Textures for Data Storage: Shadows

CSCI 4830/7000
Advanced Computer Graphics
Spring 2010

Shadows in Computer Graphics

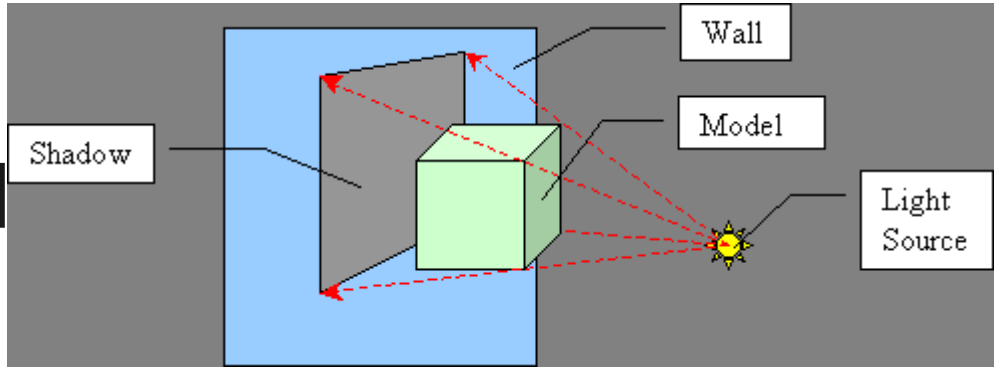
- Shadows are important to realism
  - Depth cues
  - Relative positions of objects
- Doesn't “just happen” when lighting is turned on
  - Nor is there a glEnable(GL_SHADOWS)
- Scene must be rendered 2-4 times
- Shader implementation can be efficient
  - Draw once every time the light or scene changes
  - Draw once for every eye position

Shadow Examples

- Planar Shadows (ex32)
  - Shadows on the floor only
- Shadow Volumes (ex33)
  - True shadow, very hard
- Shadow Maps (ex34)
  - True shadows, depth in textures
- Shader Shadow Map (ex45)
  - Fast implementation via shader

Planar Shadows

- Projects object on surface
- Simplest shadows
- Fast but very limited
- The problem:
  - Surface defined by point *E* and normal *N*
  - L is the light
  - *P* is on the object
  - Find *P*' the projection of *P* on the surface



Extend $L\vec{P}$ to P'

$$P' = L + \lambda(P - L)$$

Let P' be in the plane

$$(P' - E) \cdot N = 0$$

Expand P' to

$$(L + \lambda(P - L) - E) \cdot N = 0$$

Then

$$\lambda = \frac{(E - L) \cdot N}{(P - L) \cdot N}$$

so that

$$P' = L + \frac{(E - L) \cdot N}{(P - L) \cdot N}(P - L)$$

Define

$$e = E \cdot N, \quad l = L \cdot N, \quad c = (E - L) \cdot N = e - l$$

Then

$$P' = L + \frac{c}{P \cdot N - l}(P - L)$$

Therefore

$$x' = \frac{(N_x L_x + c)P_x + (N_y L_x)P_y + (N_z L_x)P_z - eL_x}{N_x P_x + N_y P_y + N_z P_z - l}$$

$$y' = \frac{(N_x L_y)P_x + (N_y L_y + c)P_y + (N_z L_y)P_z - eL_y}{N_x P_x + N_y P_y + N_z P_z - l}$$

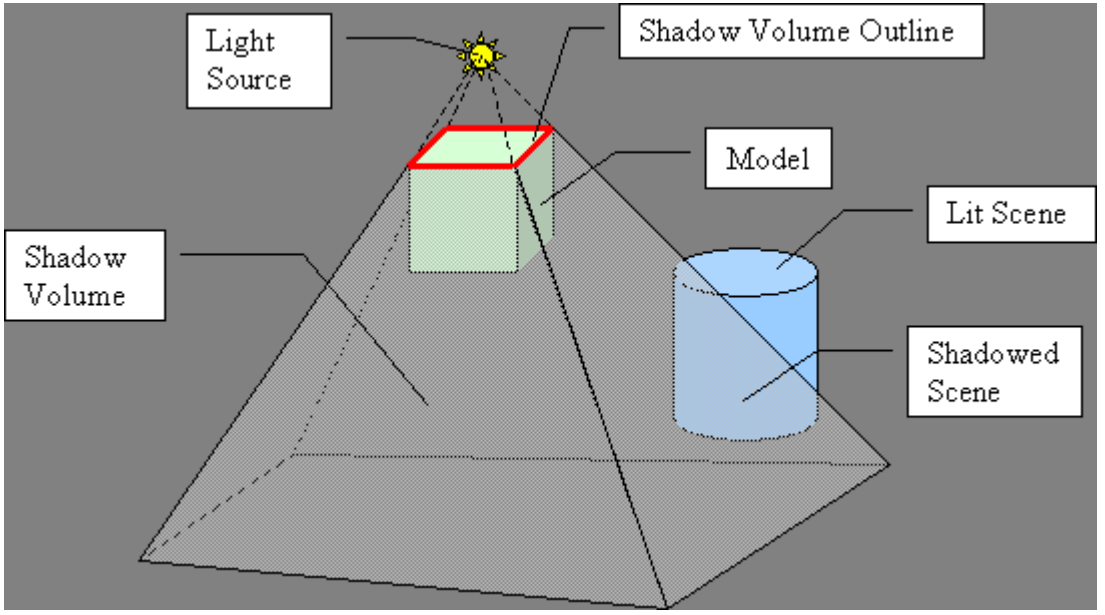
$$z' = \frac{(N_x L_z)P_x + (N_y L_z)P_y + (N_z L_z + c)P_z - eL_z}{N_x P_x + N_y P_y + N_z P_z - l}$$

so that

$$\begin{bmatrix} x' \\ y' \\ z' \\ w' \end{bmatrix} = \begin{bmatrix} L_x N_x + c & L_x N_y & L_x N_z & -eL_x \\ L_y N_x & L_y N_y + c & L_y N_z & -eL_y \\ L_z N_x & L_z N_y & L_z N_z + c & -eL_z \\ N_x & N_y & N_z & -l \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

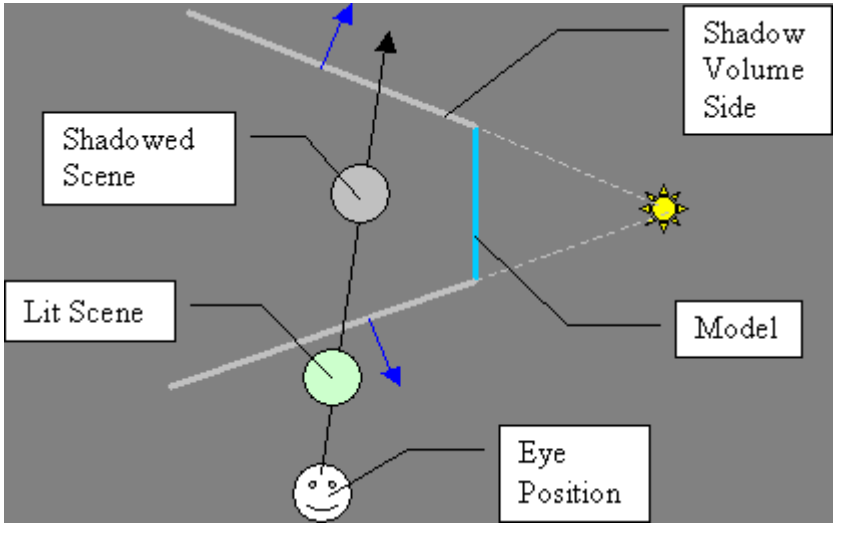
Shadow Volumes

- The volume corresponding to the shadow cast by a facet of each object
  - Potentially multiple shadow volumes per object
  - Shadow of the object is the combination of all shadow volumes for the object



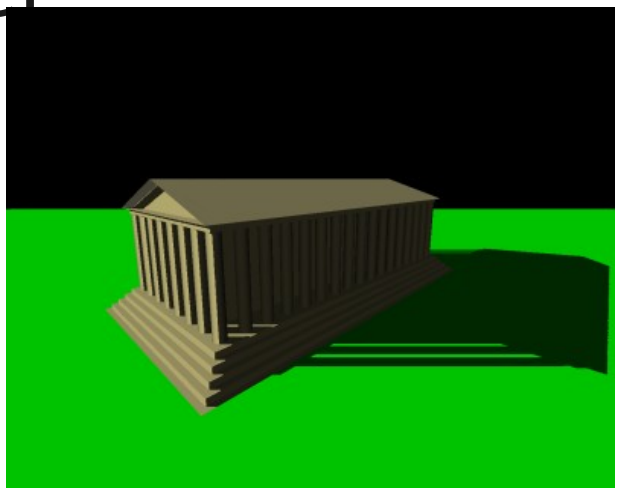
Shadow Volume Algorithm

- Count transitions in and out of shadow volumes
  - Increment of in, decrement for out
  - Similar to polygon winding rule for in/out
- Lit areas has value of zero (initial value)



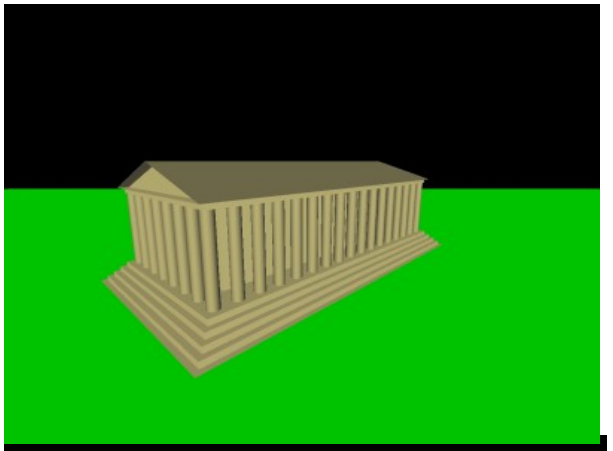
Shadow Mapping

- Project with light as viewpoint
- Depth buffer from light
- Light/shadow determined just like visibility
  - Objects in light foremost in depth buffer
  - Objects in shadow depth obscured
- Requires second depth buffer
  - Store depth to texture
  - Compare R to texture
- In OpenGL extensions
- Used in *Toy Story* etc.

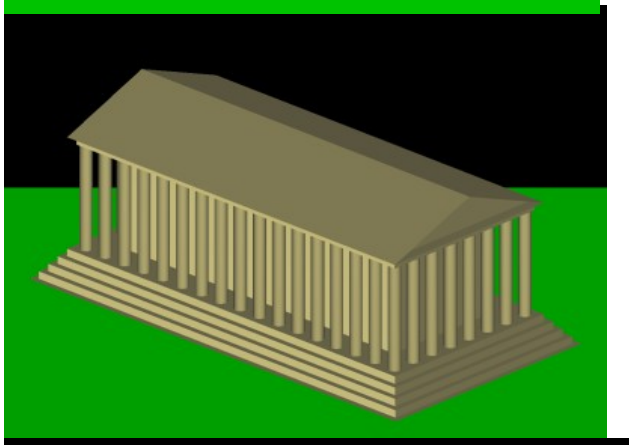


Shadow Map Example

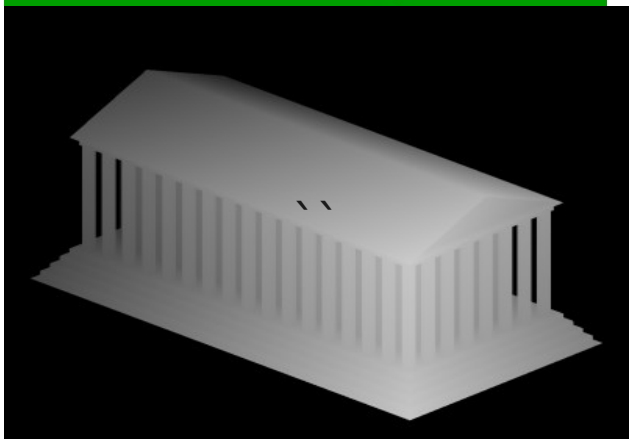
No Shadows



Light View



Light View Depth



Shadow Map Shader

- Draw shadow map
  - Bind framebuffer to depth texture
  - Draw scene with eye at light to generate depths
  - Update if light or scene changes
- Draw scene
  - Generate texture coordinates with light PoV
  - Compare depth (R) with depth texture
    - R=depth means lit – light as normal
    - R>depth means shadowed – ambient light only
- Fast, Simple, Realistic