SHADOW RENDERING TECHNIQUES: HARD AND SOFT

A Dissertation

Submitted to the Department of Computer Science and Engineering

of

BRAC University

by

Jamiur Rahman

Student ID: 06341003

In Partial Fulfillment of the

Requirements for the Degree

of

Bachelor of Science in Computer Science

May 2007

DECLARATION

We, hereby, declare that the work presented in this thesis is the outcome of the investigation performed by me under the supervision of Mushfiqur Rouf, Lecturer, Department of Computer Science and Engineering, BRAC University, Dhaka.

I also declare that no part of this thesis and thereof has been or is being submitted elsewhere for the award of any degree or Diploma.

Sign

(Jamiur Rahman)
Author

Countersigned

(Mushfiqur Rouf)
Supervisor

Abstract:

Shadows are very important issues in three-dimensional computer graphics where interactive techniques are certainly needed to make the scene near about real life scene e.g. in virtual reality systems or games.

Shadows can be defined as two categories based on their characteristics such as hard shadow and soft shadow. Various techniques have been implemented for both of the techniques throughout the world. The basic idea is to render scene with shadows just as like the real scene as much as possible. In this thesis, we have discussed about several techniques of hard and soft shadows in simple and complex scenes along with algorithms and performance related to space.

Contents

Acknowledgement

I would like to express my deepest appreciation to my supervisor Mr. Mushfiqur Rouf for thoughtful suggestions. He has chosen the topic for me and always has given me inspirations along with effective favorable comments. In a nutshell, he is one of the most wonderful persons I have ever met. Again my heartiest thanks goes to him.

My friends and teachers gave me continuous inspiration during my thesis. Their list is too long to mention here. I would like to express my gratitude to my parents who always encouraged me to study well to get my goals and fulfill the desire to be the one of the most important researcher in the world and they are still doing without any tiredness.

Chapter 1
Introduction

1.1 Shadows:

A point is in shadow if the light got blocked between the light source and point. In the following picture shadowed and non shadowed point can be differentiated easily.

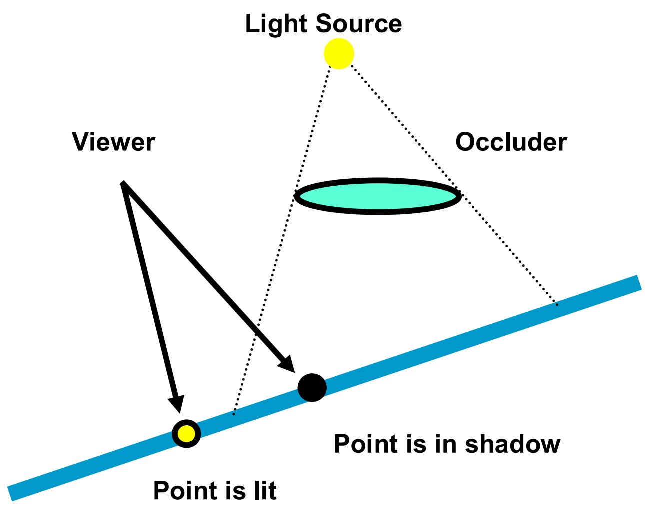
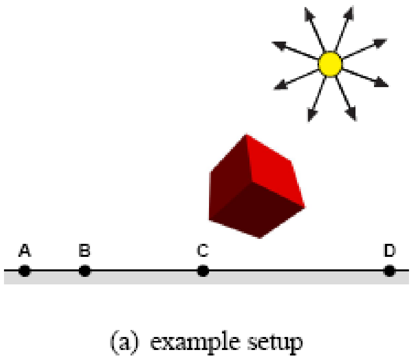


Figure 1.1.1: Shadows caused by the occluder

Rendering an object with realistic appearance we need to evaluate the reflection model on all surface points with respect to those rays of light that hit the surface at the given point. This requires an examination of the environment in which the object is placed: Firstly, the object may not be in the view of the light source, may be completely outside the cone of a spot light. Secondly, there could be other objects in the scene that block some or all of the rays from the light source to the surface point. An example of the later situation is sketched in Figure 1.1.2(a)



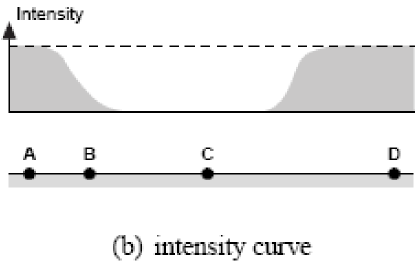
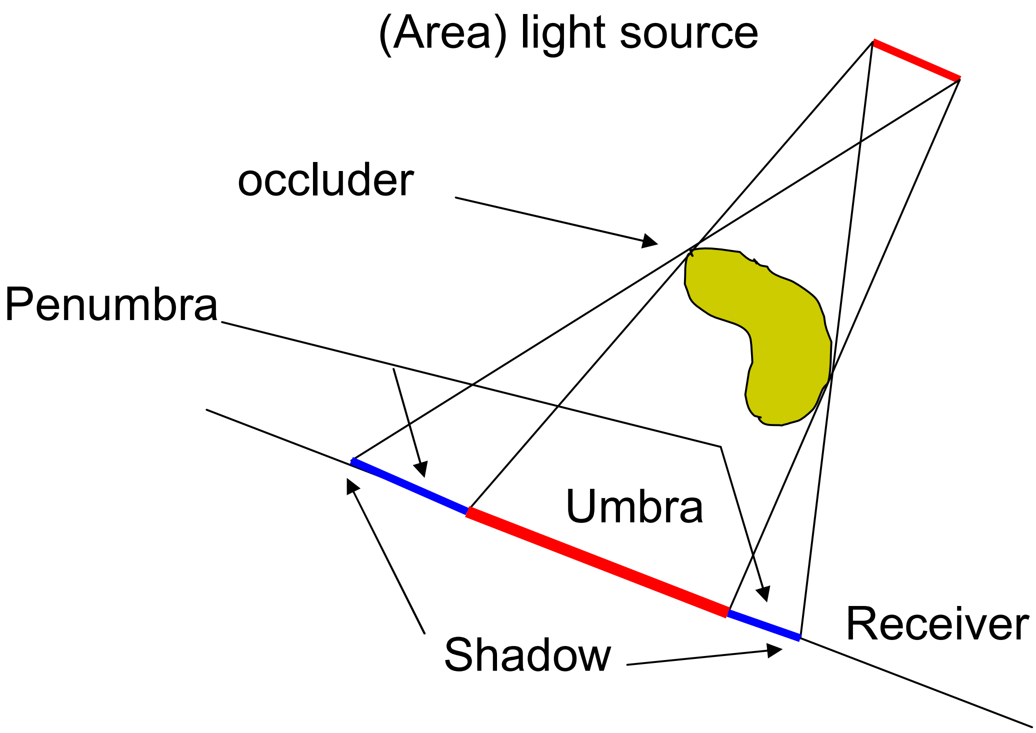


Figure 1.1.2: Example of intensity transition

this scene we want to determine the contribution of the luminaire above at various sample points (A,B,C,D) on a given surface. At point A, all rays that are emitted in the direction to A hit the surface. For point B some of the light rays are blocked by the cube in between. At sample C no light at all can hit the surface, whereas point D again receives the full energy. Computing the intensity level for all sample points on the surface results in a smooth curve as shown in Figure 1.1.2(b). We can classify the points by either fully lit (A,D), partially lit/shadowed (B), or fully shadowed (C). It is clear that this intensity curve does not only depend on the spatial arrangement of receiver surface, blocker, and luminaire but also on the type of light source used.

1.1.1 Terminology: Umbra and Penumbra



- **umbra** – fully shadowed region
- **penumbra** – partially shadowed region

Figure 1.1.1.1: Umbra and Penumbra

1.1.2 Hard Shadows and Soft Shadows:

Given a point light source which emits light from single point in space, there's only one ray of light that can potentially hit the surface point, which is exactly the ray originating from the light source and directed towards the surface point.

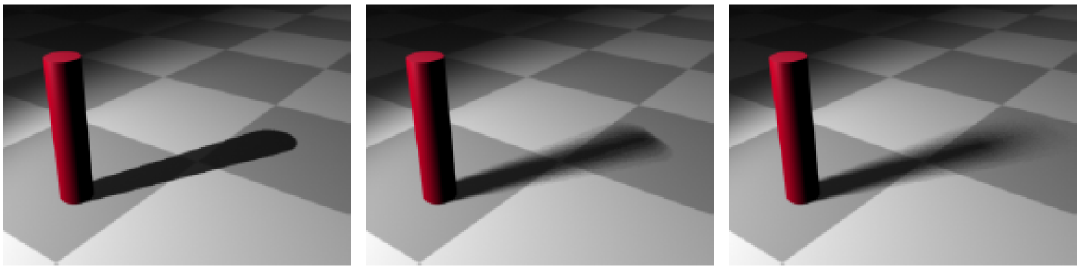
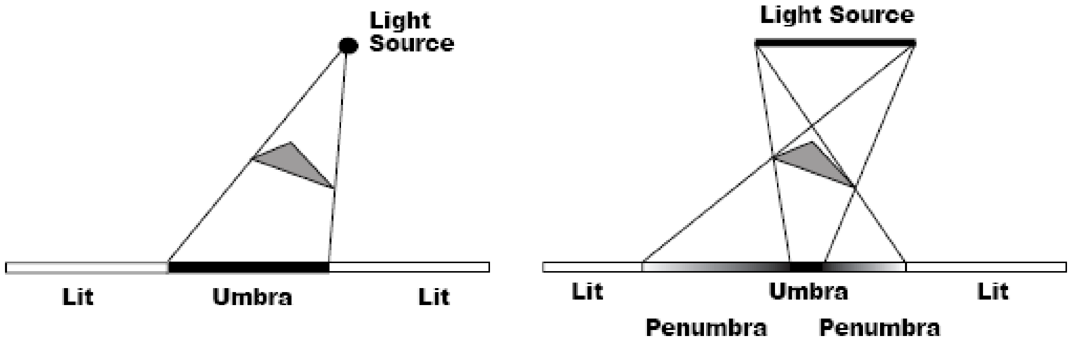


Figure 1.1.2.1: Shadows from different types of light sources.

The corresponding intensity level can therefore only be 100% (fully lit) or 0% (fully shadowed), depending whether the ray is blocked or not. The resulting shadow will exhibit sharp boundaries where the intensity level changes as can be seen in the example scene shown in Figure 1.1.2.1(a). We will refer to this type of shadow as *hard shadow* or *umbra*. Sharp shadow boundaries are common to all non-extended light sources, such as directional, point, and spot lights.

Using a linear light source, a number of light rays may hit a specific surface point, all originating from sample points on the line segment associated with the light source. Contribution therefore varies between 0% and 100%, resulting in a smooth shadow transition as shown in Figure 1.1.2.1(b). Linear light sources have a specific characteristic that can be observed at the top of the cast shadow in Figure 1.1.2.1(b): Smooth shadow regions are only generated for blocker edges that are not parallel to light source's line segment. In the parallel case a hard shadow transition will be visible, since all rays are blocked when crossing the given edge. We refer to the region in which the transition from lit to shadowed takes place as the *penumbra*.

A shadow consisting of penumbra and umbra regions is called *soft shadow*.



(a) point (b) linear

Figure 1.1.2.2: Hard vs. soft shadows.

For a linear light source we can easily visualize the geometric relationship between blocker, receiver, and light source and construct the shadow boundaries. Figure 2.6 shows such a 2D arrangement. On the left side, a point light source is used. As we can see, the contribution of the light source changes at two points on the receiver, resulting in a lit-umbra-lit transition. Using an extended light source, e.g. a linear one as shown on the right side of Figure 2.6, we can determine the relevant points at which illumination changes by drawing lines between the extremal points of the blocker and the end points of the linear light. This gives us the regions on the receiver in which the light source is fully visible, partially visible/occluded, and fully occluded, resulting in a lit-penumbra-umbra-penumbra-lit transition.

Overall smooth shadows can be obtained when using an area light source (Figure 1.1.2.1 C). As explained before, an area light is of finite size where every point on the surface can emit rays of light. As in the linear case, each point on a surface may be hit by all, some, or none of the light rays, resulting in a *soft shadow* if blocking objects are in between.

Hard shadow
– *point* light source

Soft shadow
– *area* light source

Chapter 2
Hard Shadow Rendering

In this Chapter, we will discuss some general hard shadow rendering techniques which have been improved and modified by the researchers to do the research on the various objects or methods.

2.1 Shadow Volumes: [Crow77] *Shadow algorithms for computer graphics*

From the School of Leonardo Da Vinci

Compute regions of shadow in 3D can be divided into the following categories

- Object-space algorithm
- Cast shadows onto arbitrary receiver geometry (polygons)

Crow's shadow volume algorithm [Crow77] is one of the most popular algorithms for shadow generation. By extending occluder polygons to form semi-infinite volumes, so called shadow volumes, shadowed pixels can be determined by simply testing if the pixel lies in at least one shadow volume. A hardware-accelerated implementation of Crow's shadow algorithm was later proposed by Heidmann [Heidmann91]. Especially for real-time applications it is the de-facto standard way for precise, high quality shadows. This is due to the fact that shadow information is generated in object space, meaning that shadow information is available for every window-space pixel.

We need to extend occluder polygons to form semi-infinite volumes. There will be some conditions

- Light source is *center-of-projection*
- Everything behind occluder is in shadow
- Test if point is in at least one volume!
- Extend to reach outside of view frustum

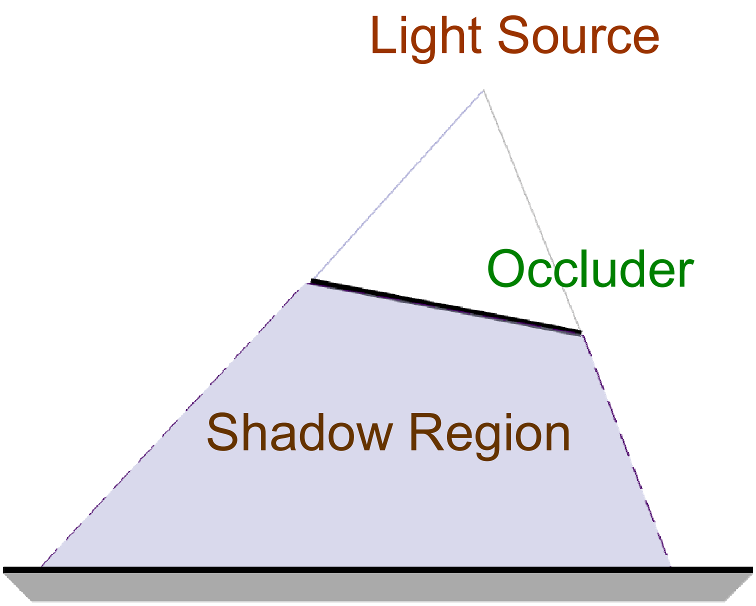


Figure 2.1.1: Shadow Region

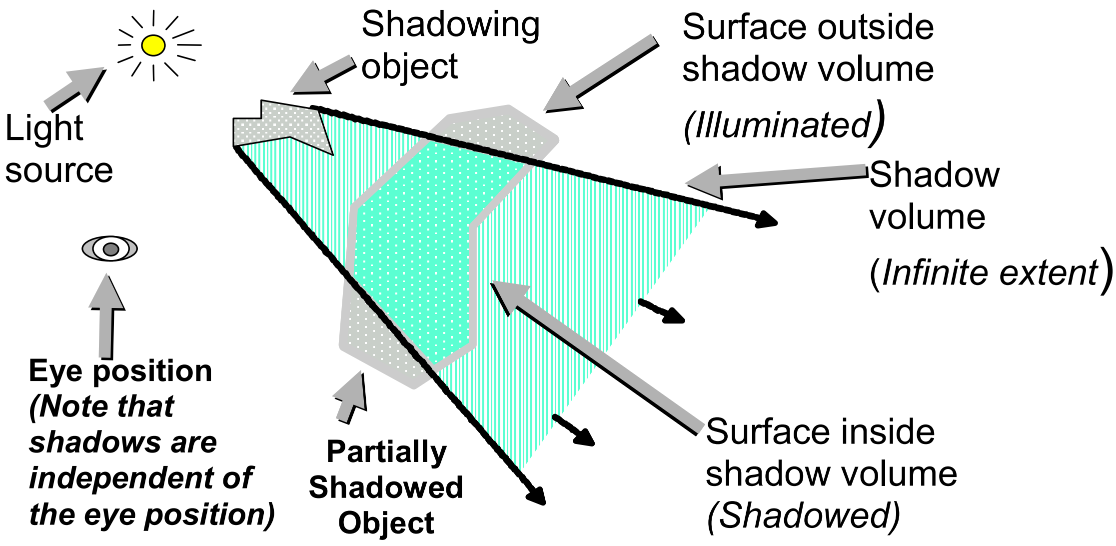


Figure 2.1.2: Shadow Volumes illustrated (2D)

2.1.1 Shadow volume generation

The Trivial way to generate shadow volume is to have one volume for each polygon but better is to apply Silhouette-based approach in which goal is to generate one shadow volume for selected occluders only.

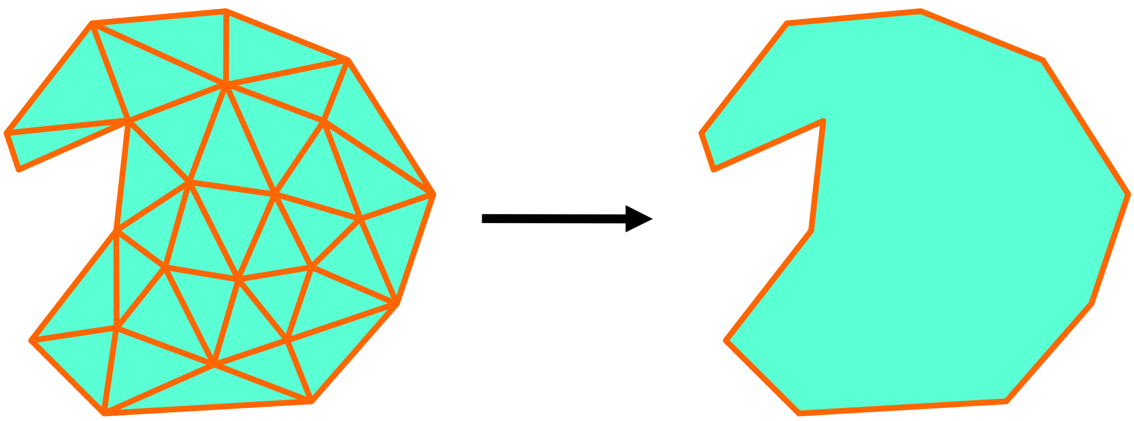


Figure 2.1.1.1: Shadow volume generation

2.1.2 Detect silhouette edges:

The shadow volumes algorithm starts with the detection of possible silhouette edges. An edge is a silhouette edge if it is an edge shared by a *front-facing* and *back-facing* triangle/polygon. For simplicity, we assume that all shadow casting objects are closed triangular meshes (2-manifold) for which connectivity information is available.

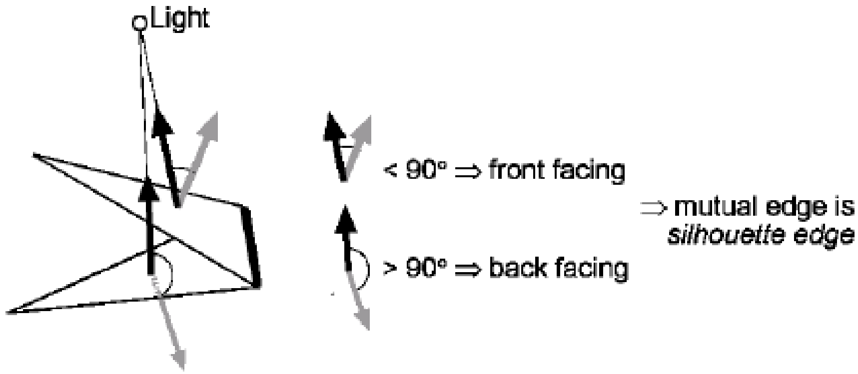


Figure 2.1.2.1: Silhouette edge detection.

To test whether a given edge is a silhouette edge we check if the edge connects a front-facing and a back-facing triangle, with respect to the light source. This is illustrated in Figure 2.1.2.1. Triangle orientation can easily be checked by taking the dot product of the face normal and the vector to the light source. If this dot product is negative, a triangle is back-facing with respect to the light, otherwise it is frontfacing. Repeating this for all edges, we obtain a set of silhouette edges that form closed loops.

Next we extrude these silhouette loops to form semi-infinite volumes. For each silhouette edge a quadrilateral is constructed by taking the two original vertices of the edge and two vertices which are computed by moving the original vertices far away to infinity along the ray originating from the light source through the vertex. Together with the object's front facing triangles, these quadrilaterals bound all regions in space which are in shadow. In order to check if a given point is in shadow all we have to do is to determine if the point lies outside of all shadow volumes.

This information can be easily obtained by following a ray from the viewer to the surface point and counting how many times we enter or leave a shadow volume boundary polygon. This counting scheme is illustrated in Figure 2.1.2.2.

Here shadow volumes have been generated for a sphere and a box illuminated by a point light source. While following the ray from the viewer to surface point *A*, we count how many times we enter (increment) and leave (decrement) a shadow boundary. The final counter value of 0 indicates that the surface point is lit by the light source, since we have left the shadow regions as many times as we entered them. Counting shadow boundaries for surface point *B* yields a value of 2, since the point is inside two shadow volumes (sphere and box).

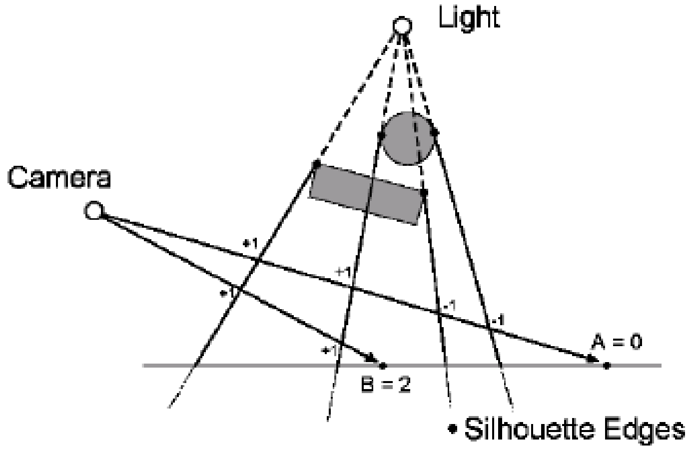


Figure 2.1.2.2: Inside-outside test.

2.1.3 Basic Shadow Volume Algorithm (Use stencil buffer as counter):

The algorithm is given below

1. Render scene (ambient color)
2. Render front-facing parts of shadow volumes with stencil operation *increment* (enter)
3. Render back-facing parts of shadow volumes with stencil operation *decrement* (exit)
4. Render scene (fully lit) with stencil test *equal zero*

2.1.4 Facts about Shadow Volumes

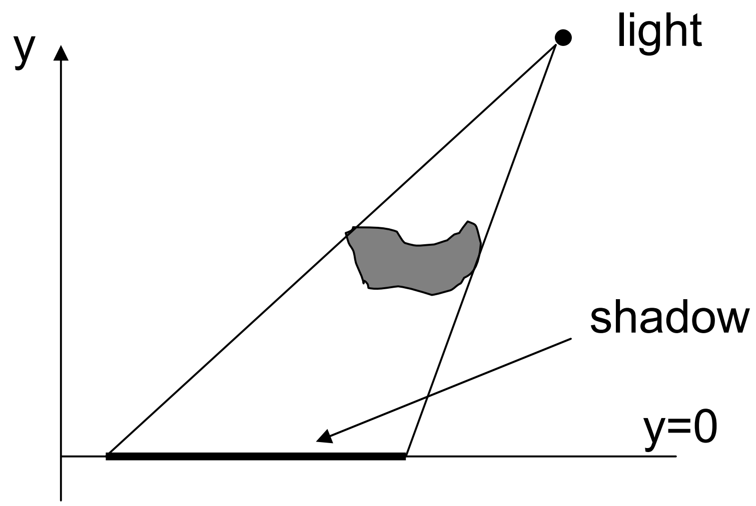
a. Shadow volumes in object precision
    i. CPU intensive, requires polygon representation
b. Shadow test in image precision
    i. Stencil buffer count
    ii. Hardware-accelerated

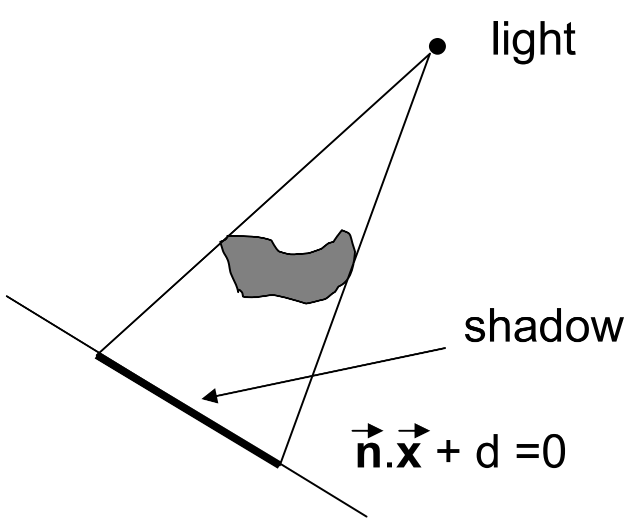
c. Many, large shadow polygons

i. Fill rate, geometry processing

2.2 Planar Shadows:

The simplest algorithm – shadowing occurs when objects cast shadows on planar surfaces (projection shadows).





Special case in planar shadow is the shadow receiver is an axis plane. Then just project all the polygon vertices to that plane and form shadow polygons.

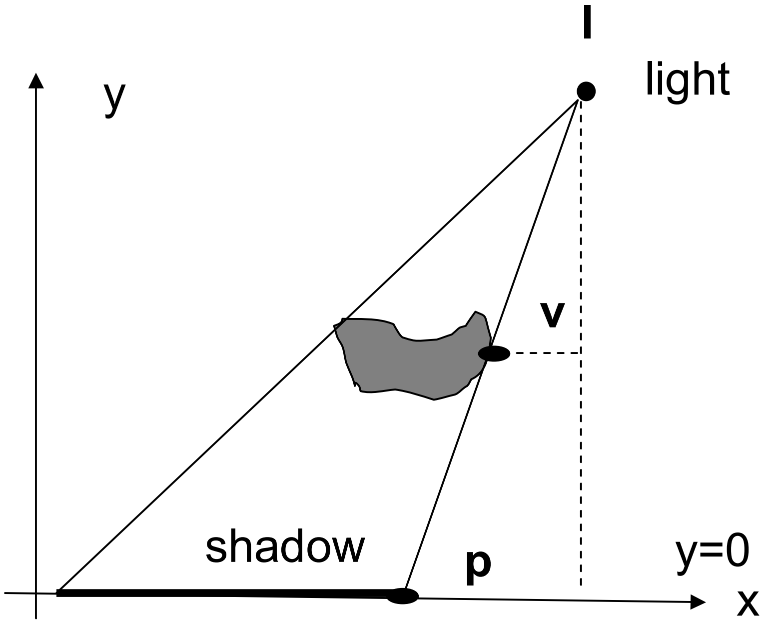


Figure 2.2.1: Planar Shadows in the plane y=0

Given:

- Light position **l**
- Plane position y = 0
- Vertex position v

Calculate: **p**

We can use similar triangles to solve **p**

$$\frac{\mathbf{p}_x - \mathbf{l}_x}{\mathbf{v}_x - \mathbf{l}_x} = \frac{\mathbf{l}_y}{l_y - \mathbf{v}_y}$$

From above equation, we can calculate p that is the point on the surface in shadow. Same principle applied to different axis planes.

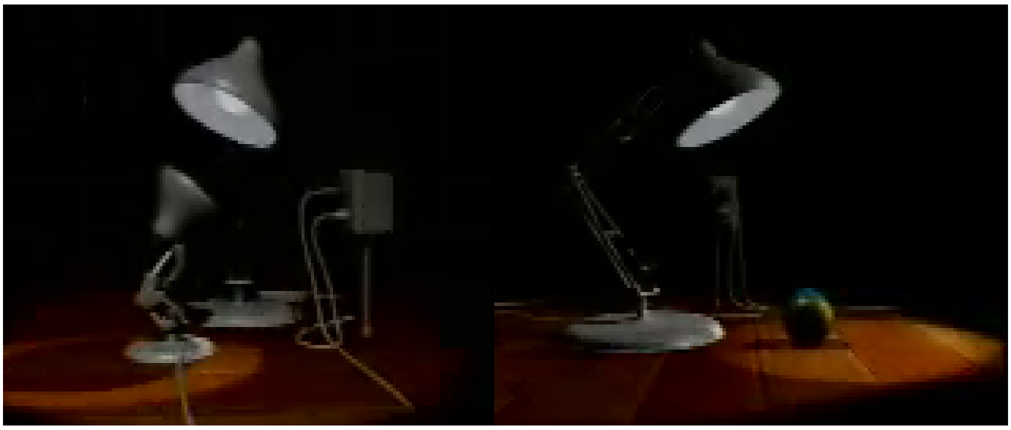
2.2.1 Basic algorithm

- Render scene (full lighting)
- For each receiver polygon
  - Compute projection matrix *M*
  - Mult with actual transformation (modelview)
  - Render selected (occluder) geometry
    - Darken/Black

2.3 Shadow Maps: [Williams78]

Williams' shadow map algorithm [Williams78] is the fundamental idea of most shadow methods working on sampled representations of the scene.

It casts curved shadows on curved surfaces for generation. Image-space algorithm is used for this techniques. The technique is well suited for hardware implementation.



Real-Time Luxo Jr. …uses three dynamic shadow maps (OpenGL, GeForce3)

Basic idea of shadow maps is objects that are not visible to the light are in shadow and in this case we need to determine whether an object is visible to the eyes. So, we need to use z-buffer algorithm, but now the "eye" is light, i.e., the scene is rendered from light's point of view. This particular z-buffer for the eye is called *shadow map.*

2.3.1 Basic Algorithm

- Render scene as seen from light source
- Save back depth buffer (2D shadow map)
- Render scene from viewer's position
    - Transform pixel coordinates to light source space
    - Compare z with z value stored in shadow map
        - Pixel is in shadow if z(light) < z(viewer)

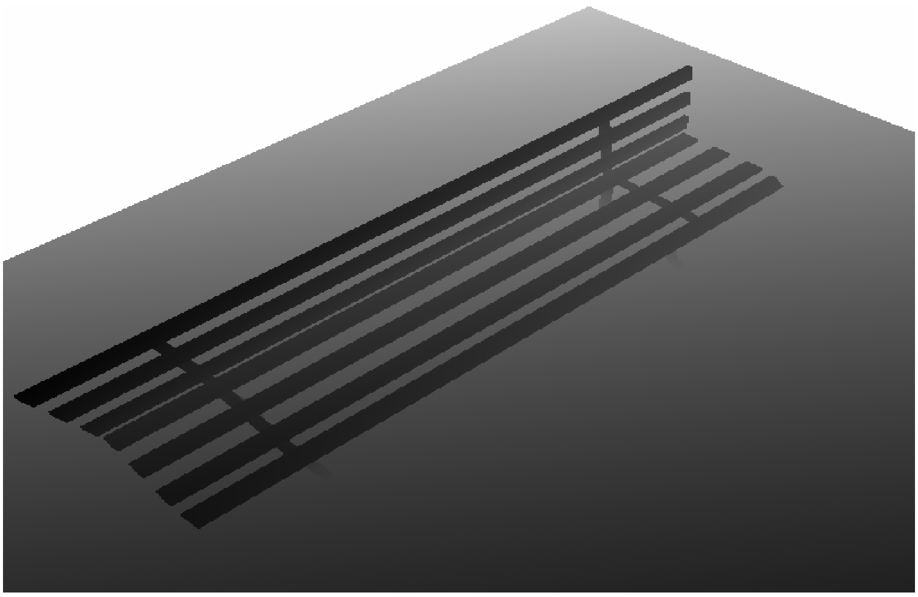


Figure 2.3.1.1: Shadow Map

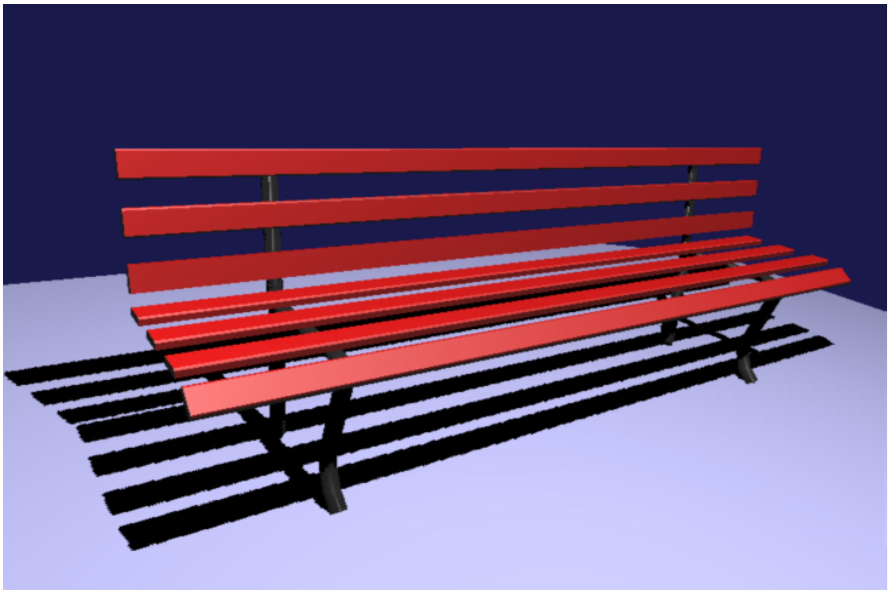


Figure 2.3.1.2: Final Scene

2.3.2 Problems:

1. Shadow Map needs high-resolution texture maps to sample fine details
   - Offscreen buffers, render-to-texture
   - Filtering : Percentage Closer Filtering (PCF)
2. There may be some numerical problems (light leaks, surface acne)
   - Bias needed [Reeves87]

2.4 Hybrid Approach

2.4.1 Algorithm overview:

1. Render scene from light source view (depth only)
2. Read back depth buffer
3. Reconstruct shadow volumes

   - Edge detection (Canny)

- Surface reconstruction (Template matching)

4. Render shadow volume with stencil operation

5. Render final scene (stencil test)

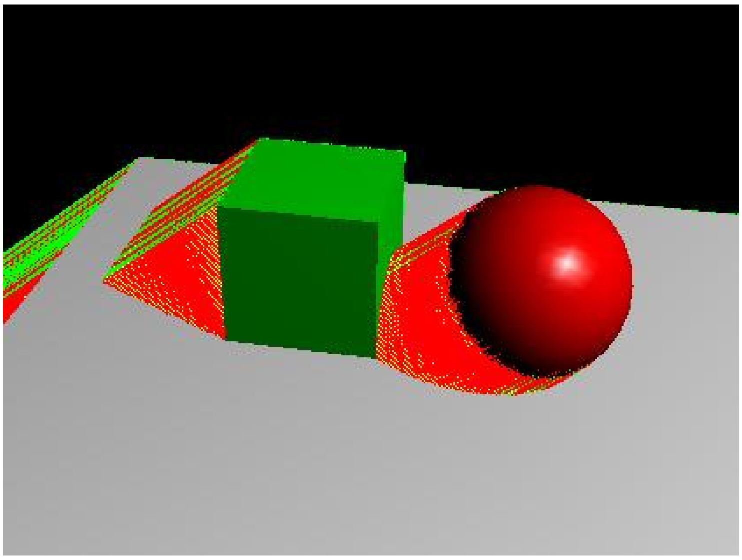


Figure 2.4.1.1: Hybrid Shadow Volume

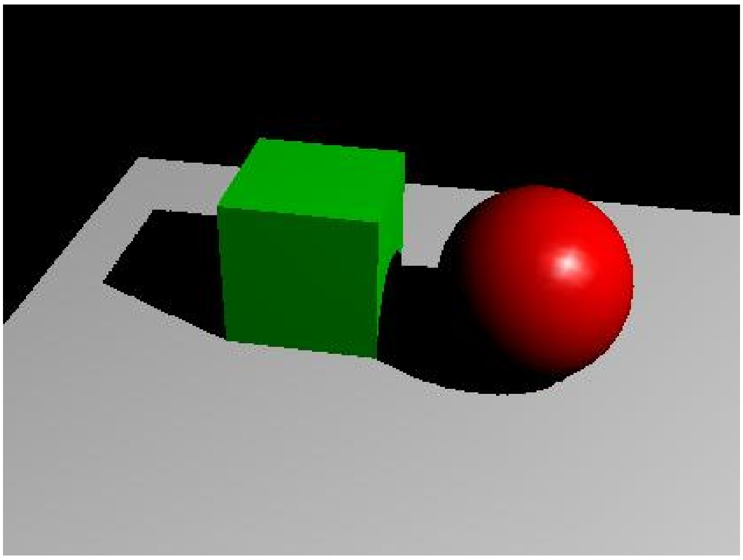


Figure 2.4.1.2: Final Scene (Hybrid Approach)

2.4.2 Facts about Hybrid Approach

Hybrid Approach is better than *normal* shadow volumes for very complex scenes

- Volume for silhouette
- Only one stencil bit (in-out toggle)

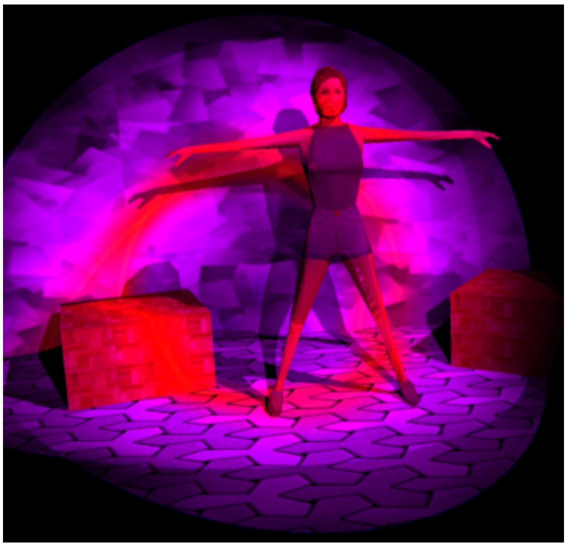
This approach needs CPU and memory transfer

- Use CPU's special instruction set
- OpenGL imaging extensions (convolution)

2.5 Extended Light Maps: [Brabec00] *Extended Light Maps*

Shadow map generation is expensive because of additional geometry pass. Besides it needs frame buffer reconfiguration which is also costly and slow process.

The main idea of this technique is that do as much work as possible during the shadow map generation.



2.5.1 Main Process

In this technique, Light Maps store pre-computed illumination and Extended Light Map uses Light Channel (RGB) which is *Normal* light map and Shadow Channel (Alpha) for Alpha-encoded depth values. It uses RGB channel for arbitrary calculations e.g.

- OpenGL lighting
- Object textures
- Realistic spotlights
- Special effects
- Reflection mapping
- Bump mapping

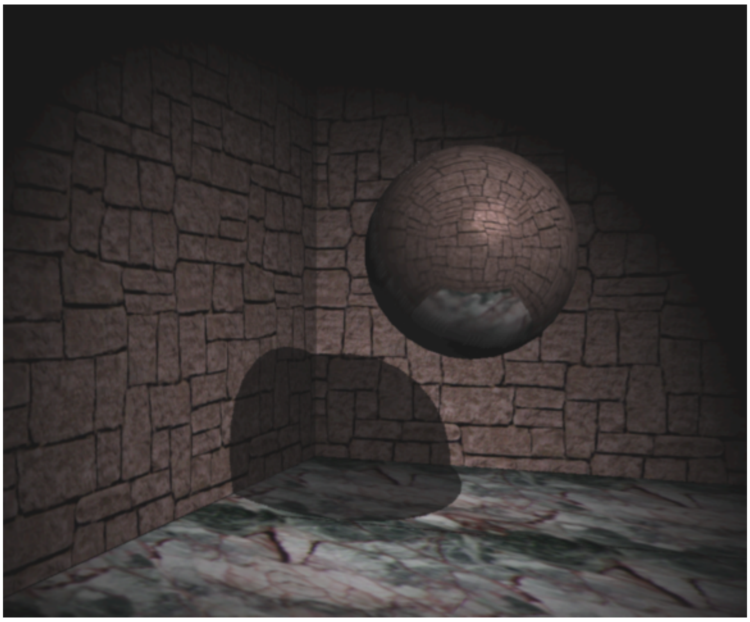


Figure 2.5.1.1: Pre-computed reflection mapping

- First pass:
  - Use color channel for environment cube map
  - Use alpha channel for z-distance

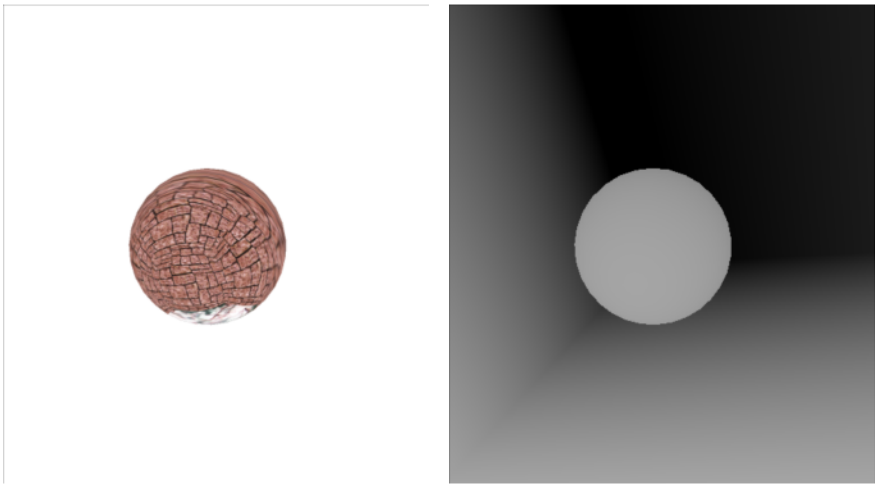


Figure 2.5.1.2: During 1st pass

- Two final passes
  - Render *normal* scene (textures & lighting)
  - Apply extended light map (shadows & reflections)

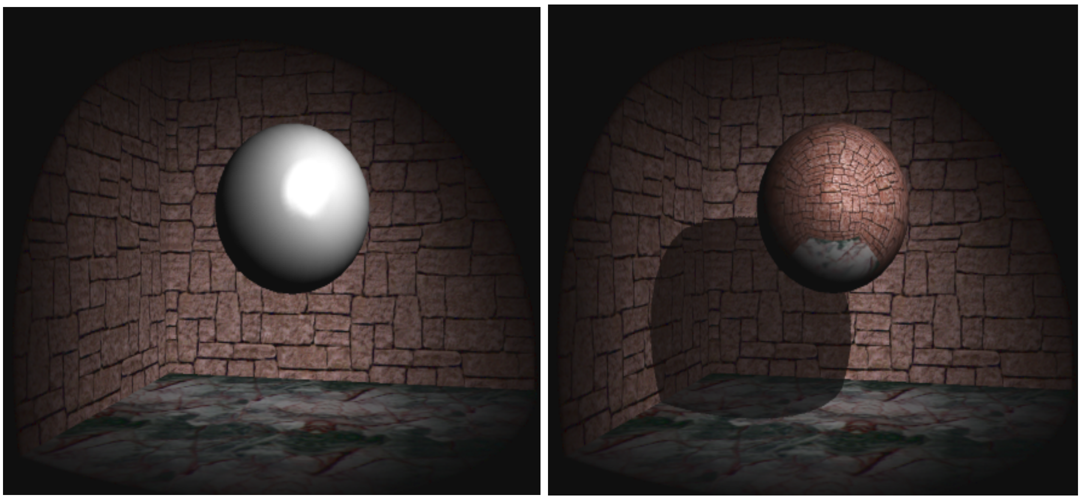


Figure 2.5.1.3: After 2 final passes

2.5.2 Facts about Extended Light Maps

Extended Light Maps based on additional calculations during shadow map generation.

- Utilize special hardware features
- Pre-compute light-relative ‘static’ things
- Use as much hardware resources as available!

Chapter 3
Soft Shadow Rendering

In this chapter, we will discuss several popular techniques of soft shadow rendering which have been modified and improved by the researchers to keep pace with their researches.

3.1 Sampling the Light Source

This process uses arbitrary hard shadow algorithm. It selects point sample on area light source then render hard shadows and sum up weighted result (e.g. accumulation buffer).

3.1.1 General Algorithm: Ground plane shadow texture

1. Initialize FB (white)
2. For each sample point do
   2a. Render scene
   2b. Subtract 1/N from FB only once for each pixel (stencil)!

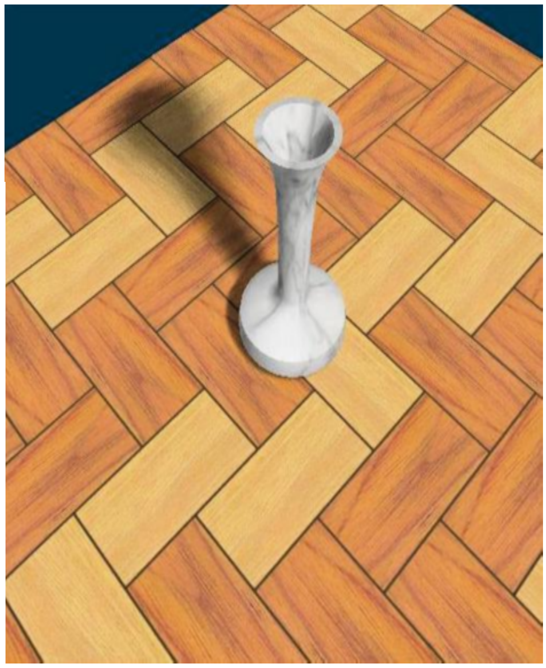
 

Image from ATI Developer's Site Result

Here Number of samples == number of passes, which is problematic for complex scenes because it is difficult to compute.

If there are N samples == (N+1) levels of shadow then shadows will be fully lit, fully shadowed with (N-1) levels of penumbra.

3.2 Soft Shadow Maps: [Heidrich00] *Soft shadow maps for linear lights*

In this technique, soft penumbra regions for linear light sources are computed. It is based on "traditional" shadow map algorithm. It is suitable for hardware and software rendering. It uses very small number of light source samples. It creates soft shadows at real-time / interactive frame rates.

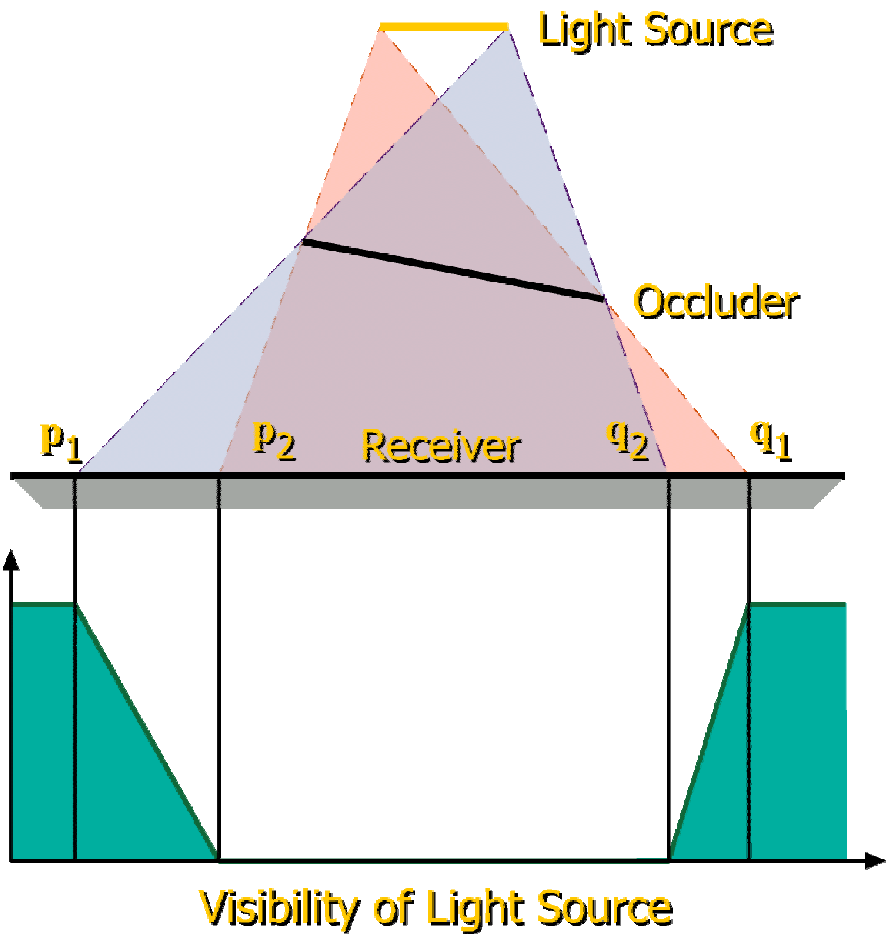


Figure 3.2.1: Visibility of light source

In the figure 3.2.1, the visibility of light source 100% to 0% for [p1, p2], 0% for [p2, q2] and 0% to 100% for [q2,q1]. The main idea is generation of *Normal* shadow maps for umbra and completely lit regions and linear interpolation of visibility for penumbra regions.

3.2.1 Visibility Map

For create the visibility map we need additional shadow map channel (percentage visibility) along with two-channel shadow map for each sample point.

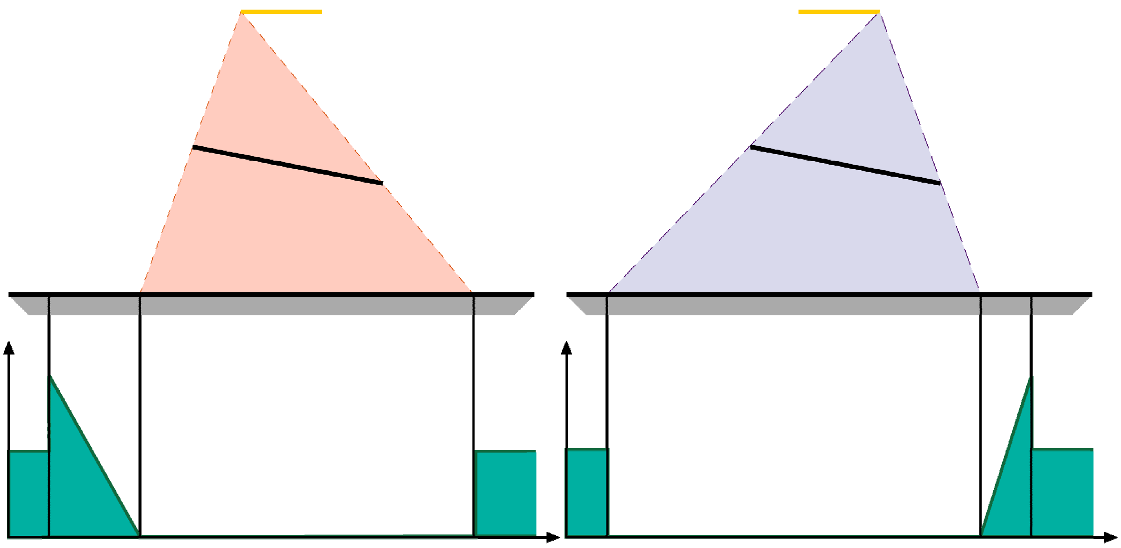


Figure 3.2.1.1: Two channel shadow map

Generating the visibility map we need to triangulate depth discontinuities (shadow map) and warp resulting skin polygons to other view.

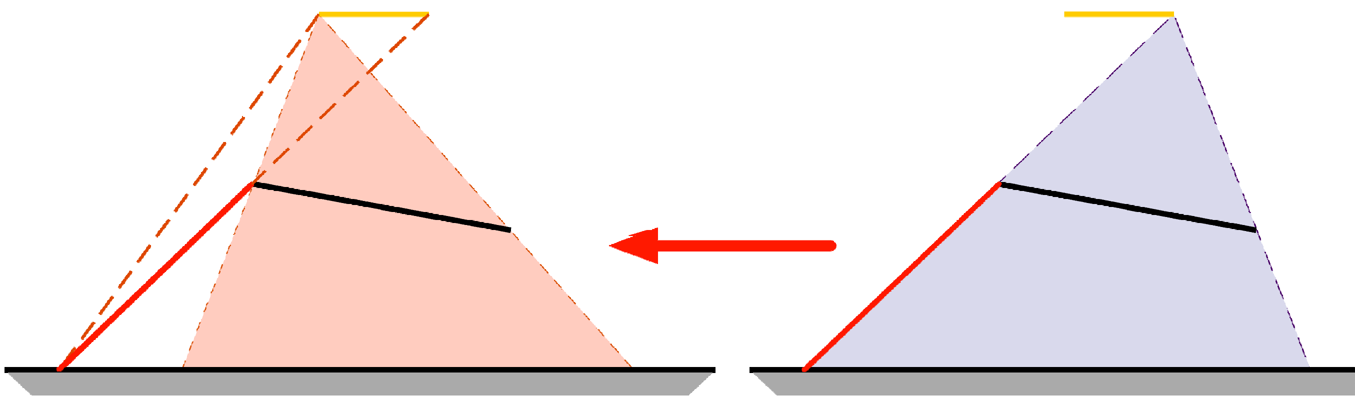


Figure 3.2.1.2: Generating the visibility map

Consequently render skin polygons to visibility map and apply Gouraud-shading (linear interpolation) such as “White” for vertices on receiver and “Black” for vertices on occluder.

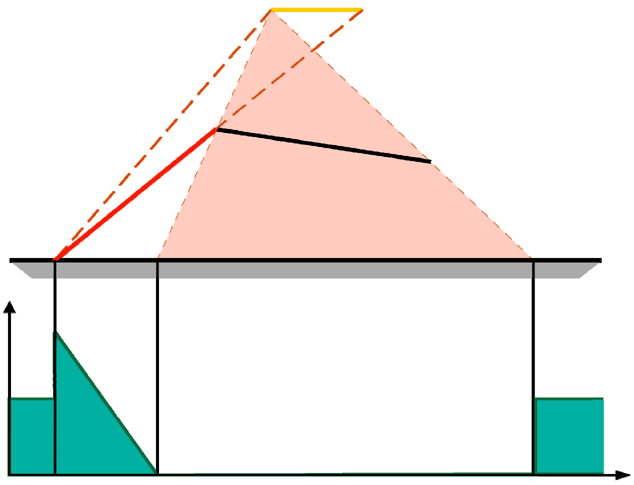
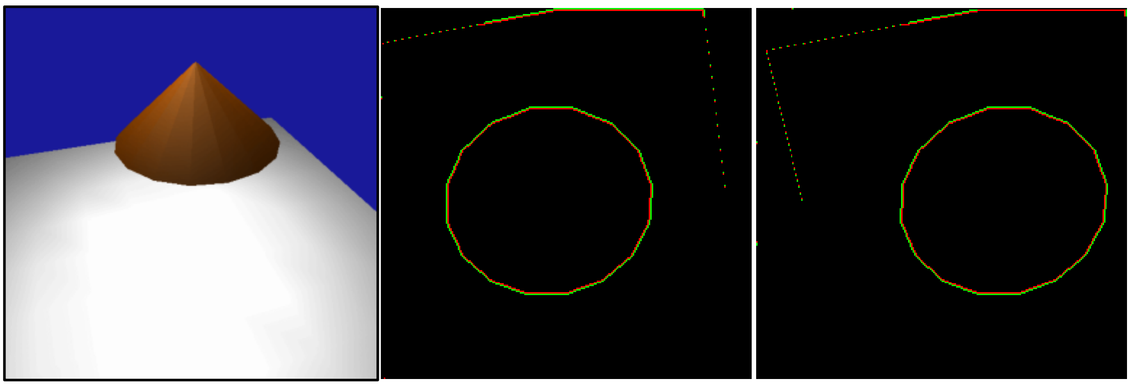


Figure 3.2.1.3: Generating the visibility map

The completely lit regions have default visibility 0.5 whereas completely shadowed regions will be first shadow map channel.



camera view *left sample point* *right sample point*

Figure 3.2.2: Edge Detection (on shadow map): Laplacian-of-Gaussian

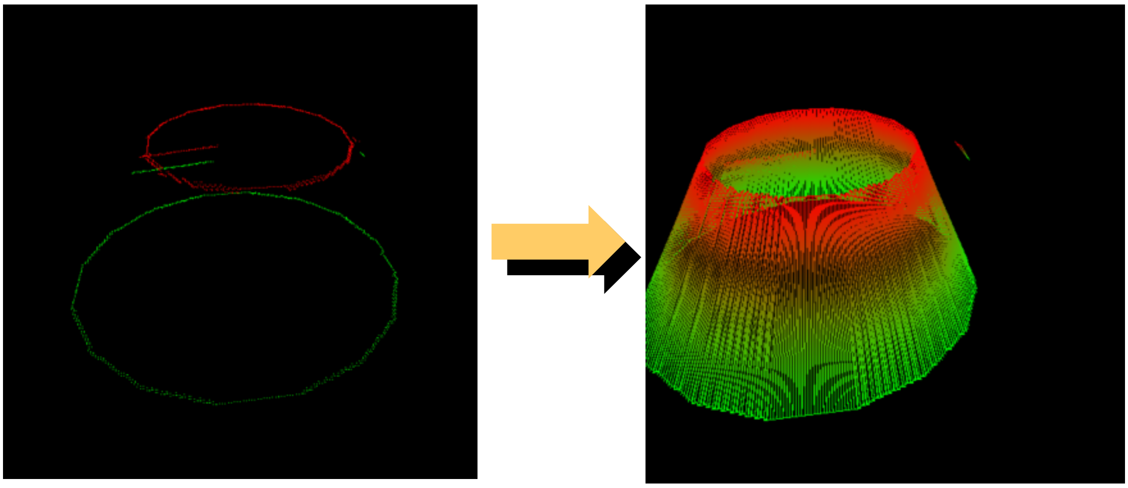


Figure 3.2.3: Triangulate and warp

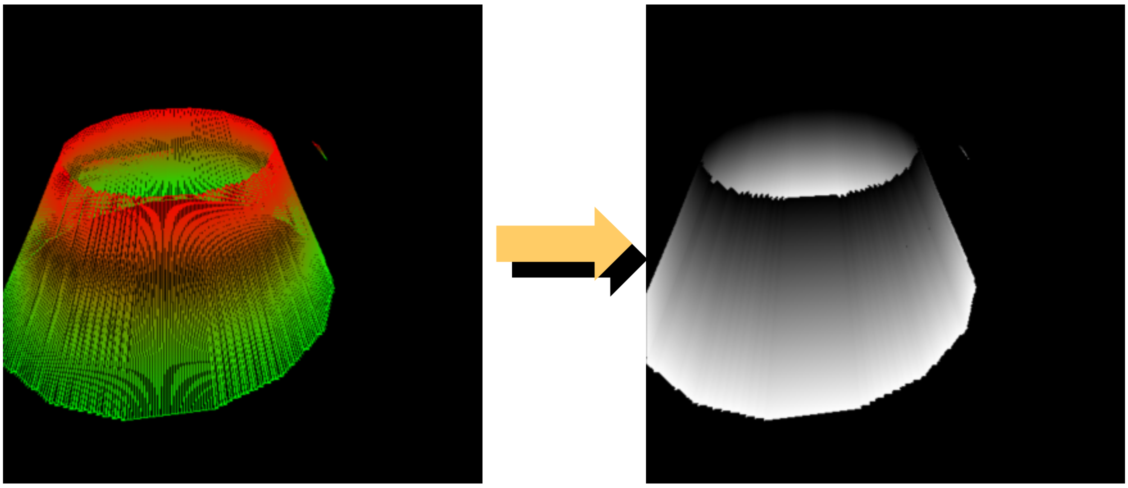
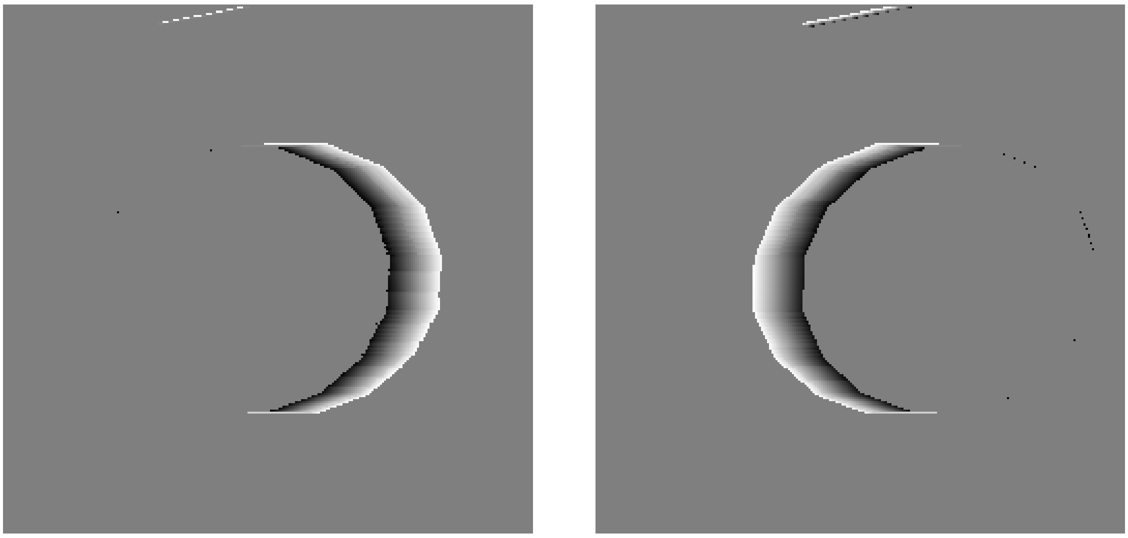


Figure 3.2.4: Render using Gouraud shading



left sample point *right sample point*

Figure 3.2.5: Visibility Maps

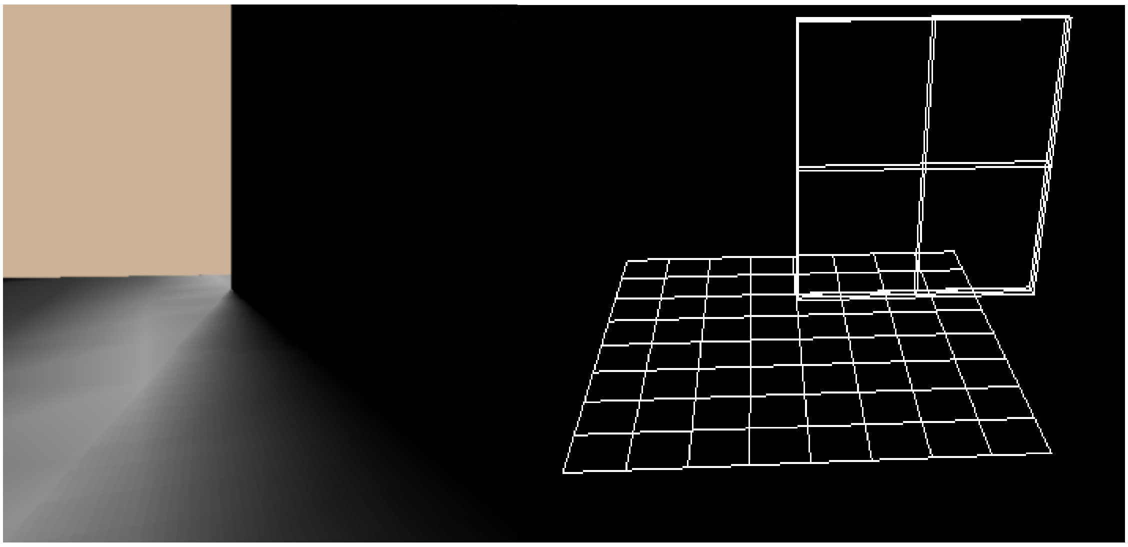
3.2.2 New shadow map algorithm for soft shadows

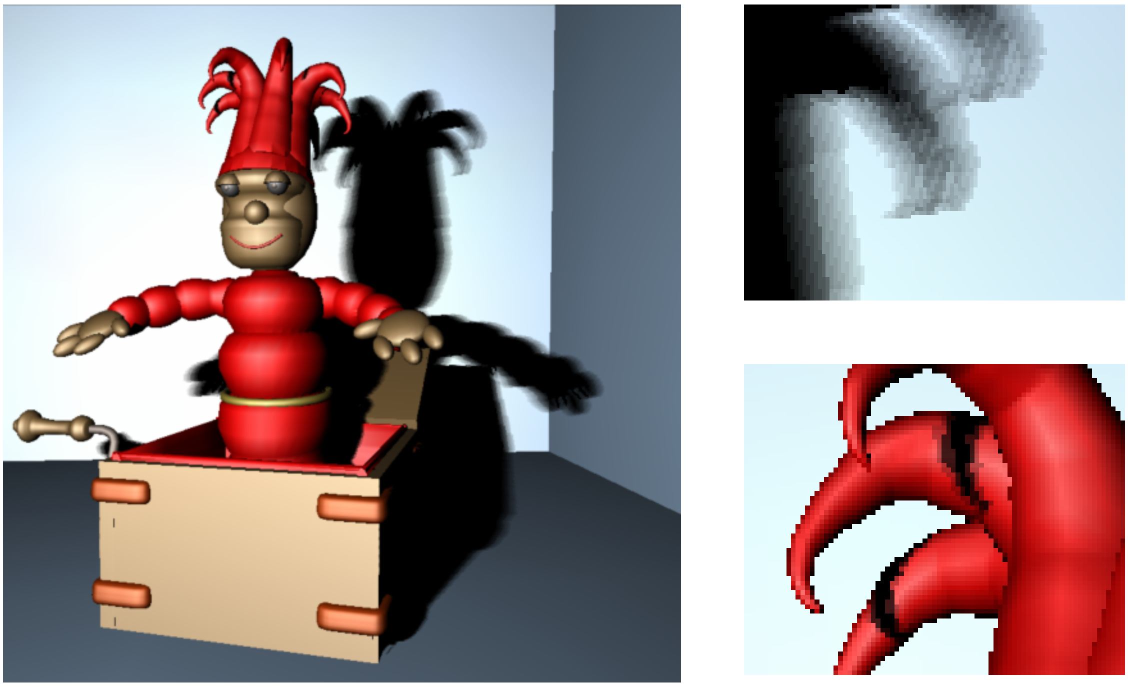
```
shade(p) {
     if( depth1(p) > S1[p] )
        I1 = 0;
     else
        I1 = V1[p] * illum(p,L1);

     if( depth2(p) > S2[p] )
        I2 = 0;
     else
        I2 = V2[p] * illum(p,L2);

     return I1+I2;
}
```

Where, p= desired point, I1=light1, I2=light2

3.2.3 Facts about soft shadow maps

In this technique, shadow & visibility maps only need to be re-computed if light and/or scene changes, so easier than ever and less costly. It has minimal overhead for “static walkthroughs”.





Soft Shadow Maps

3.3 Soft Shadow Volumes

Several researches adopted the shadow volume method to produce soft shadows for extended light sources.

Brotman and Badler [Brotman84] came up with a soft shadow version of Crow's algorithm where they generated shadow volumes for a number of light source samples and computed the overlap using a depth buffer algorithm.

Nishita et al. [Nishita85] constructed penumbra and umbra volumes for linear light sources. Since the analytic computation of these volumes is very expensive, the method is not suitable for a real-time implementation.

Akenine-M¨oller and Assarsson [Akenine-M¨oller02a, Assarsson03] presented a soft shadow volume method based on *penumbra wedges*. Instead of extruding a single quad for each of the occluder's silhouette edge, a wedge is constructed which represents the penumbra volume. Although the construction of penumbra wedges can be quite complicated, the rendering of wedges can be implemented using graphics hardware, which results in real-time frame rates.

Figure 3.3.1: Penumbra wedges based approach for soft shadow volumes introduced in Ulf Assarsson's PhD dissertation

Chapter 4
Conclusions, Future works

4.1 Conclusions

In this dissertation, we actually have tried to introduce the different techniques of shadow rendering. We have also discussed the underlying theories, algorithms and terminology along with pictures of different techniques from acknowledged source. Our main aim was to understand the background of shadow rendering and give quick overview of the techniques.

As this is a dissertation of bachelor degree, we took it as a ground of the future work. So, we have not introduced any new techniques or modification of any of these techniques rather we have covered all the major techniques of shadow rendering techniques.

The two kinds of shadow rendering have been discussed, one is hard shadow rendering and the other is soft shadow rendering. We also discussed the terminology to understand the different techniques of shadow rendering. Major techniques of hard shadow rendering have been discussed with theories, algorithms and pictures that have been rendered by different research labs. Major soft shadow rendering techniques have been also discussed with theories, algorithms and pictures to understand the underlying process with the techniques.

4.2 Future Works

Nowadays, researchers are working on the soft shadow rendering because of high demand in games and animation movies. Soft shadows are more realistic than hard shadow because of penumbrae. So, the researchers are now introducing the new techniques of soft shadow rendering and also modifying the old techniques to give the more realistic shadows than ever.

Recently, Ulf Assarsson from *School of Computer Science and Engineering* Chalmers University of Technology has done his PhD on soft shadow rendering. He has introduced couple of new techniques of soft shadow rendering with the shadow volume approach which was first introduced by Crow in 1977.
In the dissertation, Ulf Assarsson has introduced new algorithms for soft shadow volumes. He has also introduced new terminology for soft shadows which is "Penumbra Wedges". Although the construction of penumbra wedges can be quite complicated, the rendering of wedges can be implemented using graphics hardware, which results in real-time frame rates.

So, the future works regarding the soft shadow volumes may be we can eliminate the penumbra wedge techniques and can introduce any new technique to do more realistic real time shadow rendering algorithms for soft shadow volumes.

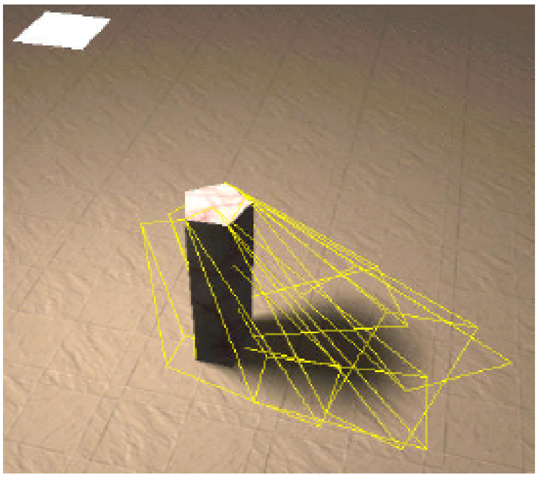


Figure 6: The wedges for a simple scene. At 512 × 512 resolution, this image was rendered at 5 frames per second using our software implementation. Note that there are wedges whose adjusted top edges differ a lot from the original silhouette edge. This can especially be seen to the left of the cylinder. The reason for this is that the vertices of the original silhouette edge are positioned with largely different distances to the light source.

Figure 4.2.1: Penumbra wedges based approach for soft shadow volumes introduced in Ulf Assarsson's PhD dissertation

References

[1] Stefan Brabec. ***Shadow Techniques for Interactive and Real-Time Applications***. PhD Dissertation, Max-Planck-Institut f ¨ ur Informatik Saarbr ¨ucken, Germany, 2003.

[2] Ulf Assarsson. ***A Real-Time Soft Shadow Volume Algorithm.*** PhD Dissertation, *School of Computer Science and Engineering,* Chalmers University of Technology, Sweden, 2003.

[3] Eric Chan, Fr´edo Durand. ***An Efficient Hybrid Shadow Rendering Algorithm.*** Eurographics Symposium on Rendering (2004) Computer Science and Artificial Intelligence Laboratory Massachusetts Institute of Technology, 2004.

[4] Wolfgang Heidrich. ***Shadow Mapping and Shadow Volumes: Recent Developments in Real-Time Shadow Rendering.*** Project Report. CS514 Advanced Computer Graphics: Image Based Rendering, University of British Columbia, 2004.

[5] Timo Ahokas. ***Shadow Maps.*** Helsinki University of Technology

[6] Joen Sindholt. ***A comparison of Shadow Algorithms.*** MS Dissertation. Informatics and Mathematical Modelling Technical University of Denmark, Lyngby, Denmark, May 2005.

[7] Samuli Laine. ***EFFICIENT PHYSICALLY-BASED SHADOW ALGORITHMS.*** PhD Dissertation. Helsinki University of Technology Publications in Telecommunications Software and Multimedia, 2006.

[8] Thomas Annen. ***Advanced Shadow Map Parameterization.*** Diploma Dissertation. Max-Planck-Institut f ¨ur Informatik Saarbr¨ucken, Germany, 2002.

[9] Samuli Laine, Timo Aila, Ulf Assarsson, Jaakko Lehtinen, Tomas Akenine M¨oller. ***Soft Shadow Volumes for Ray Tracing,*** SIGGRAPH 2005.

[10] Timo Aila, Tomas Akenine-Möller. ***A Hierarchical Shadow Volume Algorithm.*** Graphics Hardware (2004).

[11] Timo Aila and Samuli Laine. ***Alias-Free Shadow Maps.*** Eurographics Symposium on Rendering (2004).

[12] Jaakko Lehtinen, Samuli Laine, Timo Aila. ***An Improved Physically - Based Soft Shadow Volume Algorithm,*** EUROGRAPHICS 2006.

[13] Tom Lokovic, Eric Veach. ***Deep Shadow Maps.*** Pixar Animation Studios.

[14] Maneesh Agrawala, Ravi Ramamoorthi, Alan Heirich, Laurent Moll. ***Efficient Image - Based Methods for Rendering Soft Shadows.*** SIGGRAPH 2006.

[15] Randima Fernando, Sebastian Fernandez. Kavita Bala. Donald P Greenberg. ***Adaptive Shadow Maps.*** Cornell University, SIGGRAPH 2005.

[16] Marc Stamminger and George Drettakis. ***Perspective Shadow Maps.*** REVES - INRIA Sophia-Antipolis, France.

[17] Pradeep Sen, Mike Cammarano, Pat Hanrahan***. Shadow Silhouette Maps,*** Stanford University, SIGGRAPH 2003.

[18] Kenneth P. Herndon, Robert C. Zeleznik, Daniel C. Robbins, D. Brookshire Conner, Scott S. Snibbe and Andries van Dam, ***Interactive Shadows.*** Brown University, SIGGRAPH 1994.

[19] Morgan McGuire, John F. Hughes, and Kevin T. Egan***. Fast, Practical and Robust Shadows.*** November 6, 2003.

[20] Morgan McGuire***. Observations on Silhouette Sizes.*** Brown University, April 9, 2003.

[21] Naga K. Govindaraju, Brandon Lloyd, Sung-Eui Yoon, Avneesh Sud, Dinesh Manocha. ***Interactive Shadow Generation in Complex Environments.*** University of North Carolina at Chapel Hill, SIGGRAPH 2005.

[22] [Web References]
http://www.googlesyndicatedsearch.com/u/browncs?q=shadow
http://www.cs.brown.edu/~tor/
http://www.peachpit.com/articles/article.asp?p=486505&seqNum=5&rl=1
http://graphics.stanford.edu/courses/cs348b-96/global/
http://www.cs.virginia.edu/~gfx/Courses/2004/RealTime/
http://www2.imm.dtu.dk/pubdb/views/publication_details.php?id=3952
http://www.graphics.cornell.edu/pubs/2000/PFG00.html
http://www.graphics.stanford.edu/
http://sajeeb.googlepages.com/
http://en.wikipedia.com