Abstract

This paper presents a real-time technique for rendering 3D meshes in the pencil drawing style. We analyze the characteristics of pencil drawing and incorporate them into the rendering process, which is fully implemented on a GPU. Rendering examples demonstrate the high performance of the proposed technique in terms of speed and quality.

Introduction

A pencil is one of the most available and easy-to-handle tools for drawing. Pencil drawings can represent soft impressions of objects with pencil strokes, where the exact tones of strokes are determined by the properties of pencil and paper.

A simulation-based approach was proposed to mimic the characteristics of graphite pencils combined with paper effects [Sousa and Buchanan 1999]. There exists an image-based approach to obtain pencil drawings which transforms an input image into the pencil drawing style [Mao et al. 2001]. Pen-and-ink illustration can be considered similar to pencil drawing in that the tones, materials, and shapes of objects are mostly represented by strokes. Excellent techniques were developed to generate pen-and-ink illustrations from 3D objects [Winkenbach and Salesin 1994; Webb et al. 2002]. For stylization of 3D objects contours, Mohr and Gleicher [2001] used jittering to mimic the multiple contours in a pencil drawing. Loviscach [2004] demonstrated that multiple contour drawing can be supported by graphics hardware.

In this paper, we propose a real-time pencil rendering technique, which produces pencil drawing style images of an input 3D mesh in real time. For object contours, we propose a multiple contour drawing technique that imitates trial-and-errors of human in contour drawing. For interior shading, we present a simple approach for mapping oriented textures onto an object surface. To improve the quality of pencil rendering, we generate and map pencil textures that reflect the properties of graphite pencils and paper. In our pencil rendering technique, most of the components are implemented on a GPU. Consequently, the rendering speed is quite fast; 15 to 60 frames per second for reasonably complicated meshes.

Algorithm

In a drawing, objects are usually depicted by contours and interior shading. In our system for pencil rendering of a 3D mesh, the run-time process consists of contour drawing and interior shading parts, whose output images are combined to generate the final image (see Figure 1).

In contour drawing, we first render the input mesh to obtain normal and depth images from the shape. The normal and depth images are then used to detect contours in image space [Isenberg et al. 2003; DeCarlo et al. 2003]. To make the extracted contours look natural, similar to pencil drawing by a human, we draw the contours a few times with slightly distorted trajectories. The image-based contour detection is performed by a pixel shader. After contours have been obtained, another pixel shader handles multiple contour drawing and pencil texture mapping at the same time. Figure 2 shows an example of the contour drawing.

The interior shading part starts with intensity computation for the input mesh to determine the brightness of pixels in the rendered image. To introduce pencil drawing tones, we adjust the contrast of pixel brightness, similar to charcoal rendering [Majumder and Gopi 2002]. To express pencil strokes, we use texture rotation and 3-way blending for mapping oriented pencil textures in the principal curvature directions. To represent interaction among a pencil and paper, we model paper as a height field and the paper normals are used to control the pixel intensities during the mapping of oriented pencil textures. In interior shading, all steps are performed in a single pixel shader, except that a simple vertex shader is used to project the 3D principal curvature directions at vertices onto the 2D image space. Figure 3 shows an interior shading result.

After contour drawing and interior shading, the resulting images are composed and enhanced to produce the final image. This step is fully performed in a pixel shader using the images from contour drawing and interior shading as texture images mapped onto the
In the preprocessing step, we prepare several data that will be used in the run-time process. For contour drawing, we generate a texture to be used for giving pencil tones to the contours. We also construct distorted planes to be used for modifying contour trajectories. For interior shading, we generate a set of pencil textures to represent different intensities of an object interior. The principal curvature directions are computed for mesh vertices to determine the orientations in interior texture mapping. A paper model is constructed as a height field and the normal vector at each pixel is computed to provide paper effects in interior texture mapping. Once all necessary data have been prepared in preprocessing, the run-time process for pencil rendering of a 3D mesh is fully supported by graphics hardware and performed very fast with varying camera and lighting parameters.

Experimental Results

Figures 4 and 5 show examples of pencil rendering produced by our technique. The overlapped contours with slightly different trajectories show natural results. The interior shapes of the given objects are nicely described with appropriate effects of pencil drawing. Table 1 summarizes the rendering speed for the examples shown in this paper.

<table>
<thead>
<tr>
<th>Figure</th>
<th>Model</th>
<th># polygon</th>
<th>Avg. FPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 3</td>
<td>Globe</td>
<td>9,728</td>
<td>47</td>
</tr>
<tr>
<td>Figure 4</td>
<td>Drill</td>
<td>41,210</td>
<td>16</td>
</tr>
<tr>
<td>Figure 5</td>
<td>Camera</td>
<td>1,034</td>
<td>62.5</td>
</tr>
<tr>
<td>Figure 5</td>
<td>Lamp</td>
<td>12,676</td>
<td>45</td>
</tr>
</tbody>
</table>

Table 1: Rendering performance

References


