voxels.

Acknowledgment. This work was supported by grant No. R05-2002-000-00512-0 from the Basic Research Program of the Korea Science & Engineering Foundation.

References

1. Yagel, R.: Volume Viewing: State of the Art Survey, SIGGRAPH 97 Course Note 31, (1997)
2. Pfister, H., Hardenbergh, J., Knittel, J., Lauer, H. and Seiler, L.: The VolumePro Real-Time Ray-Casting System, Proceedings of SIGGRAPH 99, Los Angeles, CA, (1999) 251-260
3. Vilanova, A., Wegenkittl, R., Konig, A. and Groller, E. : Mastering Perspective Projection through Parallelly Projected Slabs for Virtual Endoscopy, SCCG'01-Spring Conference on Computer Graphics, (2001) 287-295
4. K. Kreeger, W. Li, S. Lakare, and A. Kaufman: Perspective Virtual Endoscopy with VolumePro Parallel Rendering, http://www.cs.sunysb.edu/~vislab/, (2000)
5. Levoy, M.: Display of Surfaces from Volume Data, IEEE Computer Graphics and Applications, Vol. 8, No. 3, (1988) 29-37
6. Yagel, R. and Kaufman, A.: Template-based volume viewing, Computer Graphics Forum (Eurographics 92 Proceedings), Cambridge, UK, (1992) 153-167
7. Lacroute, P. and Levoy, M.: Fast Volume Rendering Using a Shear-Warp Factorization of the Viewing Transformation, Computer Graphics (SIGGRAPH 94 Proceedings), Orlando, Florida, (1994) 451-458
8. Westermann, R. and Eart, T.: Efficiently Using Graphics Hardware in Volume Rendering Applications, Computer Graphics, (1998) 167-177
9. Yagel, R., Kaufman, A., Cabral, B., Cam, N. and Foran, J.: Acclerated volume rendering and tomographic reconstruction using texture mapping hardware, Symposium on Volume Visualization, (1994) 91-97
10. Ma, K., Painter, J., Hansen, C. and Krogh, M.: A data distributed, parallel algorithm for ray-traced volume rendering, Proceedings of the 1993 Parallel Rendering Symposium, San Jose, CA, (1993) 15-22