Overview of Shadow Rendering

Brandon Lloyd

April 9, 2007

With some slides and images cannibalized from Han-Wei Shen
http://www.cse.ohio-state.edu/~hwshen/781/

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL
Outline

• Introduction
• Planar Shadows
• Shadow Volumes
• Shadow Maps
• Soft Shadows
Why are shadows important?

- Improved understanding of an object’s shape and position.
- Realism
What is a shadow?

- Shadows occur on surfaces that are not fully visible the light.
Terminology

- umbra – fully shadowed region
- penumbra – partially shadowed region
“Hard” and “Soft” Shadows

- Depends on the type of light sources
  - Point or Directional ("Hard Shadows")
  - Area ("Soft Shadows")
“Hard” and “Soft” Shadows

- Hard shadow
  - *point* light source

- Soft shadow
  - *area* light source
Outline

- Introduction
- Planar Shadows
- Shadow Volumes
- Shadow Maps
- Soft Shadows
Planar Shadows

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

\[ \mathbf{n} \cdot \mathbf{x} + d = 0 \]
Issues with planar shadows

- **Shadow polygon generation (z fighting)**
  - Add an offset to the shadow polygons (glPolygonOffset)
  - Draw receivers first, turn z-test off, then draw the shadow polygons. After this, draw the rest of the scene.

- **Shadow polygons fall outside the receiver**
  - Using stencil buffer – draw receiver and update the stencil buffer

- **Shadows have to be rendered at each frame**
  - Render into texture

- **Restricted to planar objects**
Outline

• Introduction
• Shadow Volumes
• Shadow Maps
• Soft Shadows
2D Cutaway of a Shadow Volume

Light source

Eye position
(note that shadows are independent of the eye position)

Shadowing object

Partially shadowed object

Surface outside shadow volume (illuminated)

Surface inside shadow volume (shadowed)

Shadow volume (infinite extent)
Overview

1. Construct shadow volumes from occluders

2. Determine if each visible point lies inside a shadow volume
Stenciling counting

Eye

Zero +1 +2 +2 +3

Zero +1 +2 +3
Illuminated, behind (zpash)

Unshadowed object
Shadowed (zpass)
Illuminated, in front (zpass)

Shadowed object

zero

+1

+2

+2

+1

+3

zero
Problems with near clip plane

Missed shadow volume intersection due to near clip plane clipping; leads to mistaken count

Near clip plane

Far clip plane

eye

+1
+2
+3
+2
+1
+1
zero
zero

The UNIVERSITY of NORTH CAROLINA at CHAPEL HILL
Illuminated, behind (zfail)
Shadowed (zfail)

Eye position

Zero +1 +2 +3

Shadowed object
Illuminated, in front (zfail)
Constructing shadow volumes

• Find silhouette edges
  - Brute force

• Extrude edges to infinity
  - homogeneous coordinates (w=0)

• \textit{zfail} needs caps
  - Near caps = tris facing light
  - Far caps = near caps rendered at infinity
  - Needs projection matrix with zfar = \infty
Algorithm

• Render scene with ambient lighting
• Set stencil op:
  – INC on front faces, DEC on backfaces
• Render shadow volumes
• Disable writes where stencil ≠ 0
• Render scene with full lighting
Comparison with shadow maps

• **Advantages**
  - No aliasing or bias problems
  - Omnidirectional lights

• **Disadvantages**
  - Model must be polygonal
  - Requires extra passes
  - FILL-RATE CONSUMPTION
Fill-rate consumption

Shadow volumes in transparent yellow

Final image
Clamped Shadow Volumes

Standard shadow volumes

Discrete clamping
Video

Standard Shadow Volumes  CC Shadow Volumes
Outline

• Introduction
• Planar Shadows
• Shadow Volumes
• **Shadow Maps**
  - Basic algorithm
  - Self shadowing
  - Aliasing
• Soft Shadows
Shadow maps

Use the depth map in the light view to determine if sample point is visible

Point in shadow visible to the eye by not visible to the light
Recipe

• Render light-view to a depth texture
• Setup up eye view matrices
• Setup texture coordinates
  – glTexGen …
  – EYE_LINEAR mode
  – Specify identity matrix (multiplied by inverse model view)
Recipe

- **Setup texture matrix**
  
  \[
  \begin{bmatrix}
  0.5 & 0 & 0 & 0.5 \\
  0 & 0.5 & 0 & 0.5 \\
  0 & 0 & 0.5 & 0.5 \\
  0 & 0 & 0 & 1
  \end{bmatrix}
  \times L_P \times L_{MV}
  \]

- **Enable shadow map texturing (depth compare)**
Result

False self-shadowing

Image courtesy of Tomas Akenine-Möller
Solution - Bias

Offset the depth values by using `glPolygonOffset()`.

- Constant
- Slope dependent
Correct bias is non-trivial

Too much bias – floating shadow

Image courtesy of Tomas Akenine-Möller
Aliasing

Well-defined edge

Light pointing toward viewer

Blocky artifacts

Image courtesy of NVIDIA
Percentage Closer Filtering

- Average binary results of all depth map pixels covered
- Soft anti-aliased shadows
Poor sample distribution

Images courtesy of Stamminger and Drettakis 02
Solution: Warp depth image

Images courtesy of Stamminger and Drettakis 02
How to warp?

Stamminger and Drettakis 02

Martin and Tan 04

Wimmer et al. 04

Chong and Gortler 04
Outline

• Introduction
• Planar Shadows
• Shadow Volumes
• Shadow Maps

• Soft Shadows
  – Overview
  – Sampling methods
  – Heuristic methods
  – Physically-based methods
Overview

- Point lights are either visible or not
- Area lights can be partially visible
The Idea

Composite hard shadows into soft shadows
• Use texture mapping for display

Many Hard Shadow Textures
Accumulation Buffer

Soft Shadow Texture
Texture mapped into the scene

Area light source
Occluder
Receiver Polygon
Sampling methods

4 samples

1024 samples
Sampling methods

• Advantages:
  - Converges to correct solution
  - Can use existing techniques (shadow maps and shadow volumes)

• Disadvantages:
  - SLOW! Takes many samples to converge
Heuristic methods

• **Advantages:**
  - Capture qualitative appearance
  - Can be very fast

• **Disadvantages:**
  - Never will converge to correct solution
Heuristic methods

- shadow map
- umbra region
- light
- blocker
- shadow map
- umbra region

\[ d_1 \]
\[ d_2 \]