Real-time Rendering of Dynamic Clouds (sap_0159)

Yuki Shimada  Mikio Shinya  Michio Shiraishi  Takahiro Harada
Promethech Software, Inc.  Toho University  Toho University  The University of Tokyo

1 Introduction
This paper presents a fast cloud rendering method for dynamic scenes, where cloud shapes and lighting environments dynamically change. Although the Harris method [Harris and Lastra 2001] has been widely used for static cloud rendering, it can be fatally slow for real-time applications when, for example, light directions change. By introducing the 3D attenuation buffer and re-arranging the algorithm, we improved the rendering speeds of dynamic clouds by a factor of 10-100 times. The image quality is also improved due to a finer representation of light distribution.

2 Harris Method
In the Harris method, the density distribution of a cloud is modeled as a set of particles, each of which represents a Gaussian density distribution, processed as a textured polygon in the GPU.

The method consists of a pre-processing step and a rendering process. Pre-processing calculates the intensity distribution of the direct light and the rendering process refers to the intensity distribution created by pre-processing when generating cloud images.

In the pre-process step, the cloud particles are first sorted along the light direction. Then, in the near-to-far order, the light intensity distribution is calculated by drawing a particle as a textured polygon onto the frame buffer with alpha blending. Since the buffer is overwritten when other particles overlap, it is read back to the CPU every time when drawn. The value at the center of the particle is then stored in main memory, which is referred to in the rendering process. Because read-back is very slow, this data transfer is the bottleneck of the entire process. The sorting process also slows execution speeds. Moreover, since only one value is stored per particle, intensity gradation within particle is neglected, making the cloud appear flat.

The rendering process is conducted in a similar way but without read-back, and is much faster than pre-processing.

3 Proposed Method
The proposed method introduces a 3D texture to avoid read-back. To avoid the sorting process, we calculate the ‘attenuation ratio distribution’, instead of light intensity itself. Let us consider the example shown in Figure 1. In the figures, the incident light with intensity $I_0$ illuminates two particles along the y-axis in the voxel space. Each particle attenuates the light and creates a cylindrical “shadow polygon”. Point $x_1$ is within the shadow polygon of Particle 1, and the intensity is $G_1 \times I_0$, where $G_1$ indicates the transmittance of the particle. Point $x_2$ is in the shadow polygon of both Particle 1 and 2, and the intensity there is $(G_1 \times G_2) \times I_0$. Note that this is a simple scalar product and particle processing order can be arbitrary.

This computation can be performed in the following way: for each particle, calculate the shadow polygon, and render it to a 3D texture, called the attenuation buffer, with the transmittance value of the particle. As mentioned before, the results do not depend on particle order, so sorting is not required. The resulting attenuation buffer is the intensity distribution itself, which can be directly referred to in the rendering process. Therefore, read-back is also unnecessary. Due to current GPU specs, we implemented multiplicative blending onto a frame buffer by additive blending in log-scale.

4 Experiments

Table 1: Comparison of CPU time. (voxel resolution).

<table>
<thead>
<tr>
<th></th>
<th>9382</th>
<th>32914</th>
<th>60134</th>
<th>particles</th>
</tr>
</thead>
<tbody>
<tr>
<td>Harris method</td>
<td>0.721 sec</td>
<td>3.164 sec</td>
<td>5.934 sec</td>
<td></td>
</tr>
<tr>
<td>Proposed (384°)</td>
<td>0.038</td>
<td>0.154</td>
<td>0.161</td>
<td></td>
</tr>
<tr>
<td>(128°)</td>
<td>0.007</td>
<td>0.015</td>
<td>0.016</td>
<td></td>
</tr>
</tbody>
</table>

We measured the time taken by the pre-process step on a PC with Geforce8800GTX, Intel Core 2 Duo 1.83GHz. Table 1 shows a comparison of the original Harris method and the proposed method. As shown in the table, the improvement in speed was significant: the proposed method was 10-100 times faster than the previous method. Figure 2 demonstrates improvements on image quality. Although the previous method failed to capture the detailed features of the clouds, the proposed method successfully generates delicate gradation inside a cloud.

5 Conclusions
This paper presented a fast rendering method for dynamic clouds. Significant improvements over the Harris method were made by utilizing the attenuation buffer and resolving the bottleneck caused by GPU-CPU read-back. The proposed method allows real-time rendering of dynamic scenes with moving clouds and light sources. The image quality was also improved by its finer representation of light distribution.

References