# Fast Generation of Stereoscopic Virtual Environment Display Using P-buffer

Jun-Hyeok Heo, Soon-Ki Jung, and Kwang-Yun Wohn

## Abstract

This paper is concerned with an efficient generation of stereoscopic views for complex virtual environments by exploiting frame coherence in visibility. The basic idea is to keep visible polygons throughout the rendering process. P-buffer, a buffer of image size, holds the id of the visible polygon for each pixel. This contrasts to the frame buffer and the Z-buffer which hold the color information and the depth information, respectively. For the generation of a consecutive image, the position and the orientation of the visible polygons in the current view are updated according to the viewer's movements, and re-rendered on the current image under the assumption that, when the viewer moves slightly, the visibility of polygons remains unchanged. In the case of stereoscopic views, it may not introduce much difficulty when we render the right(left) image using visible polygons on the left(right) image only, The less difference in two images is, the easier the matching becomes in perceiving depth. Some psychophysical experiments have been conducted to support this claim. The computational complexity for generating a right(left) image from the previous left(right) image is bounded by the size of image space, and accordingly It is somewhat independent of the complexity of the 3-D scene.

## I. Introduction

Application of interactive 3-D graphics including virtual reality(VR) and interactive 3-D visualization must satisfy certain performance criteria independently of the complexity of the virtual world they deal with. In particular, the VR system should maintain a suitable update rate and respond to the user as quickly as possible. In a typical walk-through or fly-through application, the visual display must be updated at least 10 frames per second in order to produce the effect of contiguousness[1]. The system must also respond to the user command within 0.1 seconds in order for the user to maintain good continuous control. Further, the required performance in the visual display increases twice higher in generating stereoscopic images since we need to render all geometric primitives composing the virtual environments twice. Of course, with the advances in computer hardware technology there is a good chance for further reducing the processing time involved in the image generation. However, as the demand for more complex and more realistic virtual environments are simultaneously increased, it will become even more important to develop techniques that reduce the computing cost and meet the performance requirements with currently available hardware, at the cost of gracefully degrading the perceptual quality.

Although there have been substantial amount of research efforts devoted in improving the rendering performance for real-time graphics applications, there are surprisingly few published results that exploit the temporal coherence. To the best of the author's knowledge, there is no method for improving the performance of visualizing virtual environments in stereoscopic views which can be adopted to walkthrough systems based on any commercially available workstation. The reason seems to be partly because a time consuming complex algorithm on runtime to exploit temporal coherence can not meet the real-time requirement. As the stereoscopic pair can be considered as two temporally consecutive frames, the method using the frame coherence or the temporal coherence for enhancing the overall rendering performance may be used for efficiently generating stereoscopic views. A related approach can be found in ray tracing. Although ray tracing requires yet too much computing resource to be of any use for interactive real-time systems as a rendering scheme, ray tracing techniques are the

Manuscript received Oct. 14, 1996; accepted Feb. 3, 1998.

J. H Heo is with the Dept. of Computer Science at KAIST.

S. K. Jung is with the Dept. of Computer Science at University of Maryland as a research associate.

K. Y. Wohn is with the Dept. of Computer Science at Korea Advanced Institute of Science and Technology(KAIST) as an associate professor.

most similar ones to our proposed method in that they reduce the computing time for generating the new image by making use of some results from the previous computation.

In the proposed method, the next or consecutive images are generated from the visible polygons in the current image. Consequently, the computational complexity is bounded to the image space, rather than the number of polygons composing the virtual world which can be drastically increased for a more complex virtual world. It is apparent that the upper bound of the number of visible polygons is usually very small compared with the total number of polygons composing the complex virtual world. In case of stereoscopic views, it may not cause any problem to render the right(left) image with only visible polygons on the left(right) image, because the less different two images are, the more easily we can match to perceive depth. The proposed method is very simple and uses the z-buffer algorithm, so applications using an available graphic-special workstation having z-buffer hardware can adopt our method to generate stereoscopic views efficiently. In section 2, the previous work is briefly reviewed. Some more details of stereoscopic views generation are given in section 3. Our proposed method and some psychophysical experiments to support our approach are presented in section 4. In section 5, we compare the rendering performance by the ordinary method with the performance enhanced by our method. Section 6 gives the conclusion.

## II. Previous Work

Hubschman presented a method for exploiting the temporal coherence in the object space when rendering image sequences where only the camera moves and the objects are stationary, closed, convex and non-intersecting polyhedra[2]. These assumptions are too strict to apply to general objects composing virtual environments. Badt described a method of reprojection to generate frames by employing image-space coherence that reused the pixels from one frame of a diffuse-object animation to determine many of the pixels in the next frame[3]. As another related work, Schaufler presented a load-adaptive rendering algorithm which exploits the frame-to-frame coherence[4]. Rasterized images of objects are not discarded after each frame, but they are reused in subsequent frames as textures called imposter, as long as the imposter is considered valid. As a precursor to the work presented in this paper, we have proposed an efficient culling method by exploiting the frame coherence in visibility for walk/fly-through[5]. Mann uses this frame coherence in visibility to improve the performance of a walkthrough in remote virtual environments on network[6].

Stereoscopic image generation may be viewed as a special case of the generation of temporally consecutive image frames. A stereo pair may be thought of as two consecutive frames such that the viewer position is shifted slightly in the perpendicular direction to the gaze direction. Adelson *et al.* adapted Badt's method to the problem of image generation of ray-traced stereoscopic pairs[7] and solved many limitations of the reprojection method, and improved the method so that it can solve general case of generating ray-traced consecutive images[8]. Jevans[9] exploits the object space temporal coherence in generating ray-traced images. He assumes that the camera is fixed and objects are moving.

As previously mentioned, ray tracing is not suitable for VR systems. VR systems commonly use polygonal rendering methods such as flat, gouraud or phong. Adelson *et al.* published a non-ray-tracing algorithm which may be applicable to VR systems[10]. This algorithm clips and fills each polygon for the left view and the right views simultaneously and saves some computational time by exploiting that the two y positions of a polygon on two views are the same. But this method requires two z-buffers or one z-buffer and a BSP tree constructed after preprocessing the virtual environments for hidden surface elimination. Furthermore, the computational time to generate the second view is still dependent on the complexity of world or the total number of polygons composing virtual environments, which increases as the complexity of virtual world increases.

## III. Stereoscopic Views Generation

Human perceives the depth of an object using many kinds of information such as the thickness of crystalline lens called as accommodation, stereopsis by two eyes, relative motion of moving objects, and shapes and patterns of objects. When we see an object with two eyes, the projected position of it on the left retina is a slightly different from that on the right retina. This difference is called *disparity*. Disparity by stereopsis plays an important role in discriminating distance of two objects, it is necessary to provide the stereoscopic views for making virtual environments feel more realistic.

### *1. The two-centers-of-projection method*

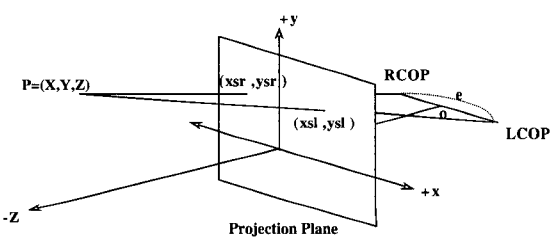



**Fig. 1.** Two centers of projection

The two-centers-of-projection method is known as the best method for generating stereoscopic views[11]. It assumes no vertical parallax. Therefore, when a point on the viewer space are projected on the image plane, the projected y values are equal, or *ysr-ysl=0* in Figure 1. The method uses two centers of projection. The left image is generated with a projection point *LCOP(Left Center of Projection)* and the right image is generated with another projection point *RCOP(Right Center of Projection)* shifted by *e* from *LCOP* on the viewer space. The x values, xsl and xsr, of the projected point with respect to *LCOP* and *RCOP* are given as follows:

$$xsl = f\frac{x-\frac{e}{2}}{Z} + \frac{ef}{2c} \qquad xsr = f\frac{x+\frac{e}{2}}{Z} - \frac{ef}{2c} \tag{1}$$

$$ysr = ysl = f\frac{y}{Z}$$

The value $f$ denotes the distance from the origin of viewer coordinate to the projection plane. The value $c$ denotes the distance to the focused object. The focused object has no disparity because it is projected to the same position on both retina. A value for $e$ should be carefully selected, after considering many factors, to generate more effective stereoscopic views. The binocular distance is one of those contributing factors. The main reason that the two-center-of-projection method is the most popular in stereoscopic views generation is the easiness in implementation with commercially available graphics workstation. All objects in virtual environment are first translated by the amount of *e/2* and then the ordinary perspective projection is performed. Finally the entire window is panned by the amount of *ef/2c*.

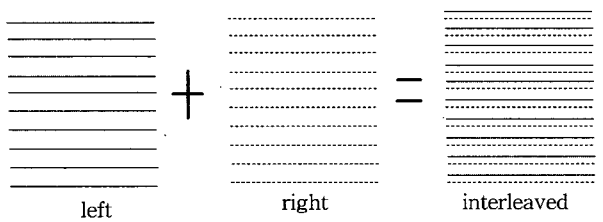



**Fig. 2.** Field Sequential method.

There are two methods for displaying stereoscopic image generated by the two-center-of-projection method. One is time-multiplexed method and the other is time-parallel method[11]. When time-multiplexed method is used, left and right images are displayed sequentially in time. In general, LCD shutter glasses are used to prevent left or right eye from seeing images for the other eye. In contrast to the time-multiplexed method, two images for both eyes can be displayed at the same time on two small LCD monitors mounted on, say, a head-mounted helmet. This method is called as time-parallel. In the time-parallel method, polarized glasses may be used. There are also two methods for storing stereoscopic images such as frame-sequential method and field-sequential method[12]. Frame-sequential method stores left and right images into two different frame buffers. Field-sequential method stores two images into one frame buffer by interleaving one image with the other as shown in Figure 2.

*2. Image warping*

As the stereoscopic pair may be considered as two consecutive frames in time, methods exploiting the frame coherence for enhancing the overall rendering performance may be used for generating stereoscopic views efficiently. We can enhance the rendering performance by generating the next image by warping the previous one according to the motion flow due to the viewer motion. When the viewer moves only slightly, the next image to be generated will be very similar to the current one. Using this similarity, the next image is generated quickly by moving the pixels on the previous image into the new position by motion flow, instead of being rendered through the full graphic pipeline. We call this approach *image warping*, and it is similar to Badt's reprojection method[4]. The processing time to generate a new image using the rendered image is dependent on the image space and independent of the world complexity.

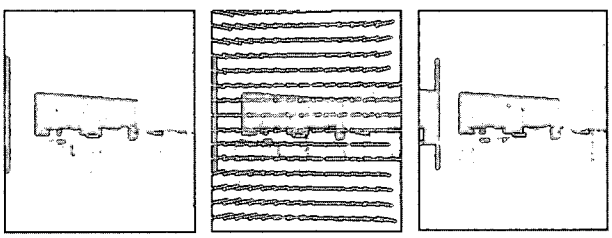



**Fig. 3.** (a) The original image, (b) motion flow, and (c) the image generated by the full graphic pipeline.

Figure 3(a) is the rendered image of an office room. When the viewer moves *20cm* to the left and pans 3 degrees around $y$ axis, the motion flow of pixels which presents the movement of pixels is depicted as shown in Figure 3(b). Figure 5(a) is the result of moving the pixels according to the motion flow. When the new image is generated on the image space, three problems arise[7].

The first problem is due to the overlapping pixels as shown in Figure 4(a). The left image is generated by image warping from the right image in all examples. When two pixels are warped into the same position, we have to decide one of them. This problem can be easily solved by choosing the visible pixel from the new position of viewer. In the case of stereoscopic views and generating the left image from the right image, scanning from the right to the left of the right

image and overwriting always results in correct visible pixels[7]. The second problem occurs when the holes which are unfilled appear on the new image as shown in Figure 4(b). Some pixels appear anew. Consequently, it is impossible to map them from any pixel on the previous image. The holes can be filled by interpolating nearby pixels, but this approach may result in an incorrect image. The last problem is due to bad pixels which must be invisible, but generated by shifting some pixels in the previous image(see Figure 4(c)). Mann uses a high-end graphics workstation server at a remote site to check these errors by computing the difference image between the correctly rendered image and the warped image[6]. When these errors become larger than the selected threshold, the remote server sends the necessary information to the client for building the new reference image. This technique can improve the performance of a walkthrough in remote virtual environments.

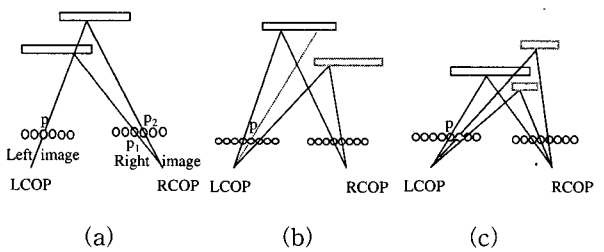



**Fig. 4.** (a) pixels p1 and p2 are overlapped, (b) no pixel moves into p(hole), and (c) bad pixel p.

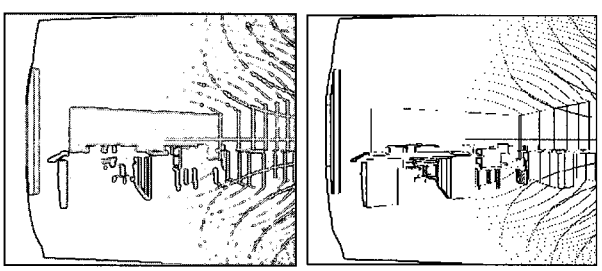


**Fig. 5.** Image Warping : (a) the generated image and (b) the difference image with the correct image.

Figure 3(c) is generated by rendering through the full graphic pipeline, and Figure 5(b) is the difference between two images generated by full graphic pipeline and by image warping.

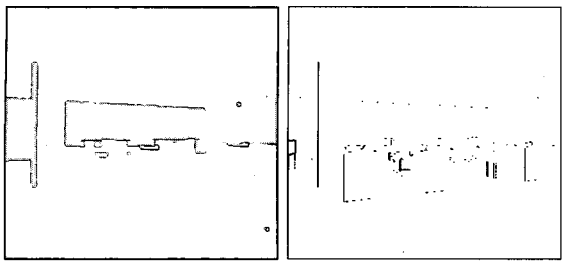
## IV. Efficient Stereoscopic Views Generation with P-Buffer

As mentioned in the previous section, the image warping technique results in holes and suffers from bad pixel problems. To alleviate these difficulties, we use bigger primitives such as polygons rather than pixels. Image warping method can be viewed as warping or moving visible pixels according to the motion flow. We can obtain the next image by moving only the visible polygons into new positions, and then re-rendering, rather than warping the pixels of the current image. When the viewer moves only slightly, there is a great chance of no polygon appearing anew. We may obtain the correct image using previously visible polygons only. Even when there are some newly visible polygons, the occupied portion on the image by these polygons will be very small. In other words, using frame coherence in visibility, we may obtain a perceptually better image than the image by image warping. The computation time for generating the next image is bounded by the image space, similarly to the image warping method.

```
Procedure p-buffer
var
    pz, pid : integer
begin
for all polygons set the visible-flag FALSE
for y := 0 to YMAX do
    for x := 0 to XMAX do begin
        WriteZ(x,y,0)
        WritePixel(x,y,BACKGROUND_VALUE)
        WritePolyID(x,y,0)
    end
for each polygon do begin
    pid := polygon's id
    for each pixel in polygon's projection do begin
        pz := polygon's Z-value at pixel coordinate (x,y)
        if pz >= ReadZ(x,y) then
            WriteZ(x,y,pz)
            WritePixel(x,y,pixel's color)
            WritePolyID(x,y,pid)
        end
    end
end
for each pixel in the image space do begin
    pid := a polygon's id at pixel coordinate (x,y)
    set the visible-flag of pid TRUE
end
end
```

**Fig. 6.** The p-buffer algorithm.

As mentioned above, we consider only the polygons seen in the previous frame for generating the next image. The problem of deciding which polygons are visible on the image is a classic one in computer graphics. The z-buffer algorithm, among many algorithms for visibility, is famous for its efficiency, simplicity and the availability with hardware



(a) (b)

**Fig. 7.** P-buffer method : (a) the generated image and (b) the difference image with the correct image.

support. We have developed the modified z-buffer algorithm which utilizes a buffer that stores the identification of visible polygons for each pixel. This buffer is also updated when the z-test is successful and the frame buffer is updated with a new color value. We call this buffer the *p-buffer*. Similar approaches were taken previously by Weghorst *et al.* for improving the computation in ray tracing with *item buffer* [13], and by Cohen *et al.* for calculating form-factors in their hemi-cube method [14]. Figure 6 is the pseudo code of the modified z-buffer algorithm with the p-buffer, the bold face lines are added to the existing z-buffer algorithm excerpted from Foley *et al.* [15]. A visible flag is associated with each polygon, and the visible flag of each visible polygon on the rendered image is set to *TRUE*. When each polygon is scan-converted, the "polygon id" is saved in each pixel of the p-buffer. After rendering procedure, the visible flags are updated by the p-buffer. Figure 7(a) depicts the image generated with p-buffer method from Figure 3(a), and the difference between two images (generated with p-buffer method and by full graphic pipeline) is contrasted in Figure 7(b).

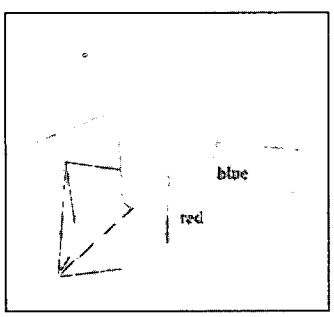
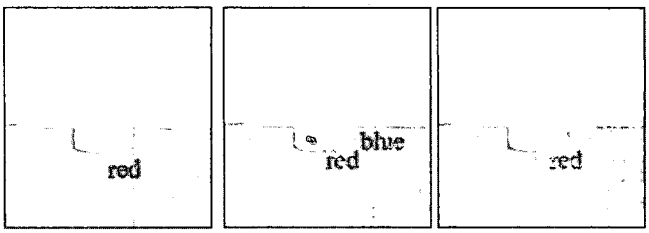



**Fig. 8.** Experiment 1.

If we generate right(left) image with only visible polygons on left(right) image found by p-buffer method, we may miss some newly visible polygons on right(left) image. However, it is likely that, in the case of stereoscopic views, newly visible polygons in the right view which were not seen in the left view can be rather annoying to the viewer who is trying to perceive the depth of unmatched polygons. The depth perception of stereoscopic views is due to the binocular disparity of perceived object in both eyes[16]. It is proposed as a heuristics for displaying stereo pairs that left and right images should match as closely as possible in terms of brightness, focus, size and colors[17].




(a) (b) (c)

**Fig. 9.** (a) Left image (b) Right image with the blue cylinder (c) Right image without the blue cylinder.

We did some psychophysical experiments to explore this factor. We can classify newly visible polygons into two classes by their colors or patterns with respect to surrounding polygons. Newly visible polygons may have different color or pattern from surrounding polygons or else they have the same color or pattern with surrounding polygons. In the first case the rendered image without newly visible polygons may rather make it easy to perceive objects by mitigating the difference of two images and in the second case missing newly visible polygons may cause some problems by exposing polygons with different colors or patterns.

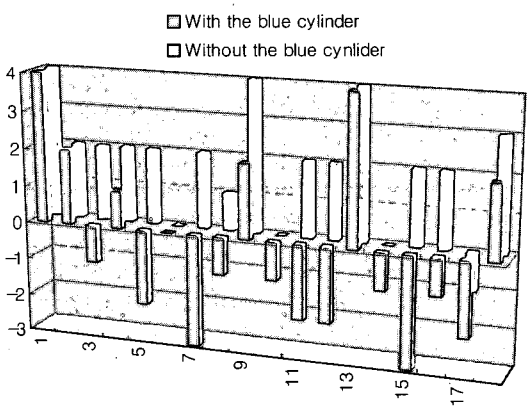



**Fig. 10.** The result of experiment 1.

In the first experiment, we presented two stereoscopic images of a scene like Figure 8 to eighteen subjects. The scene has four objects - a red cylinder, a blue cylinder, and a rectangular wall and the floor with a pattern. The blue

cylinder is located behind of the red cylinder in such a way that it can be seen from the right eye only. Thus, it is a newly visible polygon whose color is different from surrounding polygons of the red cylinder and the wall. Two stereoscopic images were displayed for each subject. One is generated with the blue cylinder as in Figure 9(b) and the other is generated without it as shown in Figure 9(c).

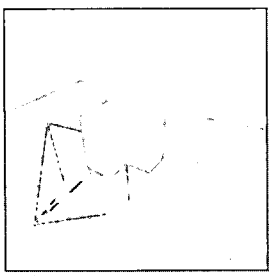
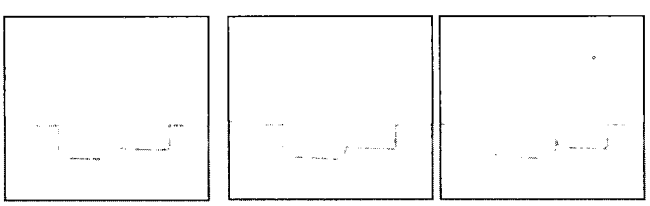


**Fig. 11.** Experiment 2.

To remove some psychophysical side-effect in the order two stereoscopic images are displayed, to a half of the subjects we show the image with the blue cylinder first and to the other subjects we show the image without it first. After looking at two stereoscopic images sequentially, each subject was requested to answer the question about the degree of easiness. The degree of easiness is scaled from -4(very hard to see) to 4(very easy to see). Figure 10 summarizes the experimental result. The X axis of the graph represents each subject and the Y axis represents the degree of easiness. The average degree of easiness in the case with the blue cylinder is -0.222222 and the average degree in the case without it is 1.833333. The standard deviations are 2.211083 and 1.424574, respectively. With these statistical values, we performed t-test with the null hypothesis that *both degree of easiness in the case with the blue cylinder and without it are the same.* As a result, the null hypothesis is rejected and an alternate hypothesis - *the degree of easiness in the case without the blue cylinder is higher than the degree of easiness in the case with the blue cylinder* - is accepted if a confidential interval below 99.8769% is used.



(a) (b) (c)

**Fig. 12.** (a) Left image (b) Right image with the polygon at center (c) Right image without the polygon at center.

In the second experiment, we gave two stereoscopic images of a scene such as Figure 11 to eighteen subjects. The scene has three objects - a blue concave solid and a rectangular wall and the floor with a pattern. The concave solid is located in such a way that its center can be seen from the right eye only. Thus, it is a newly visible polygon whose color is the same as the surrounding polygons of the concave solid.

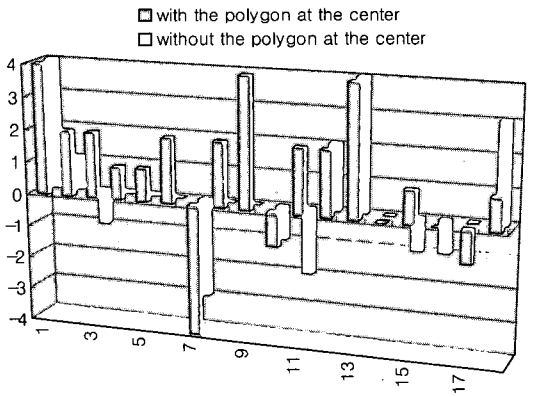



**Fig. 13.** The result of experiment 2.

Two stereoscopic images were displayed to each subject. One is generated with the polygon at the center as in Figure 12(b) and the other is generated without as in Figure 12(c). Figure 13 shows the experiment result. The average degree of easiness in the case with the polygon at the center is 1.222222 and the average degree of easiness in the case without it is 0.277778. The standard deviations are 1.986885 and 1.903729, respectively. With these statistical values, we performed the t-test with the null hypothesis that *both degree of easiness are same.* As the result, the null hypothesis is accepted if a significance level below 15.4537% is used.

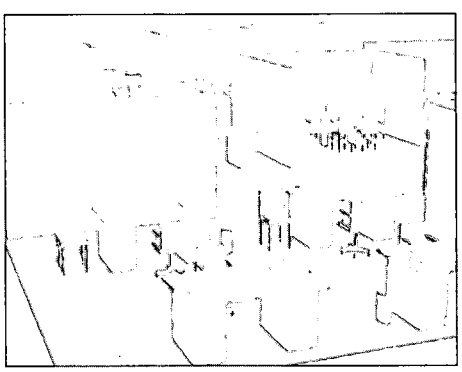


**Fig. 14.** Office model.

As the consequence of performed experiments, we can claim that it does not cause any serious problem in human perception by generating stereoscopic views using visible polygons by one eye only. To make sure that this fact holds

in more general environments such as architectural walkthroughs, we performed another experiment.

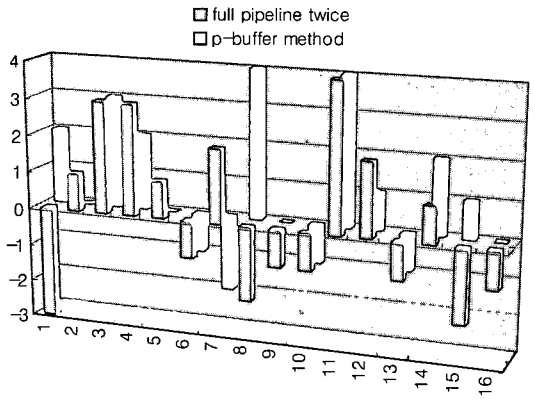
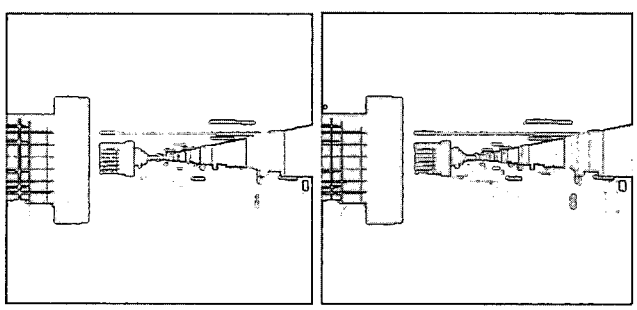



**Fig. 15.** Walkthrough experiment.

Stereoscopic images generated during the walkthrough along a predefined path in an office floor of Figure 14 are displayed to sixteen subjects. Two walkthrough sequences were shown to subjects. One is generated by rendering all the polygons twice for both eyes, and the other is generated by p-buffer method; the left image is generated by rendering all the polygons with p-buffer and the right image is generated by rendering only visible polygons previously found with p-buffer. The frame rate was maintained at 5.6 frames per second for both sequences. After looking at both sequences, the subjects answered the question about the degree of easiness in watching each sequence.



(a) (b)

**Fig. 16.** The left image (a) is generated from the right image (b) by using p-buffer method.

The result is demonstrated in Figure 15. The average degree of easiness for the sequence generated by applying the full-pipeline method twice is 0.3125 and the average degree of easiness for the sequence generated with the p-buffer method is 0.875. The standard deviations are 2.08866 and 1.821172, respectively.

With these statistical values, we performed the t-test with the null hypothesis that *both degree of easiness are same*. As the result, the null hypothesis is accepted if a significance level below 42.3338% is used. The result consistently supports the previous claim that it does not cause any serious problem in human perception by generating stereoscopic views using visible polygons by one eye only.

## V. Experiments

In this section we would like to show that the p-buffer method can be used to generate stereoscopic views very efficiently by presenting several experimental results. All the experiments were carried out on SUN Sparc10 with 64M memory. The simplest way to generate stereoscopic views is to render twice the entire graphics primitives through the full graphic pipeline for both views. It is easy to realize that the p-buffer method significantly reduces the computation time of generating another view because it considers the visible polygons on the first view only. We used the office model as shown in Figure 3(a) for experiments after removing walls and adding posts to make it somewhat open. The stereoscopic views were generated by processing the ordinary full graphic pipeline twice and the p-buffer method while increasing the virtual world complexity by copying the office room from one time to fifteen time, and we measured the computation time for rendering the left image and the right image.

**Table 1.** Result of experiments.

| polygons | visible polygons | full pipeline (seconds) | | p-buffer method (seconds) | |
|---|---|---|---|---|---|
| | | right | left | right | left |
| 3652 | 301 | 0.67 | 0.67 | 1.14 | 0.39 |
| 7308 | 474 | 1.00 | 0.99 | 1.70 | 0.45 |
| 10966 | 536 | 1.04 | 1.04 | 1.94 | 0.48 |
| 14624 | 569 | 1.27 | 1.26 | 2.09 | 0.39 |
| 18280 | 584 | 1.43 | 1.42 | 2.35 | 0.43 |
| 21932 | 604 | 1.59 | 1.58 | 2.48 | 0.37 |
| 25590 | 617 | 1.77 | 1.76 | 2.50 | 0.33 |
| 29248 | 623 | 2.01 | 1.99 | 2.60 | 0.41 |
| 32906 | 626 | 2.10 | 2.06 | 2.91 | 0.37 |
| 36564 | 632 | 2.36 | 2.36 | 3.12 | 0.43 |
| 40220 | 634 | 2.48 | 2.48 | 3.15 | 0.37 |
| 43878 | 636 | 2.53 | 2.52 | 3.35 | 0.44 |
| 47536 | 638 | 2.80 | 2.77 | 3.56 | 0.36 |
| 51194 | 638 | 2.96 | 2.85 | 3.62 | 0.37 |
| 54852 | 640 | 3.17 | 3.15 | 3.79 | 0.37 |

The image is 300 by 300 pixels in size. For the experiments, we implemented a rendering routine in which the z-buffer algorithm (in software) was modified to have the p-buffer and we used the constant shading algorithm. Note that, by using the *polygon id* as the color value, the p-buffer method can be easily adopted to conventional graphic workstations with hardware z-buffer and the read operation of

the frame buffer. The left and right images of fifteen office rooms, the most complex case, are shown in Figure 16. The result of the experiments is given in Table 1.

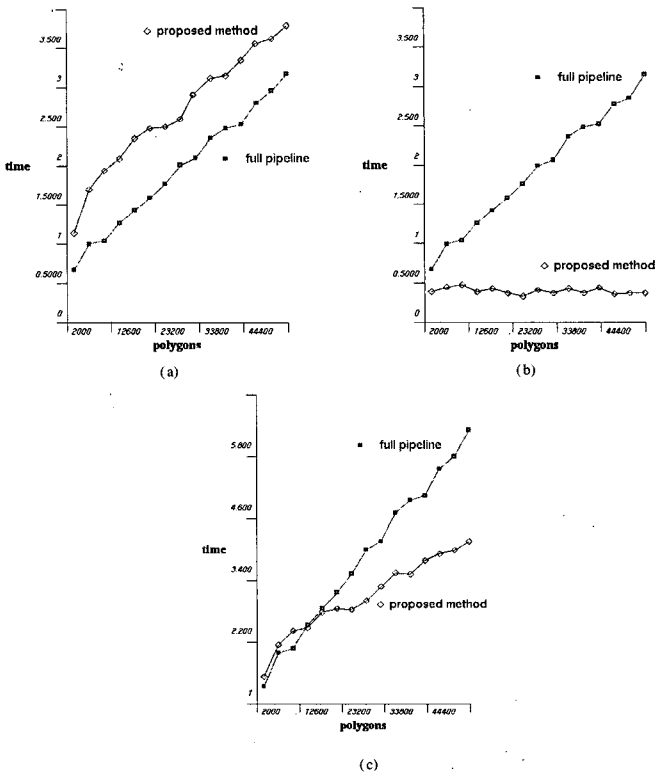



**Fig. 17.** Computation time to generate the right view(a), the left view(b), and both views(c), respectively.

*Polygons* in the table indicates the number of scan converted polygons, and *visible polygons* is the number of visible polygons on the right view. As expected, the time for generating the left view by the p-buffer method is independent of the complexity of the virtual world because the number of visible polygons does not increase with the complexity of the virtual environments. Figure 17(a) shows the computation time of both methods to generate the right view. The p-buffer method took more time than the full pipeline method because it maintains the additional p-buffer and updated visible flags using the p-buffer. If the p-buffer were implemented in a hardware such as the hardware z-buffer, the gap will become negligible. Figure 17(b) shows the computation time to generate the second view. The p-buffer method maintains some level of independence from the virtual world complexity as compared with the method using the full graphic pipeline. Figure 17(c) shows the computation time to generate both views. In the case of the virtual world having more than 15,000 polygons, the proposed p-buffer method outperforms the method using full pipeline. The gap increases with the virtual world complexity.

## VI. Conclusion

In this paper, we proposed a method to generate stereoscopic views efficiently. The images for one of both eyes are generated using the p-buffer which keeps the identification of visible polygons, and then the images for the other eye are generated from the polygons previously visible(i.e., stored in the p-buffer). The proposed method generates images for the other eye within the time bounded by the image space independent of the complexity of the virtual world. We performed psychophysical experiments to support the fact that the depth perception of stereoscopic views is affected with only the polygons visible from both eyes. Consequently, the proposed p-buffer method for stereoscopic view generation can be applied without causing any serious problem in practice. We also showed that the proposed p-buffer method is more efficient than the conventional ones, while successfully maintaining a display quality sufficiently good enough for the human binocular depth perception.

## VII. Acknowledgement

This research was supported in part by Agency for Defense Development through Automatic Control Research Center, Center for Artificial Intelligence Research, and Science and Technology Policy Institute. We specially thanks to the anonymous reviewer for his or her very helpful comments on the content and the english.

## References

[1] S. Bryson, "Designing a Virtual Reality Application," *Developing Advanced Virtual Reality Applications, SIGGRAPH '94 Course Notes* 2, pp. 4.1-4.21, 1994.

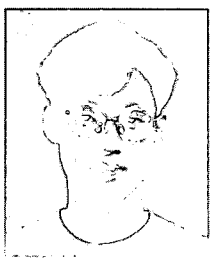
[2] H. Hubschman and S.W. Zucker, "Frame to Frame Coherence and the Hidden Surface Computation: Constraints for a Convex World," *ACM Trans. on Graphics*, 1(2), pp. 129-162, 1982.

[3] S.J. Badt, "Two Algorithms Taking Advantage of Temporal Coherence in Ray Tracing," *The Visual Computer, 4(3)*, pp. 123-132, 1988.

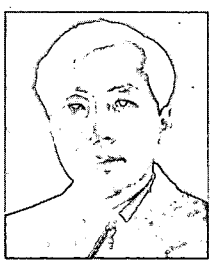
[4] G. Schaufler, "Exploiting Frame-to-Frame Coherence in a Virtual Reality System," Proc. of IEEE Virtual Reality Annual International Symposium (VRAIS'96), pp. 95-102, 1996.

[5] J. Heo, S. Jung and K. Wohn, "Exploiting Temporally Coherent Visibility for Accelerated Walkthroughs," *An International Journal of Computers & Graphics*, pp.

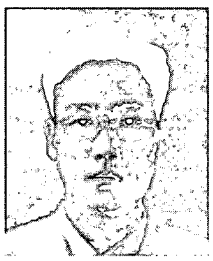
507-517, 21(4), 1997, Elsevier Science.
[6] Y. Mann and D. Cohen-Or, "Selective Pixel Transmission for Navigating in Remote Virtual Environments," *Proc. of Eurographics '97*, pp. C201-C206, 1997.
[7] S.J. Adelson and L.F. Hodges, "Stereoscopic Ray-Tracing," *The Visual Computer, 10(3)*, pp. 127-144, 1993.
[8] S.J. Adelson and L.F. Hodges, "Generating Exact Ray-Traced Animation Frames by Reprojection," *IEEE Computer Graphics and Applications, 15(3)*, pp. 43-52, 1995 May.
[9] D. Jevans, "Object Space Temporal Coherence for Ray Tracing," *Proc. of Graphics Interface '92*, pp. 176-183, 1992.
[10] S.J. Adelson, J.B. Bentley, and I.S. Chong, "Simultaneous Generation of Stereographic Views," *Computer Graphics Forum, Vol 10*, pp. 3-10, 1991.
[11] L.F. Hodges, "Tutorial: Time-Multiplexed Stereoscopic Computer Graphics," *IEEE Computer Graphics & Applications*, pp. 20-30, March 1992.
[12] Virtual i-O, "Virtual i-O i-glasses! Stereoscopic 3D Graphics and Head-tracker Development Tools," *i-glasses!$_{TM}$ Developer Kit Version 1.2*, pp. D1-D2, 1995.
[13] H. Weghorst, G.J. Hooper and D.P. Greenberg, "Improved Computational Methods for Ray Tracing," *ACM Trans. on Graphics*, 3(1), pp. 52-69, 1984.
[14] M.F. Cohen and D.P. Greenberg, "The Hemi-cube: A Radiosity Solution for Complex Environments," *Proc. of ACM SIGGRAPH '85*, pp. 31-40, 1985.
[15] J.D. Foley, A. vanDam, S.K. Feiner, and J.F. Hughes, *Computer Graphics,* Principles and Practice, 2nd Edition, pp. 669, Addison-Wesley Reading, Massachusetts, 1990.
[16] J.M. Foley, "Stereoscopic Distance Perception," *Pictorial communication in virtual and real environments,* 2nd Edition, S.R. Ellis, M.K. Kaiser, and A.J. Grunwald (eds.), pp. 558-566, Taylor & Francis Ltd., Washington, 1993.
[17] L. Harrison, D. McAllister, and M. Dulberg, *Stereo Computer Graphics for Virtual Reality*, ACM SIGGRAPH'97 Course Note 4, 1997.



**Jun-Hyeok Heo** received the B.S degree from the Dept. of Computer Engineering, Kyungpook National University in 1992 and the M.S. degree from the Dept. of Computer Science, Korea Advanced Institute of Science and Technology(KAIST) in 1994. Since 1994, he has been working towards the Ph.D. degree in the Dept. of Computer Science at KAIST. His current interests include interactive 3D graphics and virtual reality. He has also been interested in the use of multimedia methods in computerized education.



**Kwang-Yun Wohn** received the B.S. degree from the Dept. of Applied Physics, Seoul National University, Korea, in 1974, and the M.S. degree from the Dept. of Computer Science, the University of Wisconsin, Madison. in 1981, and the Ph.D. degree form the Dept. of Computer Science, the University of University of Maryland, College Park in 1984. From 1984 to 1986, he was a lecturer at the Division of Applied Science, Harard University. From 1986 to 1990, he was an assistant professor at the Dept. of Computer Information Science, the University of Pennsylvania. Since 1990, he has been with the Dept. of Computer Science at Korea Advanced Institute of Science and Technology KAIST) as an associate professor. His research activities are in the areas of virtual reality, human-computer interaction and culture computing.



**Soon-Ki Jung** received the B.S degree from the Dept. of Computer Engineering, Kyungpook National University in 1990. He received the M.S. degree and the Ph.D. degree from the Dept. of Computer Science, Korea Advanced Institute of Science and Technology (KAIST) in 1992 and 1997 respectively. Since 1997, he has been a research associate in University of Maryland, College Park. His current interests include computer graphics, computer vision and virtual reality.