Shadow Algorithms

- Realistic illumination includes shadows cast by objects
- Simple shadow generation methods:
  - Shadow maps
  - Shadow volumes
- Advanced methods:
  - Ray-tracing
  - Radiosity

Shadow Map Algorithm

- Object is in shadow if not “seen” by light source
- Idea – compute the discrete visibility of the scene from light source to decide if a pixel is shadowed

procedure SHADOWMAPPING
render z buffer from light’s point of view to depth map D
store D as texture map d(u,v)
render scene S from the eye’s point of view into image A
for each rasterized pixel of A with texture coords (u,v) in eye space and transformed distance z(u,v) do
  if d(u,v) < z(u,v) then
    pixel is shadowed
  end if
end for

Copyright
C. Gotsman, G. Elber, M. Ben-Chen
Computer Science Dept., Technion
Summary

- Shadow map algorithm can approximate the shadows of any scene which can be rendered using a Z-buffer.
- Requires separate Z-buffer for each light source.
- Every polygon is rendered \( N+1 \) times (for \( N \) light sources).

Disadvantage

- Image space algorithm: severe aliasing can occur if the light source is at orientation significantly different from the viewpoint.

Shadow Volume Algorithm

- The shadow boundary separates illuminated and shaded regions.
- Compute as extrusion of silhouettes along light direction.
- Compute intersection of extruded volume with other objects.

Shouettes

The edges between front-facing polygons and back-facing polygons.
Shadow Volumes

- Region outside shadow volume (illuminated)
- Region inside shadow volume (shadowed)

The Shadowed Regions

- Scene polygon
- Silhouette polygon
- Front facing shadow polygons cause object behind to be shadowed
- Back facing shadow polygons cancel effect of front facing ones
- Usually implemented using stencil buffer

Properties of Shadow Volumes

- Object space algorithm - does not depend on viewpoint
- High complexity per object, function of scene
- Requires geometric methods
  - Silhouette computation
  - Extrusion

Shadow Volume Algorithm

- For each object and light source compute object silhouette (and boundary if open) from light source viewpoint
- Extend each silhouette (and boundary) to form semi-infinite volumes
- Feed boundaries into regular Z-buffer as fully transparent polygons
- Front facing shadow polygons cause object behind to be shadowed
- Back facing shadow polygons cancel effect of front facing ones
- Usually implemented using stencil buffer