GPU-Based Optimization of a Free-Viewpoint Video System

Neal Orman[1,2], Hansung Kim[1], Ryuuki Sakamoto[1], Tomoji Toriyama[1],
Kiyoshi Kogure[1], and Robert Lindeman[2]

[1] Knowledge Science Lab, ATR
Kyoto, Japan +81-774-95-1401
{hskim, skmt, toriyama, kogure}@atr.jp

[2] Department of Computer Science
Worcester Polytechnic Institute
Worcester, MA +1-508-831-5000
{nealo, gogo}@wpi.edu

Abstract ***We present a method for optimizing the reconstruction and rendering of 3D objects from multiple images utilizing consumer-level graphics hardware. We accelerate visual hull reconstruction by parallelizing a shape-from-silhouette algorithm. Rendering is optimized through the application of geometry shaders to generate billboarding microfacets textured with captured images. Experimental results show that the resulting system is significantly more efficient than a purely CPU-based approach, while maintaining graphical quality.***

I. System Overview

Our free-viewpoint video system, called 'Cinematized Reality', takes images from standard video cameras and constructs a 3D model of the objects contained within, from which it renders video of the scene from virtual cameras from novel viewpoints. This is accomplished by segmenting the captured images to extract foreground silhouettes and using these silhouettes to construct a voxel-based model of the object. The original images are then used to texture this model during the rendering process [1].

The voxel model is constructed from the segmented images by using each camera's parameters to project the silhouettes into a voxel grid. Any voxel lying within the volume contained within the projection of all the silhouettes is considered filled . If, however, a given voxel lies within the background region of any of the projected silhouettes, then it is marked as empty.

This voxel model is then rendered by a microfacet billboarding process which generates a single quad for each filled voxel, oriented toward the viewpoint. Each microfacet is textured with the original images captured by the surrounding cameras. Primary camera selection for texturing is determined based on the angle at which the scene is being viewed. Then, we check for occlusion using a modified version of Bresenham's line drawing algorithm [2]. If an occlusion is detected, we select the next best camera.

II. System Optimization

The voxel reconstruction and rendering parts of the pipeline were optimized using custom shaders. The modified system begins by loading a single frame of images and segmentation masks into texture memory. These textures are used by a fragment shader which uses the cameras' parameters to render to a 3D texture that represents the voxel model of the scene. The fragment shader uses the camera parameters to determine the texture coordinate of the segmentation mask to check for each camera in order to determine if the voxel is filled.

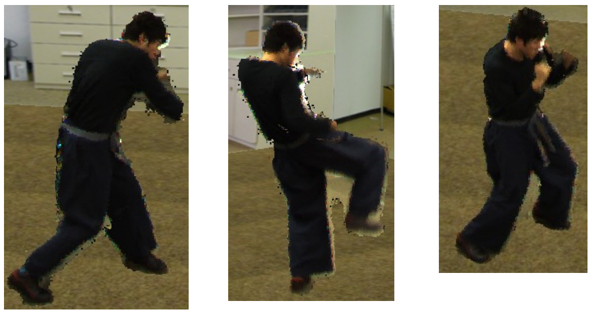
Fig. 1. Frames rendered from karate dataset

This voxel model is rendered using geometry shaders to generate the microfacets. Each voxel is passed to a geometry shader which generates the four vertices of the microfacet for each solid voxel. No geometry is generated for empty voxels. Occlusion detection is performed by rendering this voxel model as a depth texture from the perspective of each camera. If the distance from the selected camera to the voxel being rendered is greater than the distance stored in the depth texture, then the voxel is occluded and another camera is selected. Finally, the microfacet geometry shader is used to render each microfacet with the appropriate texture, calculating the texture coordnates from the camera parameters.

III. Results

Initial testing results in a substantial (> 100x) performance increase over the original CPU-based version of the voxel reconstruction and rendering algorithms on commercially available hardware. The primary reason for the performance improvement is the use of the GPU as a parallel processing architecture. This allows the GPU to process multiple voxels simultaneously, where the CPU is able to process only one at a time. Furthermore, many of the processor-intensive operations involved in the projections are hardware accelerated on the GPU.

References

[1] H. Kim, R. Sakamoto, I. Kitahara, N. Orman, T. Toriyama and K. Kogure, "Compensated Visual Hull for Defective Segmentation and Occlusion," Proc. ICAT, pp.210-217, 2007.

[2] J. Bresenham, Algorithm for Computer Control of a Digital Plotter .*IBM Systems Journal* 4, 1 (1965), 25-30.