Advanced 3D Computer Graphics - IMA904/IG3DA - Master2 2021-22

Variance Soft Shadow Mapping

Kewei XU - kewei.xu@etu.sorbonne-universite.fr

February 11, 2022

Abstract

Variance soft shadow mapping (VSSM) [5] is a soft shadow rendering technique that was proposed in 2010. It is still widely used in many real-time rendering fields due to its high efficiency and more realistic rendering effect. This report reproduces VSSM and compares it with some other shadow rendering techniques before it like percentage-closer filtering (PCF) [2] and percentage-closer soft shadows (PCSS) [1]. Understand and analyze the development process of shadow rendering technology, compare the advantages and disadvantages of different shadow rendering technologies.

Keywords
Shadow rendering, Real-time rendering

1 Introduction

Basically, this report will be divided into three main parts. The first part will introduce some basic and classical shadow rendering techniques that laid the foundation for variance soft shadow mapping (VSSM) [5], the difficulties they solved and the problems they have. The second part will focus on the VSSM shadow rendering technique, the problems it solves, and include some of the difficulties encountered in reproducing it and how they were solved. The last part will compare the test results of different shadow rendering techniques and analyze their advantages and disadvantages as well as directions for improvement.

2 Related Work

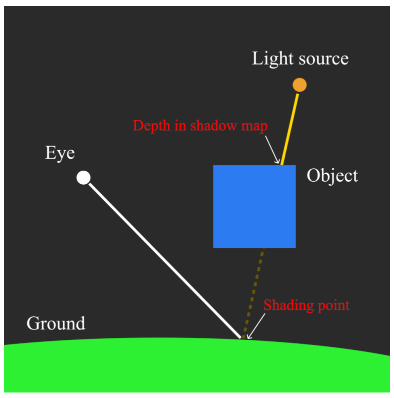
In this section, I will cover the basic shadow rendering techniques, as well as shadow mapping [4], percentage-closer filtering (PCF) [2] and percentage-closer soft shadows (PCSS) [1], how they evolved and how they laid the foundation for variance soft shadow mapping (VSSM) [5].

2.1 Shadow Mapping

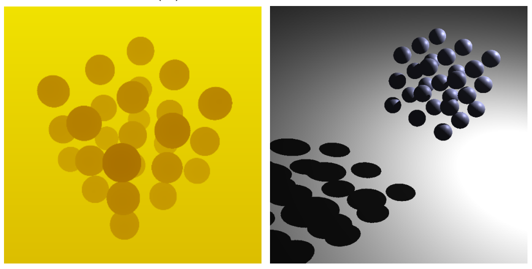
This is the basic shadow rendering technique even for early offline renderings.

2.1.1 2-Pass Algorithm

The light source view space pass generates the shadow map (Depth map) (Figure 1b). The camera view space pass use the shadow map to render the scene (Figure 1c). If the depth of the current shading point is bigger than the recorded depth in shadow map at same location in light source view space. Then this shading point will be rendered in black (Figure 1a).



(a) 2-Pass Algorithm



(b) Shadow map (c) Scene

Figure 1: Shadow mapping

Pro: No knowledge of scene's geometry is required.
Con: Causing self occlusion and aliasing issues.

2.2 Percentage-Closer Filtering

The percentage-closer filtering (PCF) [2] solves the aliasing problem in shadow mapping. The procedure can be defined as follows:

$$V(x) = \sum_{y \in \mathcal{N}(x)} w(x,y) \cdot \chi^+ \left[D_{\text{ShadowMap}}(y) - D_{\text{Scene}}(x) \right] \tag{1}$$

$$\chi^+[a] = \begin{cases} 0 & if \quad a < 0 \\ 1 & if \quad a \geq 0 \end{cases} \tag{2}$$

Here, $D_{\text{ShadowMap}}$ gets the recorded depth in shadow map, D_{Scene} gets the depth of the current shading point. $y \in \mathcal{N}(x)$ is the neighbor pixel of the current shading point defined by the filter. The function χ^+ defines the visibility. So we are doing a convolution operation for the visibility around a shading point for it's neighborhood, and in practice, the weight $w(x,y)$ basically set to 1.

2.3 Percentage-Closer Soft Shadows

Based on the results of percentage-closer filtering (PCF) [2], we found that the size of the filter has a decisive role in controlling the softness of the shadows. So the idea of percentage-closer soft shadows (PCSS) [1] is to decide the size of the filter based on the distance from different shading points to the light source, the distance from the shades to the light source, and the size of the light source, so as to control the softness of the shading at different shading points.

We define s_{filter} as filter size, which can be calculated as follows (Figure 2):

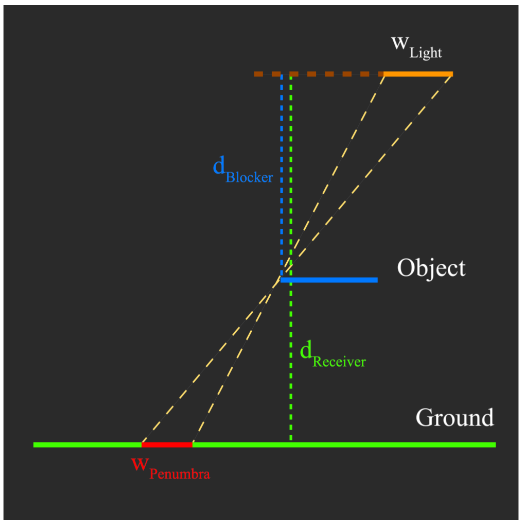


Figure 2: Filter size calculation

$$w_{Penumbra} = \frac{(d_{Receiver} - d_{Blocker}) \cdot w_{Light}}{d_{Blocker}} \tag{3}$$

$$s_{filter} \propto w_{Penumbra} \tag{4}$$

PCSS is implemented according to the following three steps:

2.3.1 Step 1: Blocker search

In this step, we get the value of $d_{Blocker}$, the average blocker depth in a certain region of the current shading point. For initialization, we need a range of region to compute the $d_{Blocker}$. In practice, this range often set to a constant value or proportional to the w_{light}.

$$d_{Blocker} = \frac{1}{N_{occ}} \sum_{x_{occ} \in V_{occ}} D_{\text{ShadowMap}}(x_{occ})$$

Here, N_{occ} is the number of blockers, V_{occ} is the set of all the blockers.

2.3.2 Step 2: Penumbra estimation

In this step, we use $d_{Blocker}$ the average blocker depth to determine s_{filter} the filter size, described in (3) (4).

2.3.3 Step 3: Percentage closer filtering

Exact same process that use s_{filter} as filter size, described in section 2.2 (1) (2).

The first and third steps are the main reason for slowing down the rendering, because the softer the shadows, the larger the size of the filter needs.

The most straightforward solution is to sparsely sample the interior of the filter, but this will make the rendered shadows noisy.

3 Variance Soft Shadow Mapping

Key idea: quickly compute the mean and variance of depths in an area to be able to simulate the data distribution of the filter's area. In order to accelerate the process in step 1 and step 3 of percentage-Closer Soft Shadows (PCSS) [1] to get a great soft shadow in real-time rendering.

3.1 Accelerated Process

Optimize the first and third steps in PCSS.

3.1.1 Compute filter's area mean

Summed Area Tables(SAT)

$$SAT_{mn} = \sum_{i=1}^{m} \sum_{j=1}^{n} SM_{ij} \tag{5}$$

$$z_{Avg} = \frac{LR - UR - LL + UL}{w \times h} \tag{6}$$

We define SM as shadow map. With Summed Area Tables (SAT) we can quickly obtain the average in the filter's area. Although the cost of computing SAT is

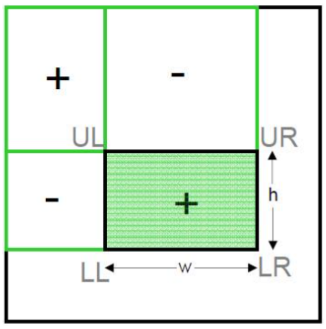


Figure 3: Summed Area Table (SAT)

still large, reaching $O(n \times m)$, but due to the high parallelism of the GPU, we can reduce the number of computations to $(\log_2(n) + \log_2(m))$. See OpenGL SuperBible: Comprehensive Tutorial and Reference, Chapter 10 [3].

3.1.2 Compute filter's area variance

Since we already know the mean, we are able to obtain the variance by using the following formula.

$$\mathrm{Var}(X) = \mathrm{E}\left(X^2\right) - \mathrm{E}^2(X) \tag{7}$$

We just need to generate a "square-depth map" along with the shadow map.

3.1.3 Compute shade's area (CDF estimation)

We want to get the percentage of pixels that are closer than the shading point.

Chebychev's inequality (one-tailed version, for $t > \mu$), μ: mean, σ^2: variance.

$$P(x > t) \leq \frac{\sigma^2}{\sigma^2 + (t-\mu)^2} \tag{8}$$

Here $\leq$ we assume is $\approx$, in practice, the actual results are not bad.

3.1.4 Estimate average blocker depth

Following the previous steps, we have successfully accelerated the Step 3 in PCSS, To accelerate the first step we need to get $d_{Blocker}$ the average blocker depth using follow equation, z_{Avg}: Average depth, z_{occ}: Average blocker depth, z_{unocc}: Average non-blocker depth, N: Filter pixel number, N_1: Non-blocker number, N_2: Blocker number:

$$\frac{N_1}{N} z_{unocc} + \frac{N_2}{N} z_{occ} = z_{Avg} \tag{9}$$

z_{Avg}, we can get from the Summed Area Tables (SAT) (6). $\frac{N_1}{N} = P(x > t)$ we get from the Chebychev's inequality (9), $\frac{N_2}{N} = 1 - P(x > t)$. z_{occ} is the variable we want to calculate, z_{unocc} is the varialbe we realy don't know. Since shadows are mostly projected on a plane, we assume that z_{unocc} is equal to the depth of the current shading point.
Now we can accelerate the Step 1 in PCSS.

4 Results

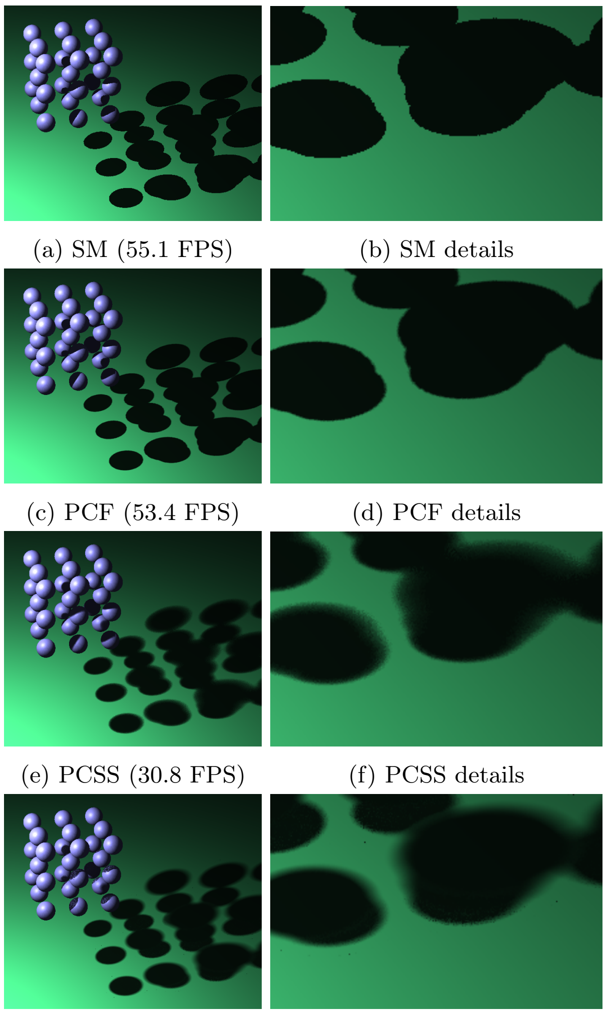


Figure 4: Shadow rendering

Render scene with PCSS, the sampling rate of filter is set to 75%. Even with a high sample rate setting, noise can be observed at the edge of the shadows, and framerate drop to 30.8 FPS. So it is difficult to use PCSS to achieve a realistic soft shadow under the requirement of real time.

Render scene with VSSM, We can see a white border at the two connected shadows, that is "light leaking". The reason is that the distribution of the depth values in the filter's area is incorrectly estimated (Figure 5).

If there is depth information for multiple objects in a filter's area, the distribution should be closer to the green curve than to the red curve. This makes the $P(x > t)$ estimated with Chebychev's inequality (9) larger than the true value, which makes the percentage of pixels that are closer than the shading point smaller than the true value, which makes the rendered shadow lighter.

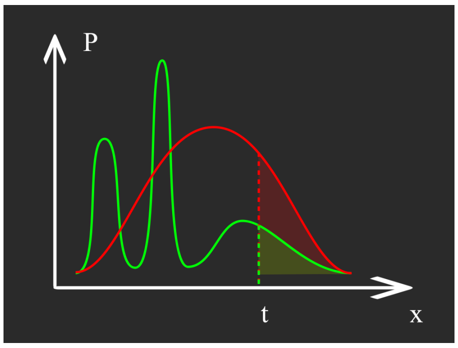


Figure 5: Light leaking

References

[1] Randima Fernando. Percentage-closer soft shadows. In *ACM SIGGRAPH 2005 Sketches*, pages 35–es. 2005.

[2] William T Reeves, David H Salesin, and Robert L Cook. Rendering antialiased shadows with depth maps. In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, pages 283–291, 1987.

[3] Graham Sellers, Richard S Wright Jr, and Nicholas Haemel. *OpenGL superBible: comprehensive tutorial and reference*. Addison-Wesley, 2013.

[4] Lance Williams. Casting curved shadows on curved surfaces. In *Proceedings of the 5th annual conference on Computer graphics and interactive techniques*, pages 270–274, 1978.

[5] Baoguang Yang, Zhao Dong, Jieqing Feng, Hans-Peter Seidel, and Jan Kautz. Variance soft shadow mapping. In *Computer Graphics Forum*, volume 29, pages 2127–2134. Wiley Online Library, 2010.