Fast Soft Shadow by Depth Peeling

Xuehui Liu[†*] Xiaoguang Hao[†] Mengcheng Huang[†] Fang Liu[†] Mingquan Zhou[‡] Hanqiu Sun[♯] En-Hua Wu[‡§]
State Key Lab of Computer Sciences, Institute of Software, Chinese Academy of Sciences[†]
Beijing Normal University[‡] Chinese University of Hongkong[♯] University of Macau[§]

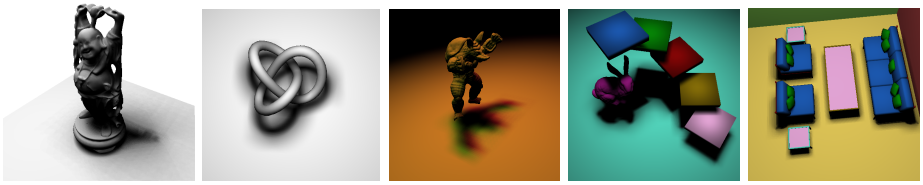

Figure 1: *Soft shadows rendered by our algorithm for several scenes.*

1 Introduction

Soft shadow generation is a challenging problem in realistic rendering. Previous methods using shadow map or shadow volume work well for point light sources but are difficult to be extended to area lights. This paper presents a new method for fast soft shadow generation under dynamic area light sources. Our algorithm encodes the depth distribution of the scene into a coarse depth grid in a preliminary pass from the light point of view. In the second pass, the scene is rendered from the camera viewpoint to capture the frontmost layer. During deferred shading, the area light is sampled and the irradiance of each shaded pixel is accumulated along the ray. Experimental results demonstrate high quality soft shadows with interactive performance for dynamic scenes and lighting.

2 Our Approach

We assume the objects in the scene to be water-tight and non-intersected with each other so that the ray intersects the objects in pairs. The algorithm starts by setting the view point to the center of the area light. The depth range of each pixel is divided into 16 subintervals uniformly to form a depth grid. We utilize 8 MRT buffers and initialize them to 0. Each two consecutive MRT channels are bind into pairs as described in [Liu et al. 2009]. Within each subinterval, fragments on the front faces will update the first channel, while those on the back faces will update the second channel using MAX blending. After the geometry pass, the depth values of the furthest fragment within each subinterval can be restored from the channel pair and negated if it is on a back face. In the second pass, the scene is rendered from the camera viewpoint to obtain the shaded points and re-project them to the light space.

$z_{i,p}$=0	$z_{j,p} > 0$	Occluded	
	$z_{j,p} < 0$	Non-occluded	
	$z_{j,p} = 0$	Non-occluded	
$z_{i,p} > 0$	$z < \|z_{i,p}\|$	$z_{j,p} > 0$	Occluded
		$z_{j,p} < 0$	Non-occluded
		$z_{j,p} = 0$	Non-occluded
	$z > \|z_{i,p}\|$	Occluded	
$z_{i,p} < 0$	$z < \|z_{i,p}\|$	Occluded	
	$z > \|z_{i,p}\|$	Non-occluded	

Table 1: *Visibility calculation for current subinterval*

*China Basic S&T 973 Research Grant(2009CB320802), National 863 High-Tec Grant(2008AA01Z301), NSFC(60573155)&UM Research Grant.

SIGGRAPH 2010, Los Angeles, California, July 25 – 29, 2010.
ISBN 978-1-4503-0210-4/10/0007

Scene	Triangles	Light Samples	Ray Samples	FPS
Budda	1.087M	10X10	10	17.2
		10X10	20	8.1
Torus-knot	2.8K	10X10	10	23.1
		10X10	20	11.7
Armadillo	213K	10X10	10	16.2
		10X10	20	13.0
Bunny with stairway	69K	10X10	30	6.7
		20X20	30	1.7
Room with furniture	3.5K	10X10	50	2.2
		20X20	50	0.5

Table 2: *Performance of our algorithm on the test scenes.*

In deferred shading, the area light is sampled and the visibility term between each light sample and each shaded point is calculated according to the depth grid. We march along the ray from a shaded point to a light sample at a fixed step. For a ray sample with a depth value z, suppose it is in the i^{th}subinterval on pixel location p in the depth grid, and the captured depth value in that grid cell is $z_{i,p}$. The algorithm searches for the first non-zero subinterval j along the ray towards the light. If there is no such an interval, the ray is non-occluded. Otherwise, according to $z_{i,p}$ and $z_{j,p}$, it is easy to determine whether the ray has been occluded or not as summarized in Table 1. The visibility term can be set to be zero if the ray is occluded or one if not. The irradiance of each shaded point can be accumulated by the product of the intensity of each light sample with its visibility term to generate realistic soft shadows.

3 Results

Table 2 shows the performance of our algorithm with different number of light and ray samples at 512x512 on a commodity PC of Intel Duo Core 2.4G Hz with 3GB memory, and NVIDIA Geforce 280 GTX. The computation for visibility term dominates the algorithm running time, which could be accelerated by taking advantage of the coherency between adjacent pixels in the future.

References

LIU, F., HUANG, M.-C., LIU, X.-H., AND WU, E.-H. 2009. Efficient depth peeling via bucket sort. In *Proceedings of the Conference on High Performance Graphics 2009*, ACM, New York, NY, USA, 51–57.