CSE 167:
Introduction to Computer Graphics
Lecture #14: ShadowMapping

Jürgen P. Schulze, Ph.D.
University of California, San Diego
Fall Quarter 2014

Announcements

- Project 5 late grading and Project 6 due Friday Dec 5
- Project 6 is on-line
- Midterm review at Center Hall 105 at 5pm TODAY

OpenGL Debugging

- Visual Studio's built-in debugger
- cerr << "Message" << endl;
  - Use cerr instead of cout
- If (glGetError() != GL_NO_ERROR) cerr << "GL error" << endl;

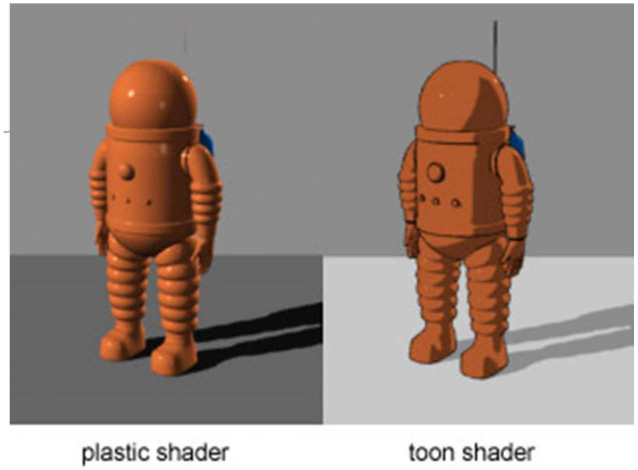
Lecture Overview

Advanced Shader Effects

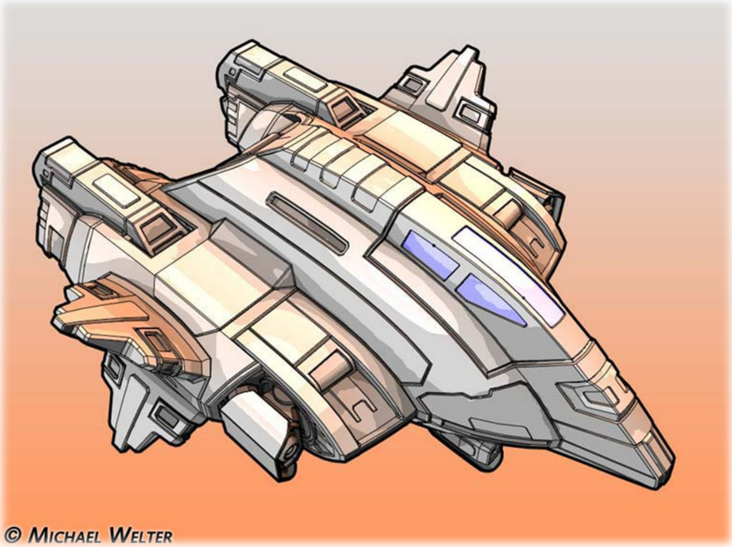
- Toon shading

Toon Shading

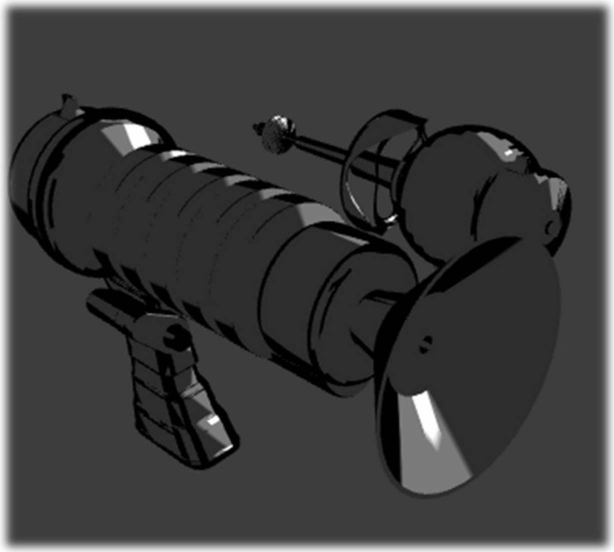
- A.k.a. Cel Shading ("Cel" is short for "celluloid" sheets, on which animation was hand-drawn)
- Gives any 3D model a cartoon-style look
- Emphasizes silhouettes
- Discrete steps for diffuse shading, highlights
- Non-photorealistic rendering method (NPR)
- Programmable shaders allow real-time performance



Source: Wikipedia



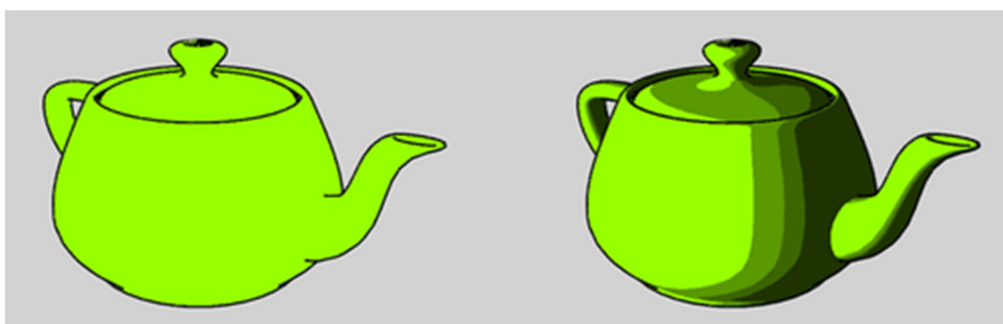
Off-line toon shader



GLSL toon shader

Approach

- Start with regular 3D model
- Apply two rendering tricks:
  - Silhouette edges
    - Emphasize pixels with normals perpendicular to viewing direction.
  - Discretized shading
    - Conventional (smooth) lighting values calculated for each pixel, then mapped to a small number of discrete shades.

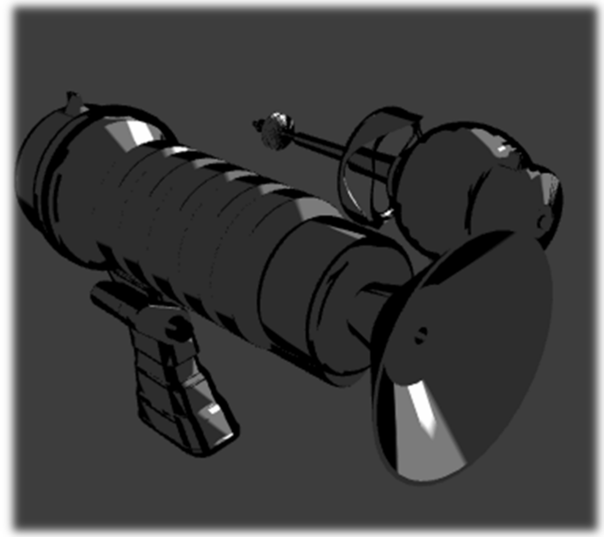


Source: Wikipedia

Silhouette Edges

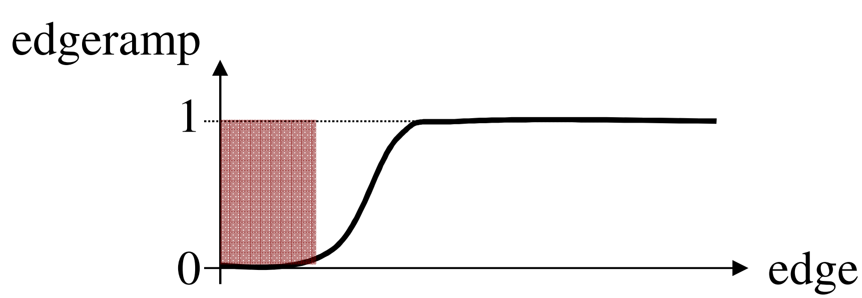
- Silhouette edge detection
  - Compute dot product of viewing direction **v** and normal **n**

$$\text{edge} = \max(0, \mathbf{n} \cdot \mathbf{v})$$



  - Use 1D texture to define edge ramp

uniform sample1D edgeramp; e=texture1D(edgeramp,edge);



UCSD

Discretized Shading

- Compute diffuse and specular shading

$$\text{diffuse} = \mathbf{n} \cdot \mathbf{L} \qquad \text{specular} = (\mathbf{n} \cdot \mathbf{h})^s$$

- Use 1D textures diffuseramp, specularramp to map diffuse and specular shading to colors
- Final color:

```
uniform sampler1D diffuseramp;
uniform sampler1D specularramp;
c = e * (texture1D(diffuse,diffuseramp) +

texture1D(specular,specularramp));
```

Toon Shading Demo

http://www.bonzaisoftware.com/npr.html

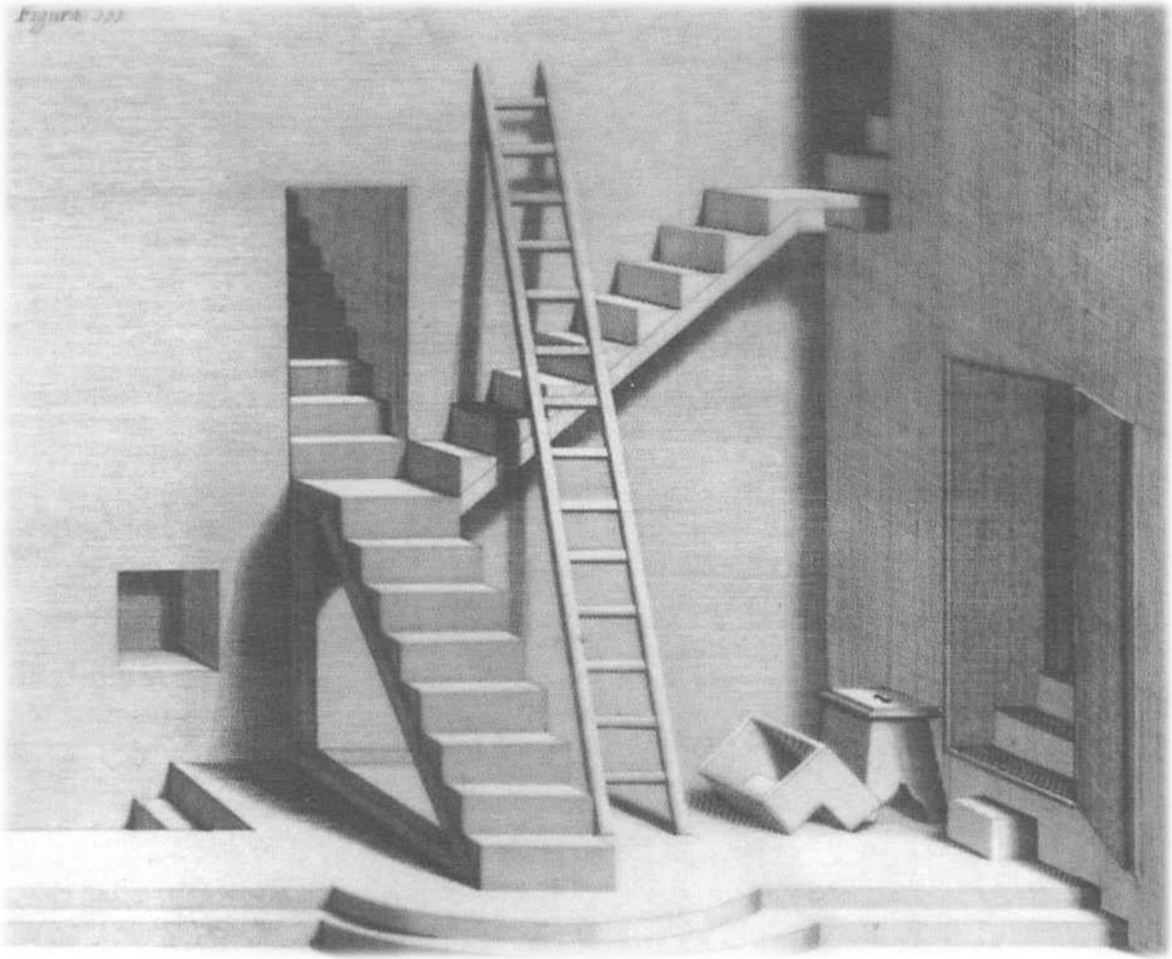
UCSD

Lecture Overview

- Shadows
- Shadow Mapping

Why Are Shadows Important?

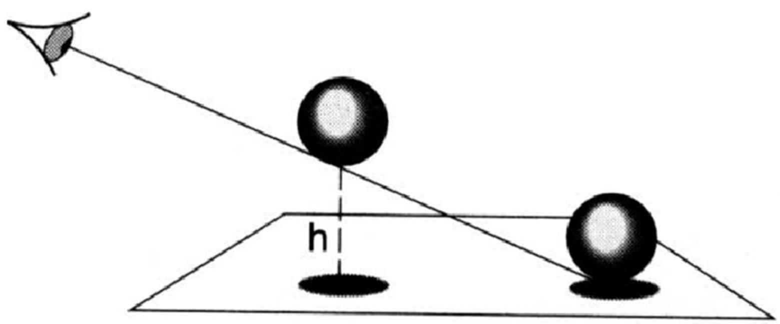
- Give additional cues on scene lighting

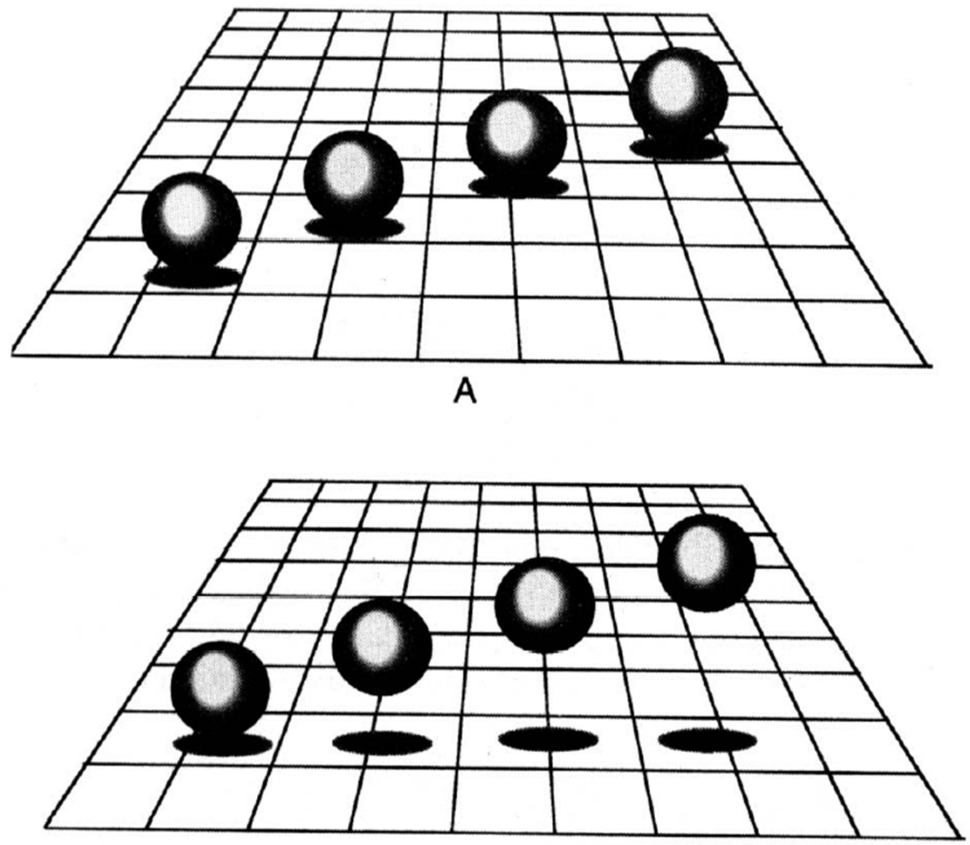


UCSD

Why Are Shadows Important?

- Contact points
- Depth cues

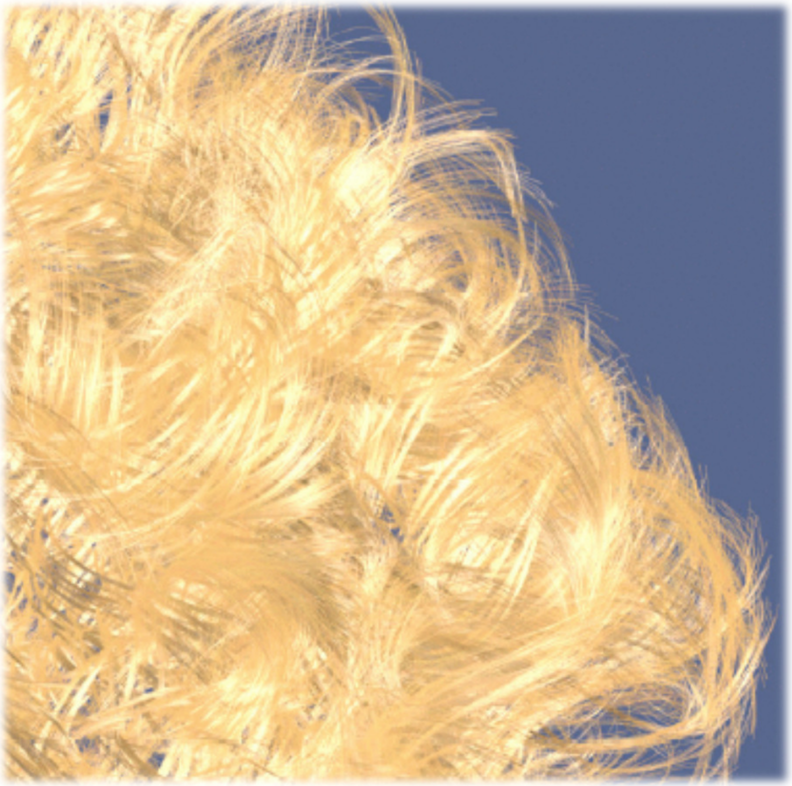




UCSD

Why Are Shadows Important?

- Realism

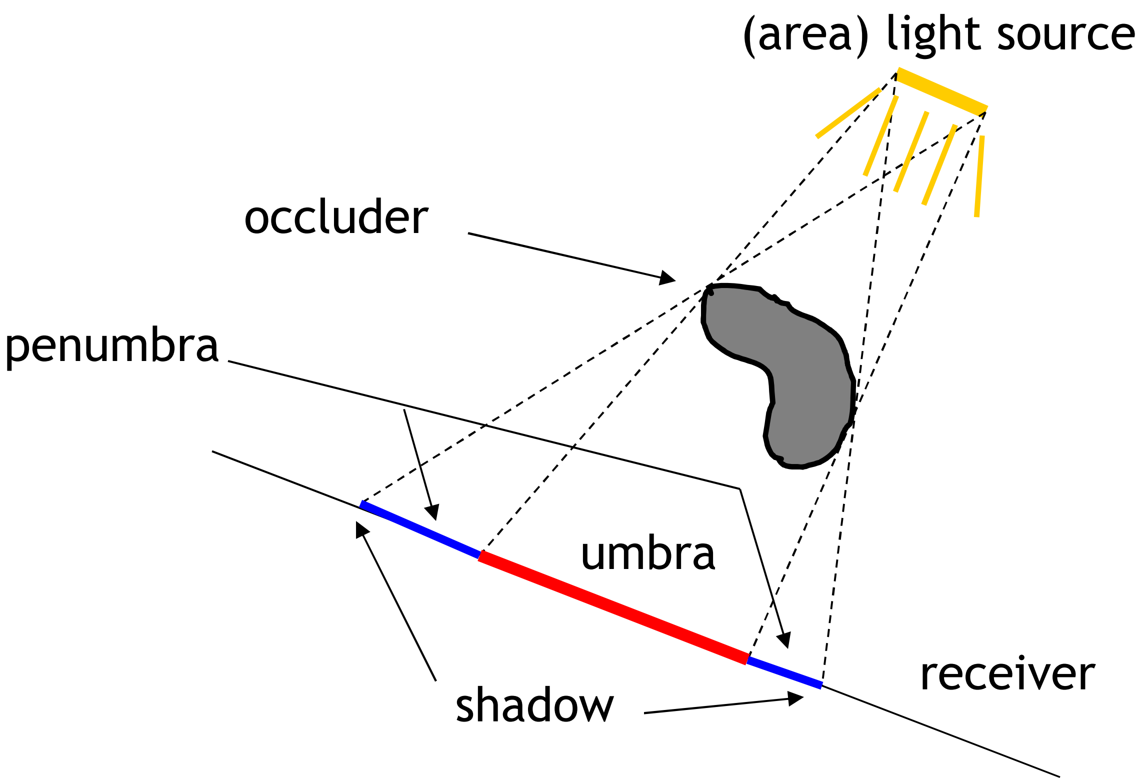


Without self-shadowing

With self-shadowing

Terminology

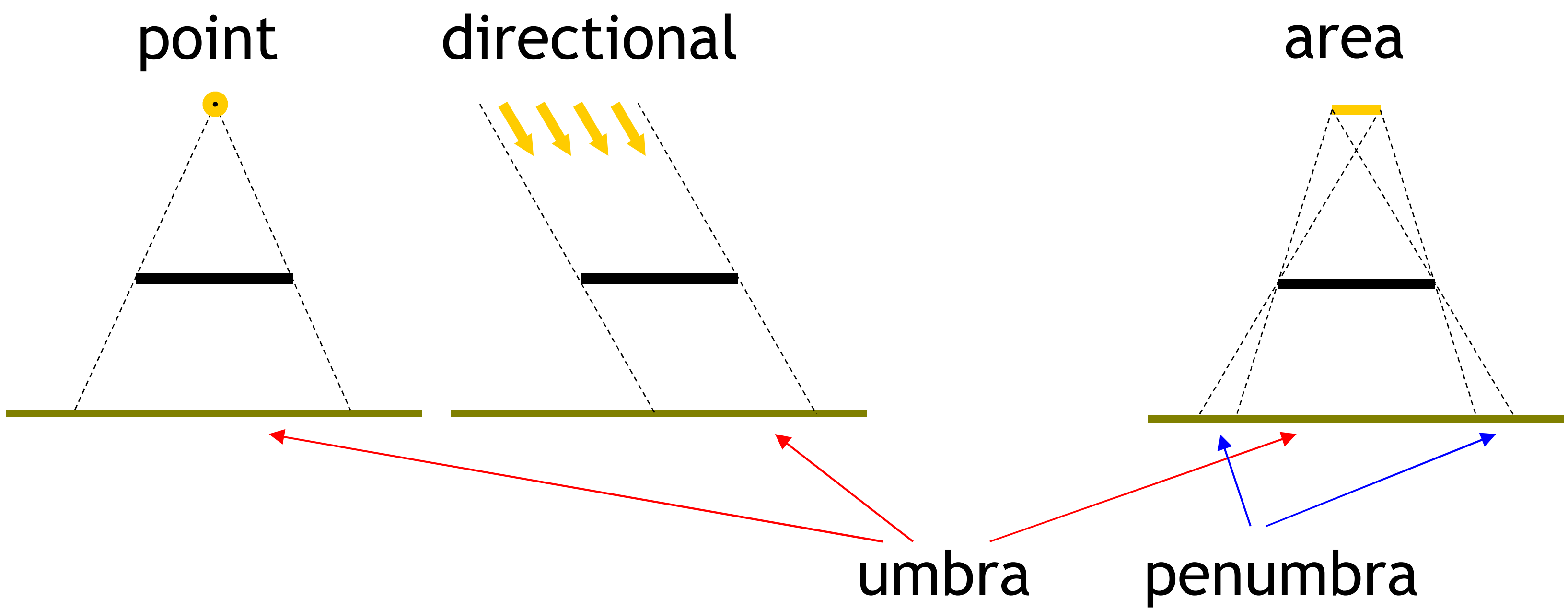
- Umbra: fully shadowed region
- Penumbra: partially shadowed region



UCSD

Hard and Soft Shadows

- Point and directional lights lead to hard shadows, no penumbra
- Area light sources lead to soft shadows, with penumbra



Hard and Soft Shadows

Hard shadow from point light source

Soft shadow from area light source

Shadows for Interactive Rendering

- In this course: hard shadows only
  - Soft shadows hard to compute in interactive graphics
- Two most popular techniques:
  - Shadow mapping
  - Shadow volumes
- Many variations, subtleties
- Active research area

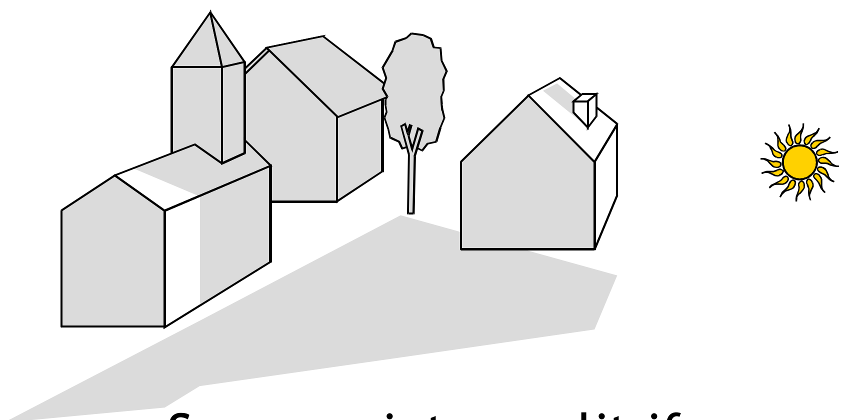
Lecture Overview

- Shadows
- Shadow Mapping

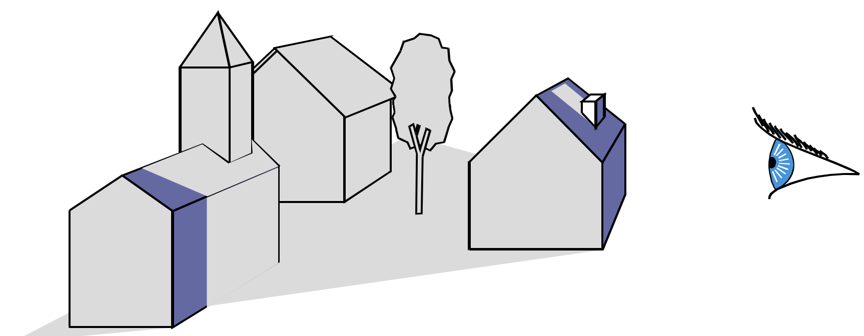
Shadow Mapping

Main Idea

- A scene point is lit by the light source if visible from the light source
- Determine visibility from light source by placing a camera at the light source position and rendering the scene from there



Scene points are lit if visible from light source



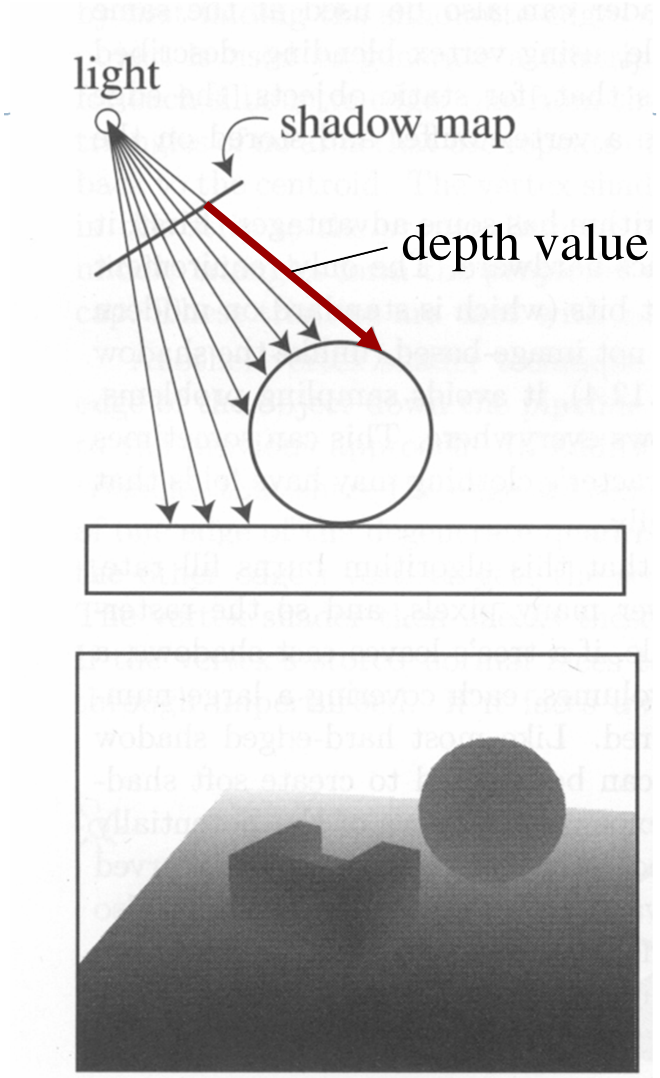
Determine visibility from light source by placing camera at light source position

UCSD

Two Pass Algorithm

First Pass

- Render scene by placing camera at light source position
- Store depth image (*shadow map*)

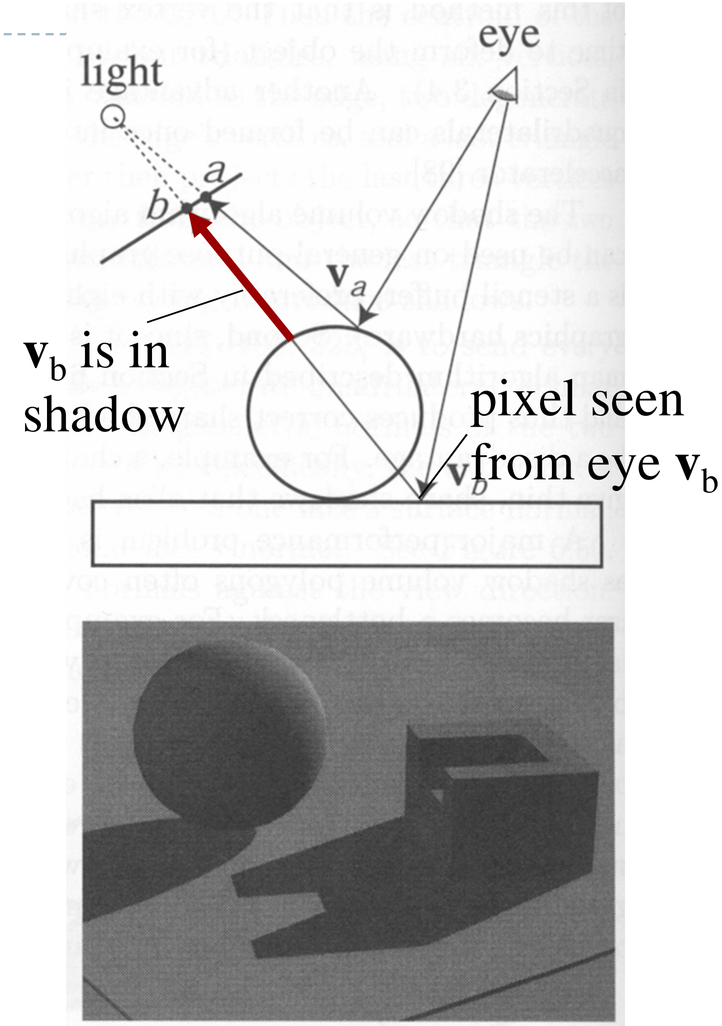


Depth image as seen from light source

Two Pass Algorithm

Second Pass

- Render scene from camera position
- At each pixel, compare distance to light source with value in shadow map
  - If distance is larger, pixel is in shadow
  - If distance is smaller or equal, pixel is lit



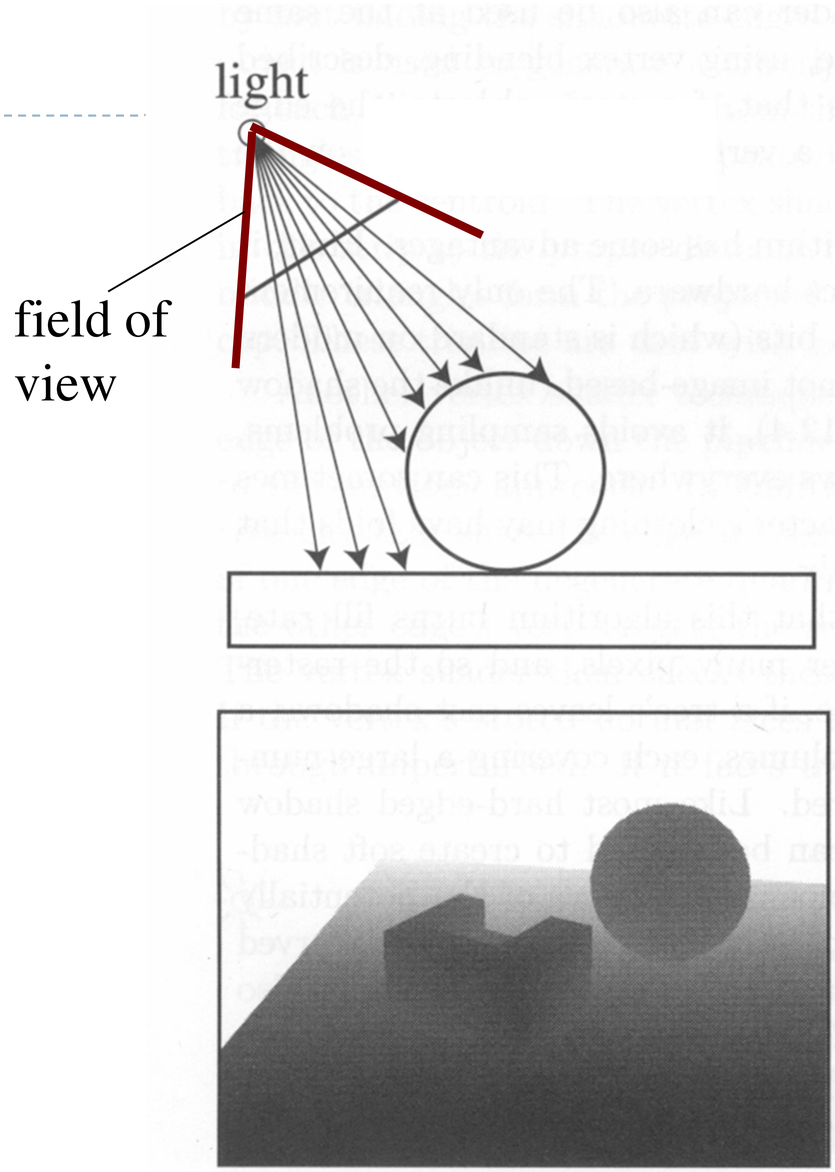
Final image with shadows

Issues With Shadow Maps

- Limited field of view of shadow map
- Z-fighting
- Sampling problems

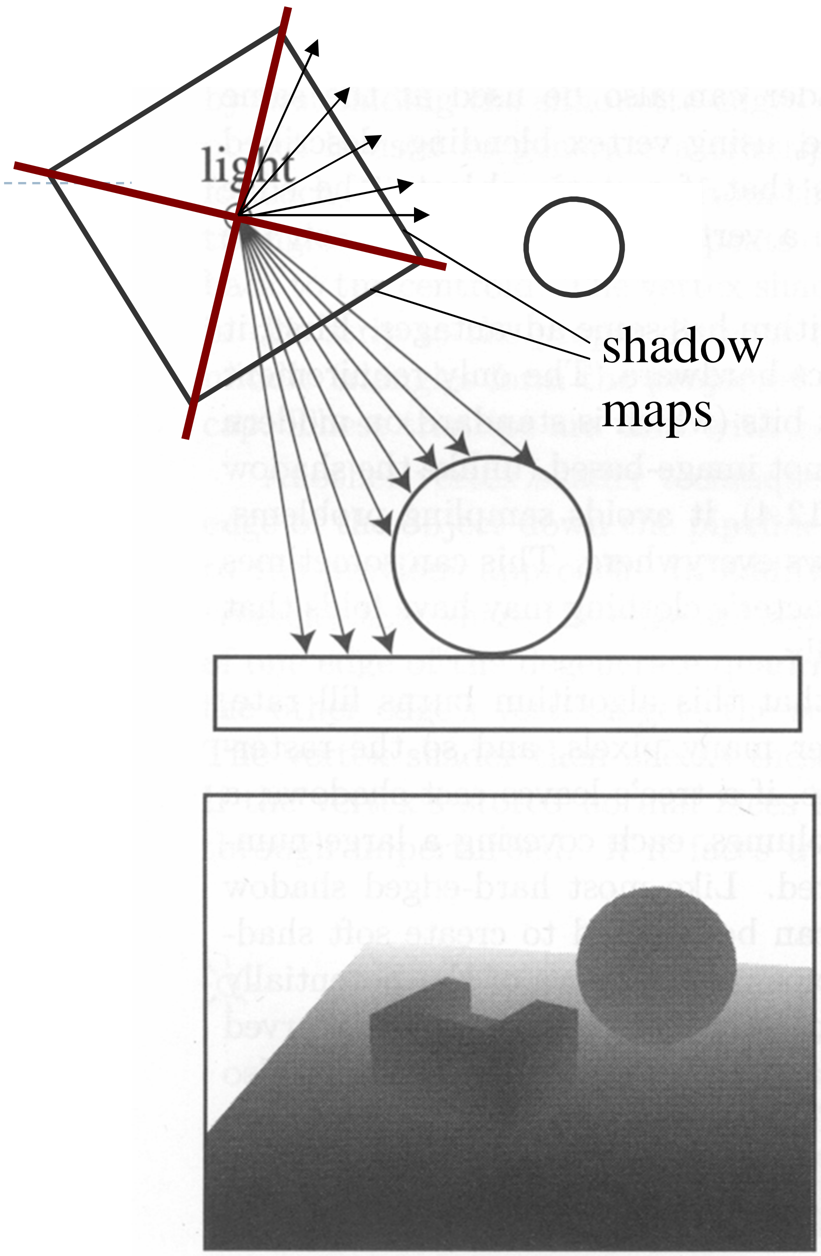
Limited Field of View

- What if a scene point is outside the field of view of the shadow map?



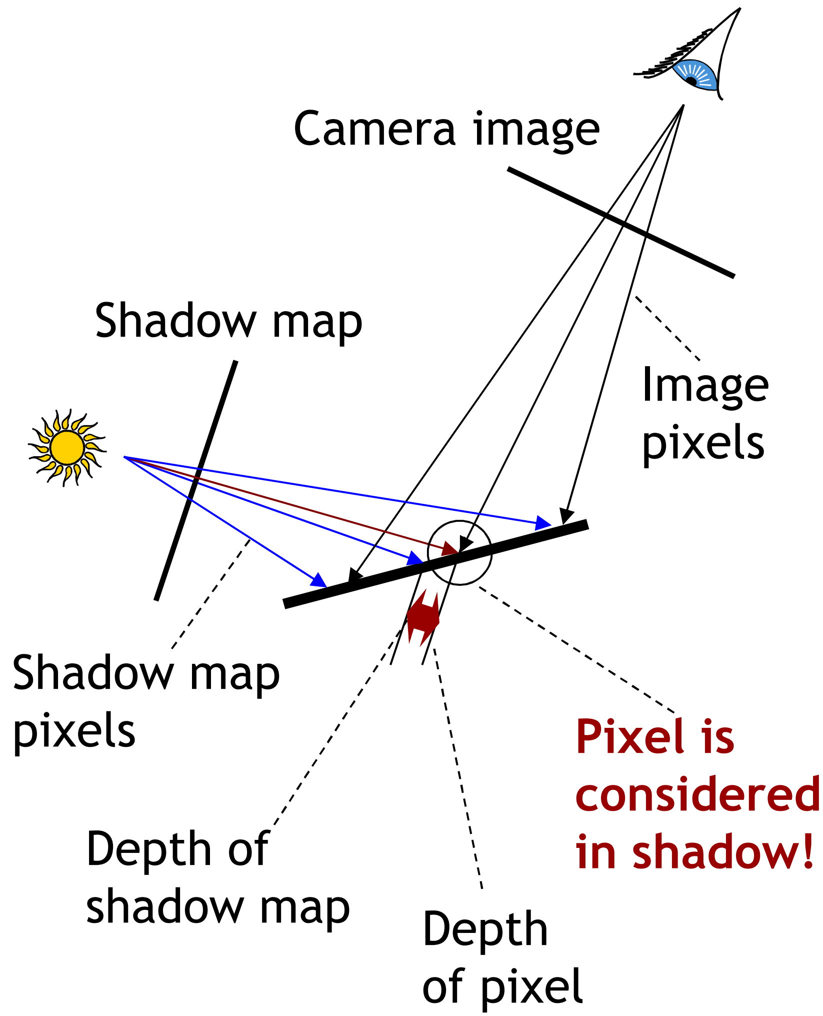
Limited Field of View

- What if a scene point is outside the field of view of the shadow map?
  → Use six shadow maps, arranged in a cube
- Requires a rendering pass for each shadow map



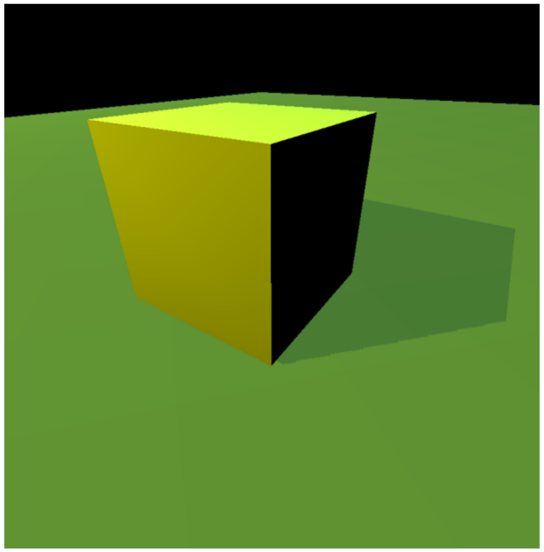
Z-Fighting

- Depth values for points visible from light source are equal in both rendering passes
- Because of limited resolution, depth of pixel visible from light could be larger than shadow map value
- Need to add bias in first pass to make sure pixels are lit

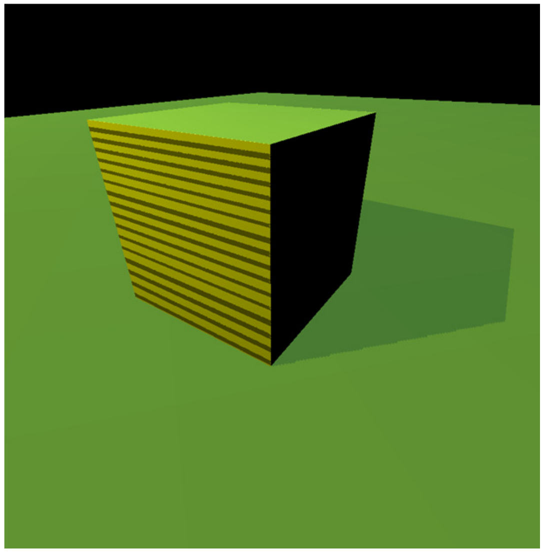


Solution: Bias

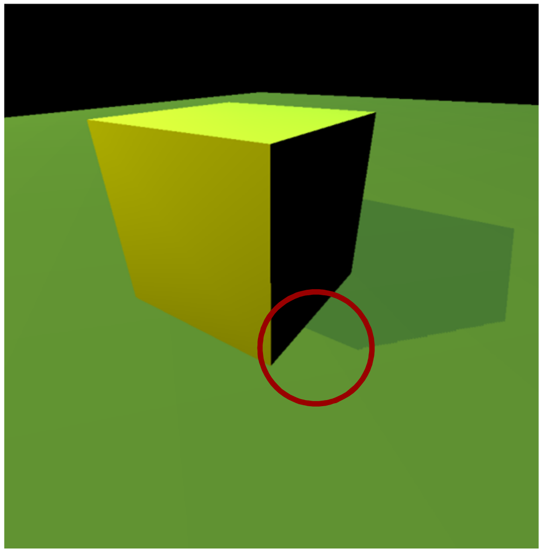
- Add bias when rendering shadow map
  - Move geometry away from light by small amount
- Finding correct amount of bias is tricky



Correct bias



Not enough bias

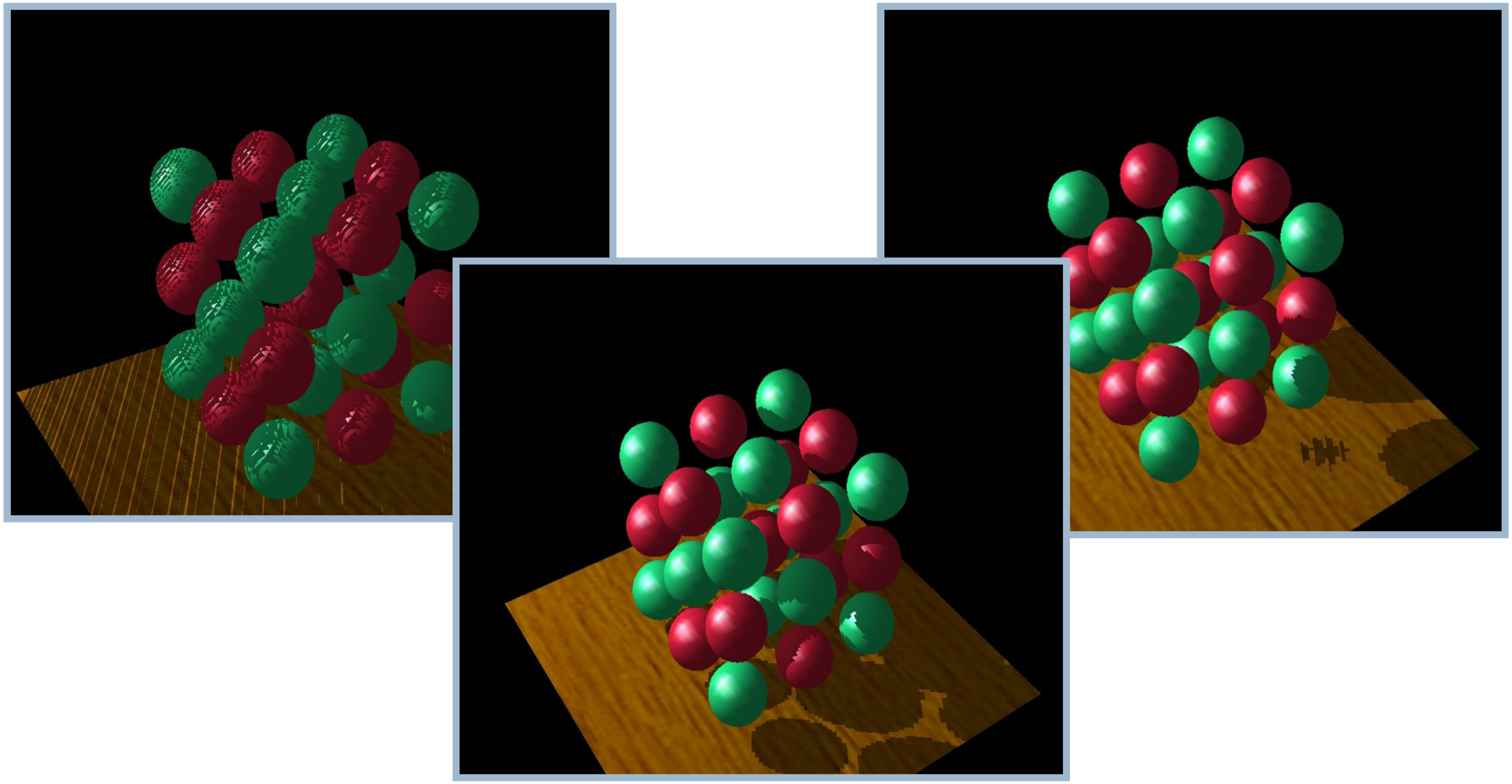


Too much bias

Bias Adjustment

Not enough

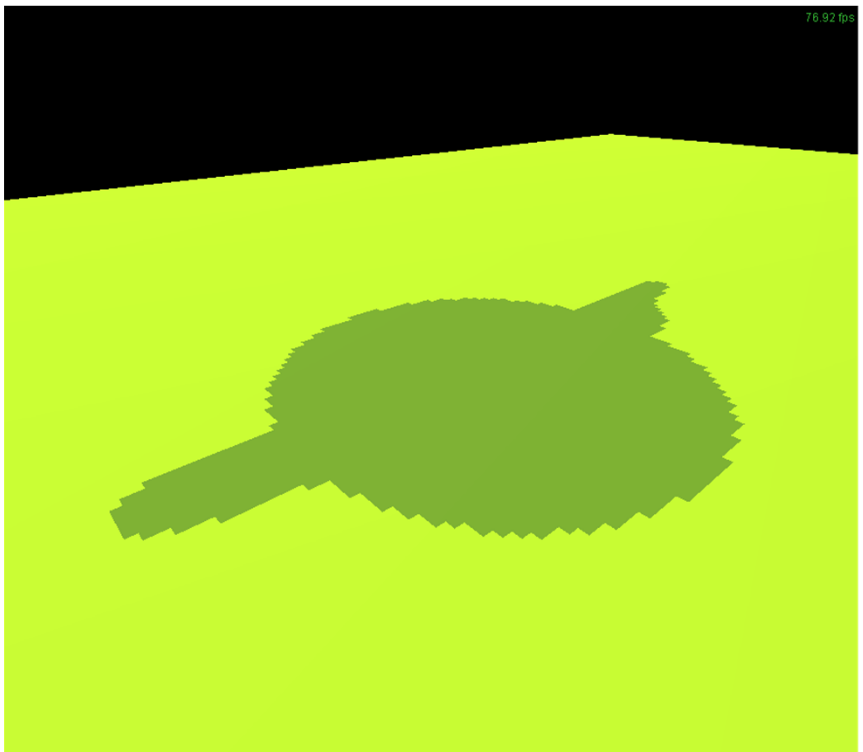
Too much



Just right

Sampling Problems

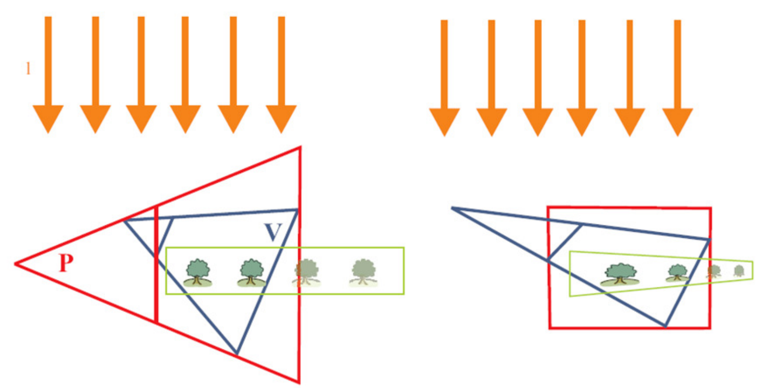
- Shadow map pixel may project to many image pixels
  → Stair-stepping artifacts



UCSD

Solutions

- Increase resolution of shadow map
  - Not always sufficient
- Split shadow map into several tiles
- Tweak projection for shadow map rendering
  - Light space perspective shadow maps (LiSPSM) http://www.cg.tuwien.ac.at/research/vr/lispsm/



- Combination of splitting and LiSPSM
  - Basis for most serious implementations