Real-Time Soft Shadows with Cone Culling

Louis Bavoil [1,2]   Cláudio T. Silva [1]

[1] Scientific Computing and Imaging Institute, University of Utah
[2] Sony Computer Entertainment America, Graphics R&D

Figure 1: (a) Wireframe model rendered using our real-time algorithm entirely on a GPU (GeForce 7800 GTX) at 15.2 fps. Image resolution: 800×600. Shadow map resolution: 1024^2. Num. samples per pixel: 88. (b) Opaque model rendered using the same technique at 15.8 fps. (c) Using a CPU implementation of our algorithm (5.0 s). (d) Using stochastic ray tracing with 1,000 shadow rays per pixel (7.38 min).

Abstract

We have developed a new physically-based technique for computing soft shadows from a shadow map for spherical lights. Shadow-map samples unprojected into world-space (micro-patches) are culled against a shadow cone wrapping all the possible shadow rays from the light to the surface location. Because of this volumetric test, the algorithm handles self-shadowing more robustly than traditional shadow mapping. The amount of shadowing contributed by each culled micro-patch is approximated by the solid angle subtended by the bounding sphere of the micro-patch over the solid angle of the light source. We describe a real-time implementation running entirely on a GPU. For each image pixel to shadow, the algorithm projects the pixel onto the near plane of the light frustum, and looks up occluding micro-patches in a dynamic disk kernel. To guarantee constant frame rates, the kernel is sampled using a fixed number of samples per pixel. Our results compare favorably to ray-tracing for images with no overlapping shadows.

1 Introduction

Interactive rendering of soft shadows on dynamic scenes is a challenging problem. Silhouette-based techniques such as soft shadow volumes can render accurate soft shadows in real-time, but they do not work with complex occluders such as wireframe objects. Another solution is to use stochastic ray tracing. However, for complex scenes, it requires hundreds of shadow rays per pixel to remove all noise artifacts. Therefore, ray-traced soft shadows are still not interactive on commodity hardware.

In our method, we reverse the problem of ray tracing by sampling the scene in a first pass using rasterization as in traditional shadow mapping. This produces an image of sample hit points as seen from the light (micro-patches). Micro-patches are shadow-map pixels, unprojected into world space. Hence, they are rectangles parallel to the image plane of the light frustum, as in Atty et. al.'s soft shadow maps [Atty et al. 2006].

Thus, our approach works in two steps: culling points contained in a shadow cone, and summing normalized solid angles. The idea of summing solid angles is similar to Bunnel [2005]. However, our input is a shadow map rather than precomputed surface elements. For performance and robustness, we approximate the micro-patches by their centers when doing the cone culling, and we approximate the micro-patches by their bounding sphere for the solid angles.

2 Overview

Our algorithm can be implemented entirely on modern GPUs, using floating-point textures and Cg. For every frame, the GPU performs the following passes. First, the scene is rendered from the eye. Colors and world-space positions are stored into multiple floating-point textures. Second, the scene is rendered from the light, and the radii of the bounding spheres of the micro-patches are stored in the alpha coordinates.

Shadows are computed from the eye map (world-space positions and colors) and the shadow map (world-space positions and radii). The shadow map is sampled in a dynamic disk kernel, and the normalized solid angles of occluding micro-patches are added together. The kernel is a Poisson distribution with a Gaussian density. Finally, the pixel color is multiplied by the total shadow intensity.

For debugging purposes, we have also developed a CPU-based version of the shadowing step using an octree of the 3D points stored in the shadow map. The octree is used to cull the points in a given shadow cone. In this version, solid angles are computed from the actual rectangular micro-patches.

3 Discussion

As every shadow mapping method, our algorithm has aliasing issues. We make self-shadowing more robust by storing the shadow map in world space. Similar to the technique of Atty et. al. [2006], our algorithm has light leaking issues in overlapping shadows. In addition, our shadow umbra are overestimated because of the bounding spheres. However, our algorithm produces plausible results with increased realism compared to other shadow mapping methods, and it can run in real-time entirely on a GPU.

4 Acknowledgments

We thank NSF, DOE, IBM, and ARO for their financial support.

References

ATTY, L., HOLZSCHUCH, N., LAPIERRE, M., HASENFRATZ, J.-M., HANSEN, C., AND SILLION, F. 2006. Soft shadow maps: Efficient sampling of light source visibility. *Computer Graphics Forum*. (to appear).

BUNNELL, M. 2005. Dynamic ambient occlusion and indirect lighting. In *GPU Gems 2*, M. Pharr, Ed. Addison Wesley, ch. 14, 223–233.