Real-Time Rendering of Realistic Trees in Mixed Reality

Alberto Candussi, Tobias Höllerer, Nicola Candussi
University of California, Santa Barbara
alberto@fluidinteractive.com, {holl, nicola}@cs.ucsb.edu

Abstract

Mixed Reality applications put very high demands on both the visual realism and the rendering times of computer graphics elements that are to be perceived as part of the physical scene. This paper presents novel techniques to render photorealistic trees in real-time Mixed Reality. Animation of the tree branches leads to a realistic effect of the tree swaying in the wind. To enhance the effect of blending the tree into a video texture, we present three levels of real-time filtering of the tree and its shadow, which has a great impact on the perceived realism.

1. Introduction

We are considering the problem of rendering trees realistically in a real-time mixed or augmented reality environment. Fig. 1 shows an example image of our main application. In the center of the image the user has placed a virtual tree that is larger than (and thereby mostly occluding) a smaller physical tree. This paper discusses the real-time rendering and blending and filtering techniques necessary to achieve such photorealistic results. We are currently integrating this code with our mobile outdoor augmented reality framework [1]. The main contribution to the AR/MR community is two-fold: real-time interactive tree rendering techniques yielding previously unrealized levels of realism and the discussion of real-time, hardware-accelerated blending and filtering techniques that drastically improve the visual realism of our integration of virtual and physical imagery.

Our work relates to videogame applications; most of them use a non-billboarded approach for leaves rendering and, where level-of-detail control is used, a single billboard representing the whole plant [5] is used for far views, making the rendering particularly flat.

Jakulin [2] approximates the entire tree by using a variable number of textured slices, giving a good visual appearance for distant objects, it is still not convincing for close-up images. Deussen et al. [4] use point and line rendering for plants far away from the viewer. Interactivity is reached only for a relatively small number of trees.

2. Matching the Real World

A virtual tree matching the real world must be fully customizable through an editor and it must have convincing lighting and animation systems. An efficient filtering technique is critical to achieve highly believable results, especially in MR.

Figure 1: The tree in the center foreground of this image was procedurally generated and rendered into the scene in real time.

2.1. Realistic Lighting and Shadows

Trunk and major branches are rendered using bump mapping and shadow mapping while minor branches use per vertex lighting and shadow mapping. Leaf clusters are rendered as transparent billboards. Small variations to size and color are applied to hide repetition. Lighting is obtained as a sum of four factors: per vertex lighting, density factor, shadow map factor, and occlusion factor.

Per vertex lighting is computed assuming the vertex normals of the leaves are oriented along the radius of a sphere centered with the tree's bounding box. This way, leaves on the side of the tree that is closer to the light are brighter than leaves on the opposite side. A density factor takes into account the number of leaves surrounding the current leaf and is pre-computed. A shadow map factor is the result of the shadow map test on the current leaf. An occlusion factor determines the intensity of the projected shadow based on the occluder's depth in the shadow map.

In the generation of the shadow map, obtained with perspective shadow mapping technique [3], billboards are rendered facing the light. Even if not physically correct, the resulting shadows appear realistic, because the sheer number of real leaves in a tree and the variance in leaf orientation makes it impossible for a human to anticipate the correct shadow perspective.

2.2. Realistic Animation

When a tree is being built all the animation properties (such as branch flexibility) are pre-computed. At run-time

Figure 2, a) no filtering, b) bilinear filtering, c) alpha channel filtering, d) combined filtering.

the tip of every branch is displaced and the rest of the branch movement is done by interpolating this transformation based on each vertex's distance to the tip.
The displacement is computed as the sum of two vectors, one along the wind direction and the other one orthogonal to it. The first vector in general has a greater magnitude than the second, which gives the appearance that the tree is swaying along the wind direction. The second vector is used to add some noise to the movement. At each frame, periodically changing factors are added to these vectors, in order to modify their magnitude and direction.
Leaves are animated depending on wind strength and direction; billboards perform an oscillation around the viewing axis, based on sine waves.

	GeForceFX 5950 Ultra	GeForce 6800 GT
No Filtering	80 fps	150 fps
Biliniear	70 fps	140 fps
Alpha	25 fps	100 fps
Combined	15 fps	70 fps

2.3. Real-Time Filtering

Three different filtering techniques have been used: bilinear filtering (Fig 2b), alpha channel filtering (Fig 2c), and the two techniques combined (Fig 2d).
With bilinear filtering, the background image is rendered into an auxiliary texture without any filtering; the virtual environment is then rendered on the same auxiliary texture, which is eventually rendered with bilinear filtering on screen. The auxiliary texture has a resolution slightly smaller than the one of the screen.
With alpha channel filtering, only the virtual environment is rendered onto the auxiliary texture, whose empty pixels are set to transparent and all other pixels set to opaque; the background image is then rendered on screen. Then we draw the auxiliary texture on screen (i.e. over the background image) using a fragment shader which computes the transparency of every pixel as the average of its alpha and of the eight neighbor pixels;
The last technique is a combination of bilinear and alpha channel filtering: the whole scene is rendered with alpha channel filtering on the auxiliary texture which is then rendered with bilinear filtering on screen.

3. Results

Table 1 shows the performances with different filtering techniques of the scene in Fig. 1, rendered with 1280x1024 screen resolution and composed by 4032 triangles for roots, trunk and major branches, 376 triangles for minor branches, 9354 triangles for leaves.
Several people, unaware of our project, have been asked if they noticed anything unusual in Fig. 1; if the virtual tree was identified at all, it never happened immediately. Without filtering, rendered trees were readily identified.

4. Conclusions and Future Work

It has been shown that with this approach it is possible to render highly realistic trees and blend them into real scenes using filtering in real-time. Filtering enhances the level of realism by a great amount. We presented 3 hardware-accelerated techniques that can be applied to MR scenes without sacrificing too much rendering speed.
Future work includes an interactive tree editor, automatic (image-based) determination of parameters and 3D modeling of the physical scene for correct shadow cast.

5. References

[1] R. Bane and T. Höllerer, 2004: Interactive tools for virtual x-ray vision in mobile augmented reality. In *Proc. ISMAR 2004 (IEEE/ACM Int. Symp. on Mixed and Augmented Reality)*, Arlington, VA, Nov. 2-5 2004.

[2] A. Jakulin, 2000: Interactive Vegetation Rendering with Slicing and Blending. Eurographics 2000 short paper, Interlaken, Switzerland.

[3] Marc Stamminger and George Drettakis, 2002: Perspective shadow maps. In Proc. ACM SIGGRAPH '02, San Antonio, Texas, 557-562

[4] O. Deussen, C. Colditz, M. Stamminger and G. Drettakis, 2002: Interactive visualization of complex plant ecosystems. In Proc. IEEE Visualization '02, pp. 219-226.

[5] Iris performer programmer's guide, 1995