Real-time image-based rendering for stereo views of vegetation

Jitendra A. Borse^{*}, David F. McAllister^{**}

Department of Computer Science, North Carolina State University

ABSTRACT

Rendering of detailed vegetation for real-time applications has always been difficult because of the high polygon count in 3D models. Generating correctly warped images for nonplanar projection surfaces often requires even higher degrees of tessellation. Generating left and right eye views for stereo would further reduce the frame rate since information for one eye view cannot be used to redraw the vegetation for the other eye view. We describe an image based rendering approach that is a modification of an algorithm for monoscopic rendering of vegetation proposed by Aleks Jakulin¹. The Jakulin algorithm pre-renders vegetation models from 6 viewpoints; rendering from an arbitrary viewpoint is achieved by compositing the nearest two slicings. Slices are alpha blended as the user changes viewing positions. The blending produces visual artifacts that are not distracting in a monoscopic environment but are very distracting in a stereo environment. We have modified the algorithm so it displays all pre-rendered images simultaneously and slicings are partitioned and rendered in a back-to-front order. This approach improves the quality of the stereo, maintains the basic appearance of the vegetation and reduces visual artifacts but it increases rendering time slightly and produces a rendering that is not totally faithful to the original vegetation model.

Keywords: image-based rendering, vegetation rendering, trees

1. INTRODUCTION

The high polygon content of 3D models of vegetation makes real time rendering difficult. Indeed, the motivation to study this problem arose from suffering from the vegetation rendering speed in a hemispherical projection environment. To try to attain real-time frame rates, very simple tree models were required for a low altitude flight simulation. Lollipop-like models were used for trees but the resulting scene realism was not acceptable. In addition, the frame rate in this hemispherical dome environment dropped drastically when even simple 3D models of vegetation were introduced in a scene.

Given these complexities we need to look at alternatives to using 3D models directly. Image-based rendering (or representation) has recently gained popularity and is currently a topic of active research. The main advantage of using image-based rendering methods over 3D models is that it is computationally less expensive and rendering time is usually constant and not dependent on object complexity. Thus, higher frame rates can be attained.

This paper makes an attempt to explore how a slicing technique proposed by Jakulin can be modified to generate stereoscopic images in real-time and how to eliminate or reduce the effect of certain visual artifacts which are not very evident in monocular viewing but become obvious in stereoscopic views. Our goal is to produce acceptable stereo images of forests of vegetation in real time and to produce views that appear "natural" in a walk-through environment. The images reproduced by our modified algorithm maintain the "treeness" property of the vegetation but they are not faithful to the original 3D model. The current implementation requires the viewer to remain approximately same height from the ground.

All stereo images are presented for cross viewing². Left and right eye images have been produced using off axis perspective projection. The model used as an example for this paper is shown in Figure I below. The performance statistics are based on a Pentium IV 1.7GHz processor and a NVIDIA GeForce2 GTS graphics card.

Figure 1: Original Tree Model

2. PREVIOUS WORK

Billboarding of vegetation is a popular technique in representation of trees in real-time applications. Orientation of the polygon based on the view direction is called billboarding and the polygon is called a billboard³. The billboard rotates around an axis and aligns itself so as to face the viewer as much as possible. Two variants of this method were tried by the authors and are described later.

Chen and Williams⁴ have investigated smooth interpolation between views but they use pre-computed correspondence maps, which will be very difficult to use for vegetation given its complexity. Max and Ohsaki⁵ modified this approach and used z-buffer views instead of images used by Chen and Williams. Z-buffer views are more suitable for finely divided objects like trees, where depth coherence at adjacent pixels is limited.

Others have investigated methods using image warping for intermediate view generation but most require some additional information. The 3D image warping method proposed by William, McMillan and Bishop⁶ require per-pixel disparity. The authors describe two approaches for reconstruction. First, they treat each reference pixel separately varying the size of the reconstruction kernel depending on disparity and normal vector orientation of the reference pixel. In a second approach, they treat the reference frame as a mesh. The compositing step warps two reference frames and combines them to generate the single frame. Another method uses layered impostors⁷ but it focuses more on representation. It represents solid objects as layers and number of such layers for a single view may be as high as 64. For an observer near a viewpoint closer to the sampled viewpoint, a single image might be sufficient. However to avoid holes, multiple images can be warped to fit the desired view. Schaufler recommends sampling images from about 20 viewpoints evenly distributed over a sphere.

Aleks Jakulin proposed a method called *slicing* that uses alpha-blended textured polygons. Pre-rendered images, called *slices*, are generated from six different viewpoints. Depending upon the angle of the observer, the two closest viewpoints among the six are chosen and appropriately blended. This method is very simple and able to generate frames at an interactive rate. We investigate this method and suggest appropriate modifications to overcome visual artifacts that result when rendering in stereo.

3. OUR APPROACHES

3.1. Billboarding

Our first attempt was the natural one where we billboard both the left and right eye views of a given orientation of a tree. While this technique works quite well and enables very fast rendering, it is limited by the fact that the viewer always sees the same view of the vegetation, regardless of his position. However, when many trees must be rendered in the same frame then the same 3D model can be used; we rotate the model to produce different views of the same tree. These multiple views can be billboarded producing the appearance of a forest. Transparency, motion parallax and vegetation density help to reduce the sense that the same tree is being used in every instance as the viewer changes position. An example of a billboarded forest where the same tree is used in every case is given in Figure II.

Figure II: A stereo forest using billboarding

A variation of billboarding was also tried to see if the technique might be used to produce continuously changing views of a tree while the viewer was in motion. The tree was pre-rendered from 60 viewpoints placed along a circle around tree, separated by 6 degrees, one image generated for each viewpoint. For every frame, the two images closest in angle to the current viewpoint were determined and blended to produce the view of the vegetation. However, the blending of billboards produced unacceptable ghosting or shadowing effects and this method was abandoned.

3.2. Slicing

To obtain different views of a tree as the viewer changes position, we modified the Jakulin slicing method described above. Consistent with the original algorithm, a slice is a planar layer, represented with an ordinary alpha or color-keyed texture. These textures are stored along with alpha values and pixels are drawn or rejected based on this alpha value. Unlike the original algorithm, blending is disabled for drawing textures so that each pixel drawn is completely opaque (i.e. alpha value is either 0 or 1). Pixels are assigned to planes based on their depth. A set of such parallel and equidistant slices is called a *slicing*. All the slices in a slicing are generated from same viewpoint. Our algorithm uses five slices per slicing.

A bounding box is defined to enclose the entire vegetation. For any generic viewpoint, the bounding box is oriented such that the top and the bottom planes of the box are parallel to the ground and the viewing direction perpendicular to the front face of the bounding box. A vector from the viewpoint to the trunk of the tree and parallel to ground defines the viewing direction. Six planes parallel and equidistant with dimensions the same as the front plane (the plane visible

from the viewpoint), are placed so that the first and last plane coincide with the front and the back face of the bounding box. Any two adjacent planes are treated as near and far clipping planes and the hidden surfaces in between them, with respect to the viewer, are eliminated. The visible surfaces that remain are projected orthographically on the back-clipping plane. The complexity of vegetation enables us to use only five layers and convey the necessary depth information.

In the modified algorithm, we split each plane vertically into two halves. We call a half plane a slice. The left halves form one slicing and right halves form another. From 3 viewpoints, distributed on a circle, 60 degrees apart and at same distance above ground, 6 such slicings are obtained. The center of this circle lies on a vertical axis V , through the centroid of the object (which is same for each slicing in our case). Each slice has same height and width. Also the distance between adjacent slices is same for every slicing. Figure III a is a top view of the bounding box; vegetation between the planes is projected on the rear plane of each adjacent pair. Figure III b shows a perspective view of the slices and how they are partitioned.

Figure III a

Figure III b

Figure III: Generation of slices.

The original algorithm determines slicings closest and second closest to the viewpoint and called primary and secondary slicings respectively. These are then displayed varying their individual levels of transparency as the viewpoint changes. The transparency of the slicing is proportional to the closeness of the slicing to the viewpoint.

This approach introduces some visual artifacts that are not acceptable in stereo. As an observer moves around the tree, primary and secondary slicings are swapped. This changes the order in which slicings are drawn and therefore, when there is a swapping, leaf and branches that were in front in previous frame appear to be at the back and vice-versa. Secondly, an alpha or blending value varies between 0 and 1. When a slicing is emerging (alpha increasing), parts of vegetation appear as if moving out of fog. Similarly, when a slicing fades out (alpha decreasing) those corresponding parts of the vegetation begin to disappear, a visual phenomenon that is very distracting in stereo. See figure IV.

Figure IV: Alpha blending of slices

To solve the problem of parts appearing and disappearing, all the slicings are displayed and alpha blending is eliminated. Initially we expected that when any slicing becomes parallel to viewing direction the planar nature of the images would be easily seen. However, if a tree is sufficiently dense this anomaly is not visible. Each of 6 slicing becomes parallel to the viewing direction once during rotation around the tree and this occurs only for 3 to 4 degrees for every slicing which would be only 2-3 frames. See figure V. The portion of the vegetation inside the circle in the left image is approximately perpendicular to the viewer in the right image (shown by circles).

Figure V: Nearly perpendicular slices.

If the images generated are not halved during pre-processing and used directly, branches and leaves supposed to be behind appear to be in front. This happens because, if complete planes were used, the slicings would intersect each other and the parts of vegetation supposed to be behind are drawn over those that are meant to be in front. Depth relationships are reversed. This doesn't create a problem in monocular viewing because there is no depth conveyed.

The modified algorithm splits the planes to obtain 6 slicings from 3 viewpoints. The slicings are then displayed in back to front order. Also, every slice of each slicing is drawn in back to front order relative to the viewpoint.

As shown in figure VI below, we number slices from 1 to 6 in counterclockwise direction. To generate the view from point A, the order would be 2-1-3-6-4-5 and from point C it would be 6-5-1-4-2-3. Also note, when drawing slicing 3 viewed from point C the order of slices would be e-d-c-b-a and from point A it would be a-b-c-d-e.

Figure VI: Order of rendering

The slicing and elimination of alpha blending produce a *doubling* effect. There are branches and parts of the vegetation that are present in two or three slicings. Since all slices are displayed, these parts would appear more than once. See figure VII for the rendered tree using our modified algorithm and compare it with the original model in Figure I.

Figure VII: Stereo views using the modified algorithm.

Another inherent defect in the algorithm is discontinuities in geometry created by the slicing. Different parts of a single branch may be projected on different planes of the same slicing. Figure VIII illustrates this. A branch with this property has been highlighted. Fortunately, this aberration usually goes unnoticed due to the dense nature and random complexity of vegetation.

Figure VIII: Discontinuities caused by the algorithm

4. IMPLEMENTATION AND RESULTS

The image-based rendering method presented in this paper was implemented using the OpenGL API. All vegetation models were rendered in 3D Studio MAX. The slicing procedure and partitioning to maintain the correct depth relationships results in 30 polygons to be rendered per tree per eye view in each frame.

The following results were obtained:

Resolution 512x512	
Number of trees	Approximate frames per second
1	28
3	28
4	21
5	19
7	15
10	13

Resolution 256x256	
Number of trees	Approximate frames per second
1	28
7	28
10	27
20	26
40	23
50	21

Table I: Frame rate for different number of trees and different texture sizes.

In the case of billboarding, only a single polygon for each eye view as required. Hence, we would expect a performance improvement of approximately 30 to 1 over the slicing method, modulo overhead caused by OS communication and graphics pipeline bottlenecks.

The same implementation was executed on a computer with Pentium III 1.0 GHz processor and 3dlabs Oxygen GVX1 Pro graphics accelerator. In this case, with only one tree in the scene only 8 frames per second were possible for 512x512 textures and 15 frames per second for 256x256. The textures must be cropped to minimum possible size to optimize the performance.

5. SUMMARY AND CONCLUSIONS

If the viewer's attention is focused on a single 3D model, then, in order to simulate realism in immersive walk-throughs, it is important to be able to produce views of the model that change continuously with the position of the viewer. The slicing method described above provides this capability. However, if presenting multiple trees in an immersive scene, the viewer's attention will normally wander and the vegetation complexity, motion parallax and transparency available in billboarding is normally sufficient to present a realistic stereo scene in an immersive walk-through. The viewer is less likely to notice that he is always presented with the same view of a given tree or different views of the same tree. In this case, because the number of polygons per tree is much lower, one can attain much higher frame rates and reduce the chance of flicker.

For both billboarding and slicing, the stereo rendering problem is easily partitionable in a multiple processor environment in a straightforward way. Hence, one would expect acceptable frame rates in this case, even with large numbers of trees in the scene. This is an area for future research.

We note that the current algorithms are not able to render correct views of the vegetation if the viewpoint lies above the highest point on the vegetation. Efficient stereo rendering of vegetation in real-time from arbitrary viewing points is still an outstanding problem. We seek solutions for this problem.

ACKNOWLEDGEMENTS

We wish to thank Aleks Jakulin for his help in implementing the original algorithm and offering advice.

REFERENCES

1. Aleks Jakulin, "Interactive Vegetation Rendering with Slicing and Blending," *Eurographics 2000* (short presentations).
2. D.F. McAllister (Editor), *Stereo Computer Graphics and Other True 3D Technologies*, p. 5,26, Princeton University Press, Princeton NJ, October 1993.
3. McReynolds, Tom, D.Blythe, B.Grantham and S.Nelson, "Programming with OpenGL: Advanced Techniques," course 17 notes at *SIGGRAPH'98*, 1998.
4. S.E.Chen and L.Williams, "View interpolation for image synthesis," *Proc. ACM SIGGRAPH 93*, pp. 79-288, 1993.
5. N.Max and K.Ohsaki, "Rendering trees from precomputed Z-buffer views," *Eurographics Workshop on Rendering 1996*, pp. 165-174, 1996.
6. M. William, L. McMillan and G. Bishop, "Post-Rendering 3D Warping," *Proc. of the 1997 Symposium on Interactive 3D Graphics*, pp. 7-16, 1997.
7. G.Schaufler, "Image-based object representation by layered impostors," *ACM symposium on Virtual Reality Software and Technology '98*, pp. 99-104, 1998.

* jaborse@unity.ncsu.edu; bjitu@yahoo.com; phone 1 919 833-7067; P.O.Box 8206/226 Withers Hall, Raleigh, NC, 27695; ** mcallist@csc.ncsu.edu; phone 1 919 233-8164; fax 1 919 233-8578; Dept. of Computer Science, North Carolina State University, Raleigh, NC 27695-8206