

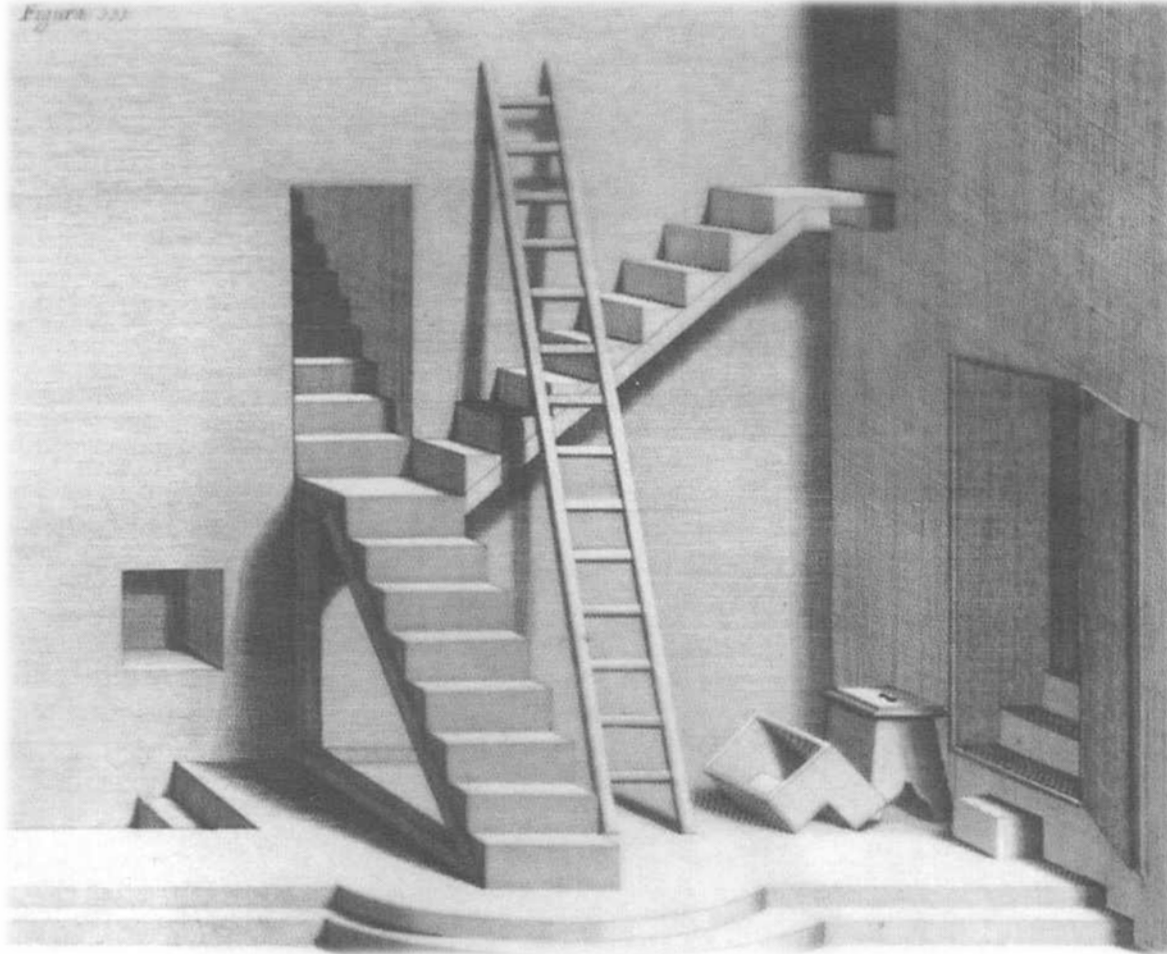
CSE 167:
Introduction to Computer Graphics
Lecture #16: Shadow Mapping

Jürgen P. Schulze, Ph.D.
University of California, San Diego
Fall Quarter 2019

Announcements

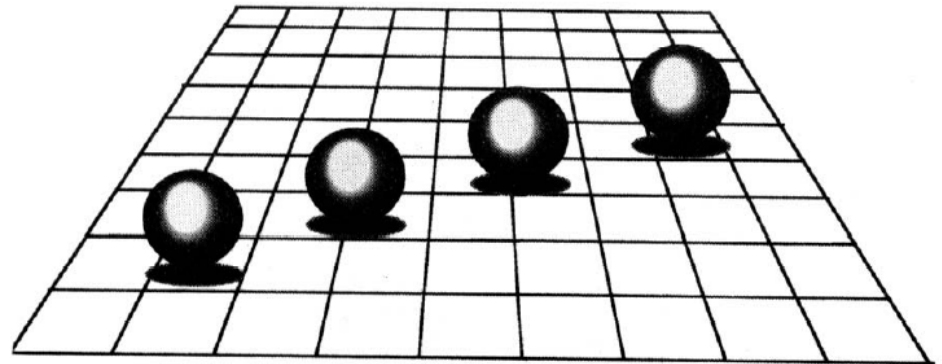
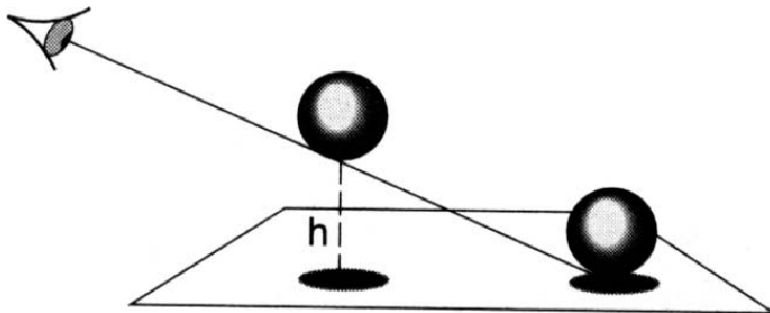
Why Are Shadows Important?

- ▶ Give additional cues on scene lighting

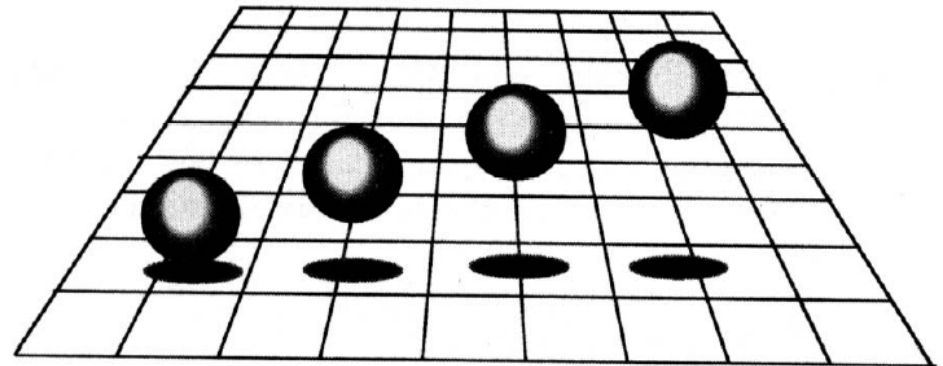


Why Are Shadows Important?

- ▶ Contact points
- ▶ Depth cues



A



Why Are Shadows Important?

► Realism



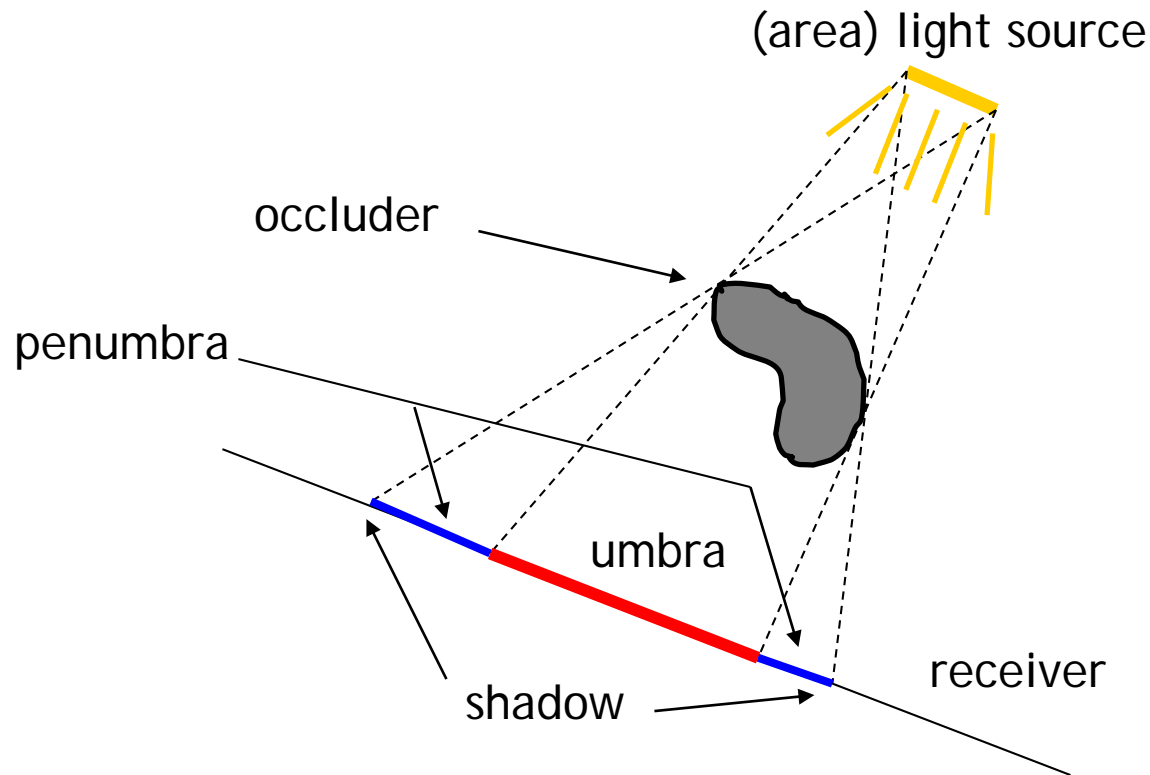
Without self-shadowing



With self-shadowing

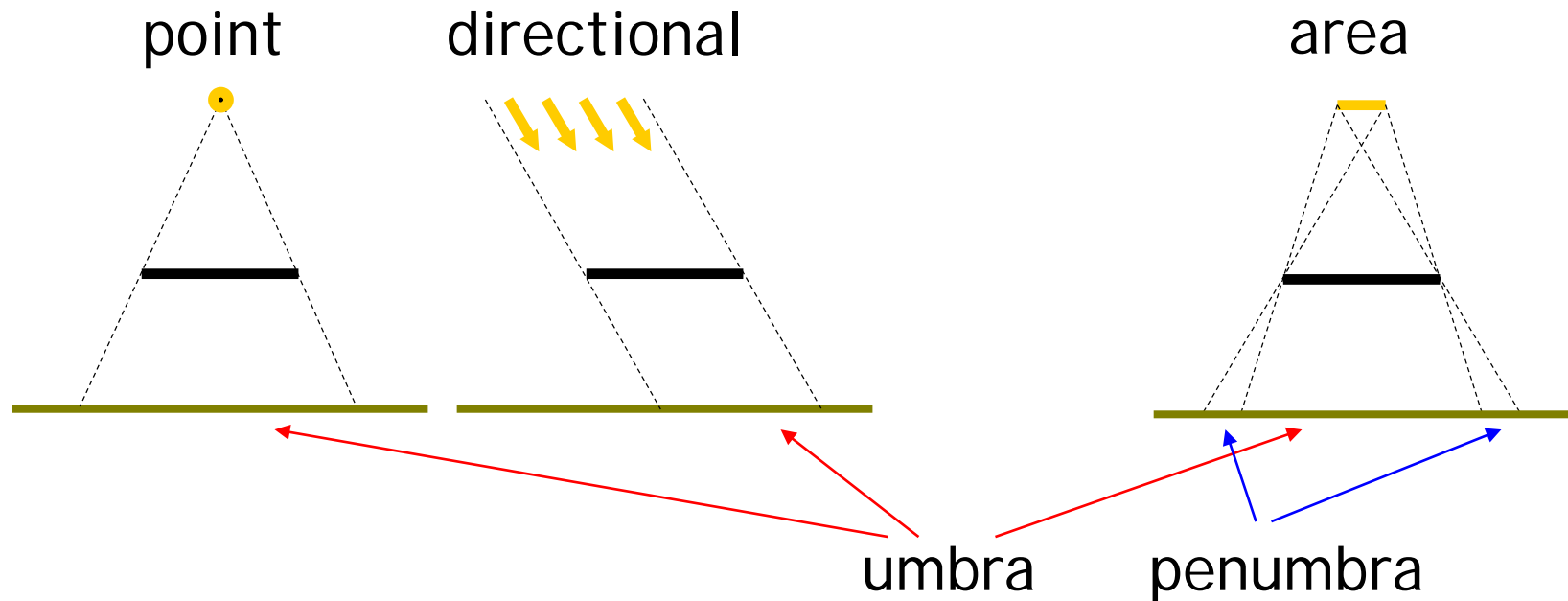
Terminology

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



Hard and Soft Shadows

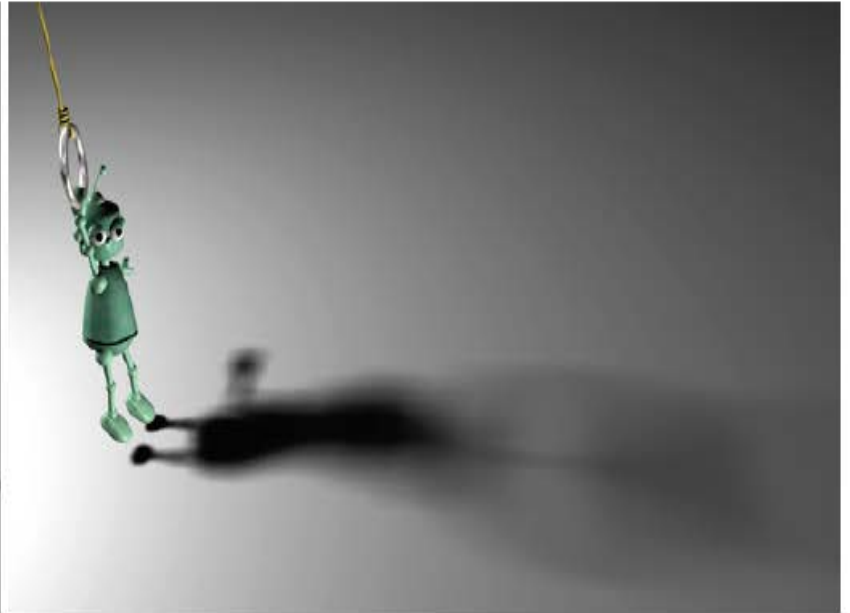
- ▶ Point and directional lights lead to hard shadows, no penumbra
- ▶ Area light sources lead to soft shadows, with penumbra



Hard and Soft Shadows



Hard shadow from
point light source



Soft shadow from
area light source

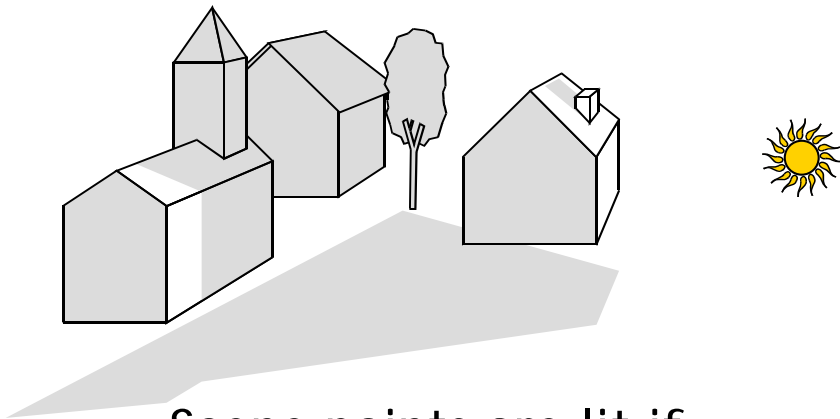
Shadows for Interactive Rendering

- ▶ 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

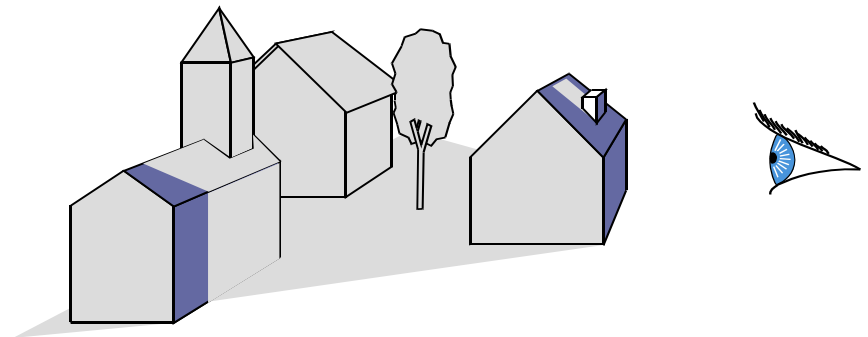
Shadow Mapping

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

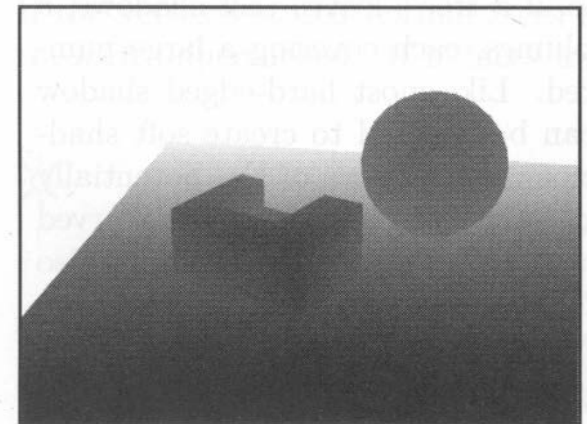
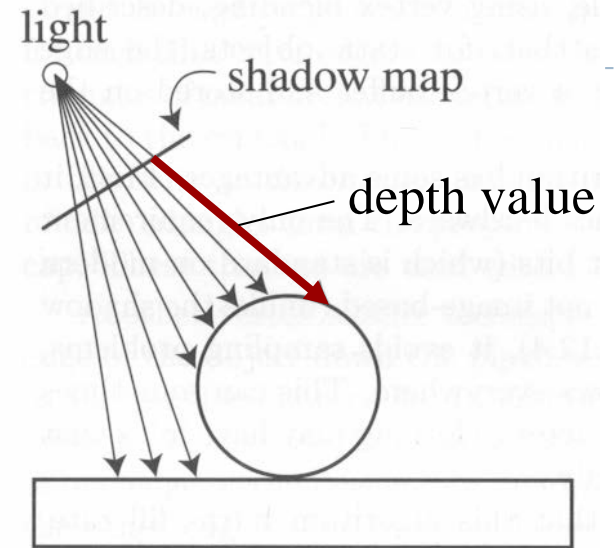


Determine visibility from light source by placing camera at light source position

Two Pass Algorithm

First Pass

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

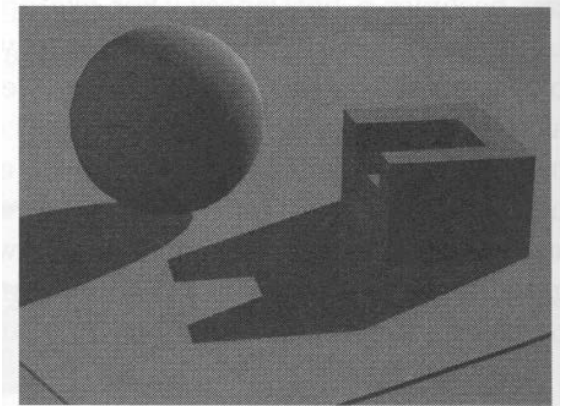
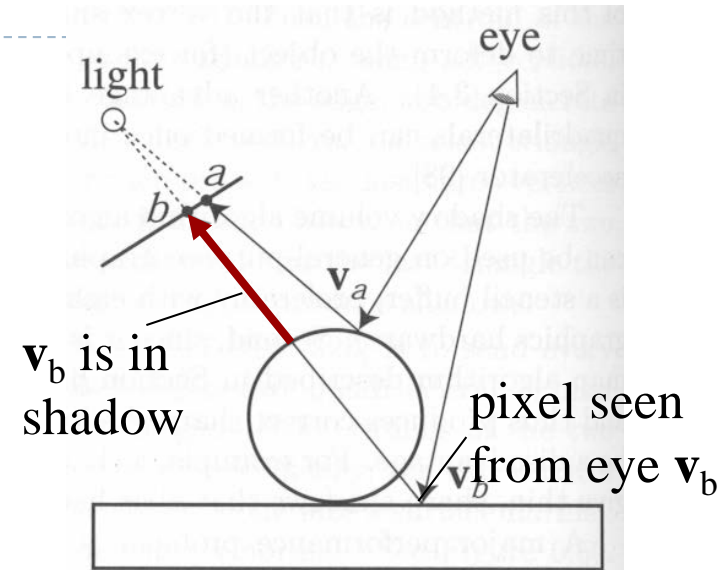


Depth image as seen
from light source

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



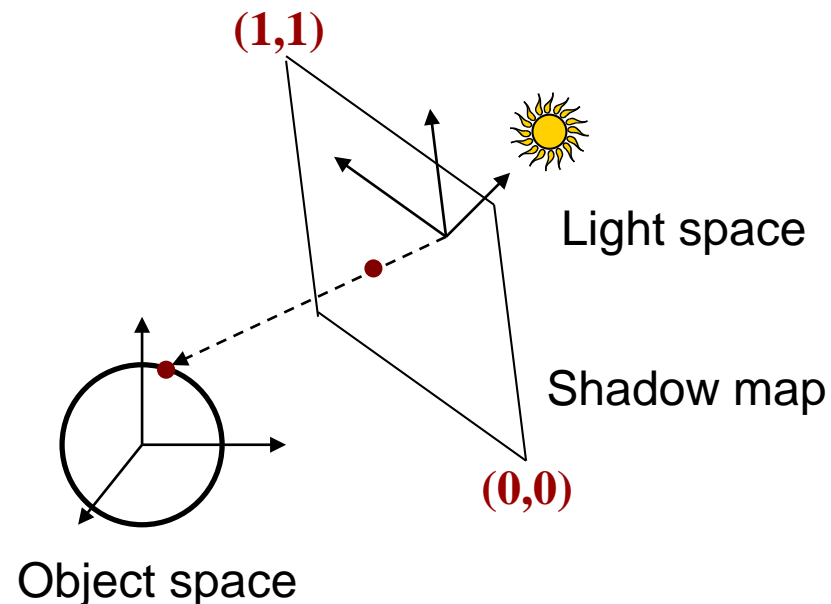
Final image with shadows

Shadow Map Look-Up

- ▶ Need to transform each point from object space to shadow map
- ▶ Shadow map texture coordinates are in $[0,1]^2$
- ▶ Transformation from object to shadow map coordinates

$$\mathbf{T} = \begin{bmatrix} 1/2 & 0 & 0 & 1/2 \\ 0 & 1/2 & 0 & 1/2 \\ 0 & 0 & 1/2 & 1/2 \\ 0 & 0 & 0 & 1 \end{bmatrix} \mathbf{P}_{light} \mathbf{V}_{light} \mathbf{M}$$

- ▶ \mathbf{T} is called texture matrix
- ▶ After perspective projection we have shadow map coordinates



Shadow Map Look-Up

- ▶ Transform each vertex to normalized frustum of light

$$\begin{bmatrix} s \\ t \\ r \\ q \end{bmatrix} = \mathbf{T} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

- ▶ Pass s,t,r,q as texture coordinates to rasterizer
- ▶ Rasterizer interpolates s,t,r,q to each pixel
- ▶ Use **projective texturing** to look up shadow map
 - ▶ This means, the texturing unit automatically computes $s/q, t/q, r/q, 1$
 - ▶ $s/q, t/q$ are shadow map coordinates in $[0,1]^2$
 - ▶ r/q is depth in light space
- ▶ Shadow depth test: compare shadow map at $(s/q, t/q)$ to r/q

GLSL Specifics

In application

- ▶ Store matrix **T** in OpenGL texture matrix
- ▶ Set using `glMatrixMode(GL_TEXTURE)`

In vertex shader

- ▶ Access texture matrix through predefined uniform `gl_TextureMatrix`

In fragment shader

- ▶ Declare shadow map as `sampler2DShadow`
- ▶ Look up shadow map using projective texturing with `vec4 texture2DProj(sampler2D, vec4)`

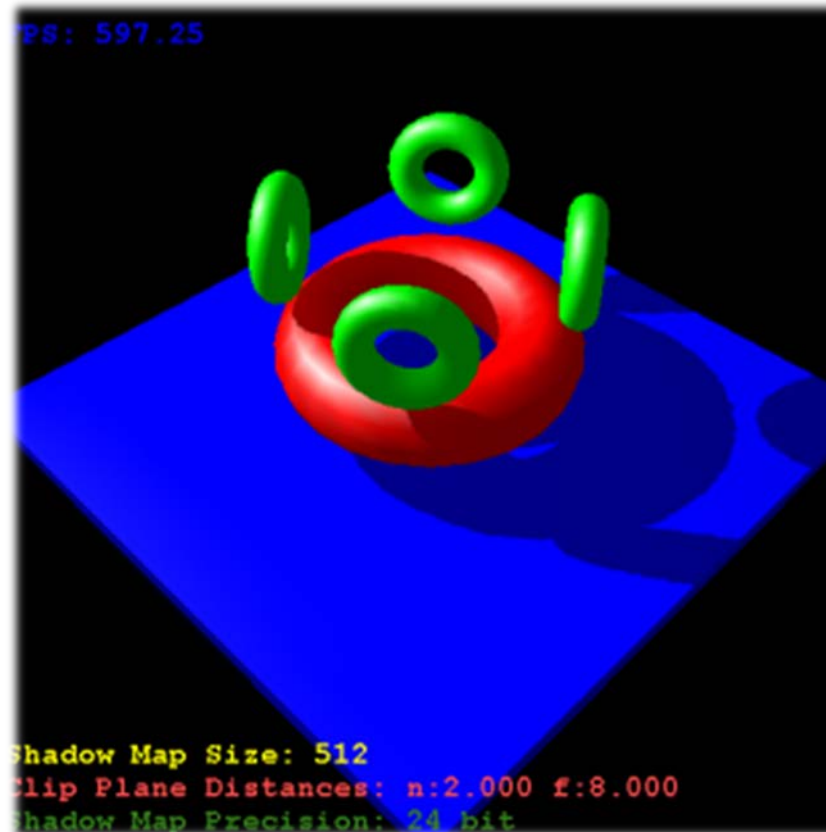
Implementation Specifics

- ▶ When you do a projective texture look up on a `sampler2DShadow`, the depth test is performed automatically
 - ▶ Return value is (1,1,1,1) if lit
 - ▶ Return value is (0,0,0,1) if shadowed
- ▶ Simply multiply result of shading with current light source with this value

Demo

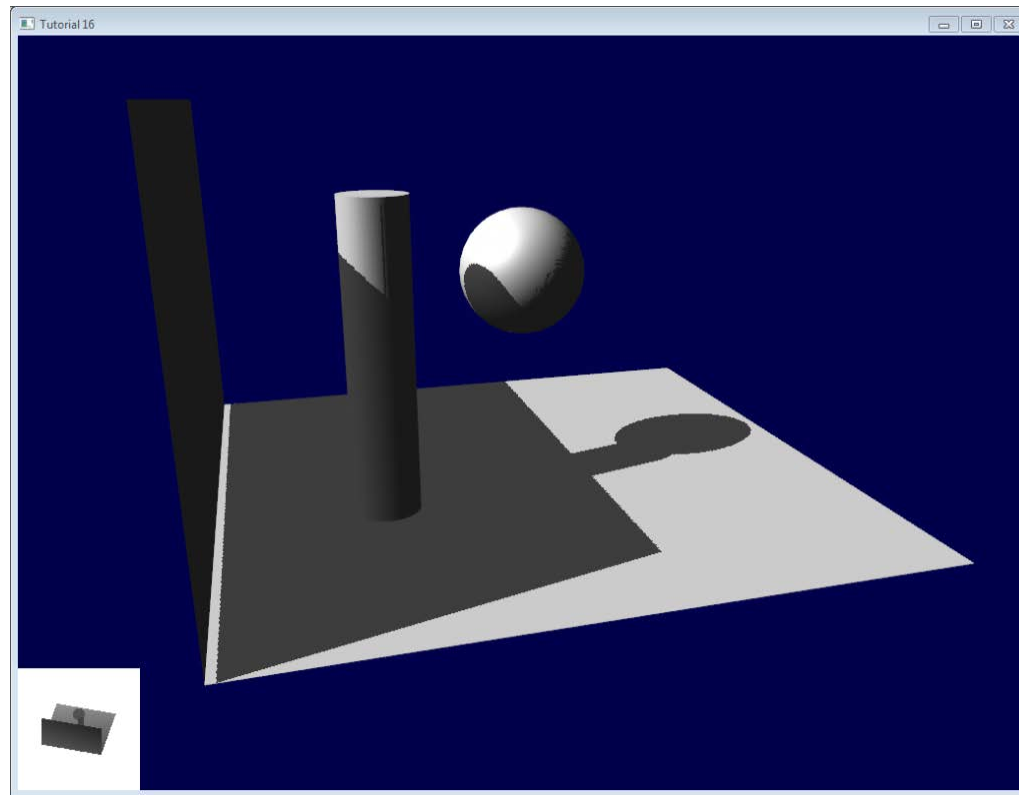
- ▶ Shadow mapping demo from

<http://www.paulsprojects.net/opengl/shadowmap/shadowmap.html>



Tutorial URL

- ▶ <http://www.opengl-tutorial.org/intermediate-tutorials/tutorial-16-shadow-mapping/>

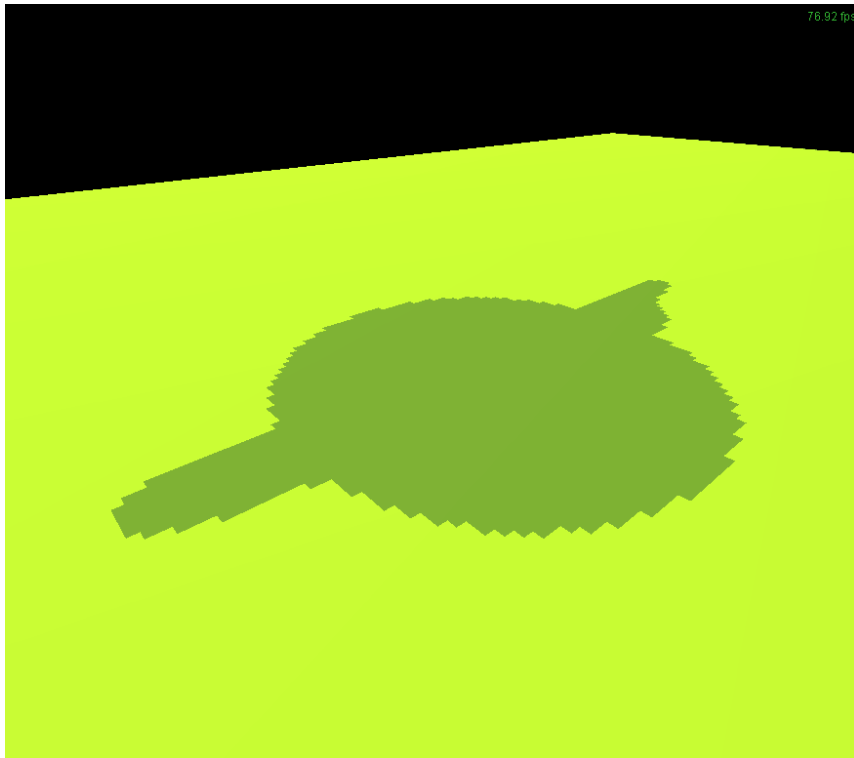


Issues With Shadow Maps

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

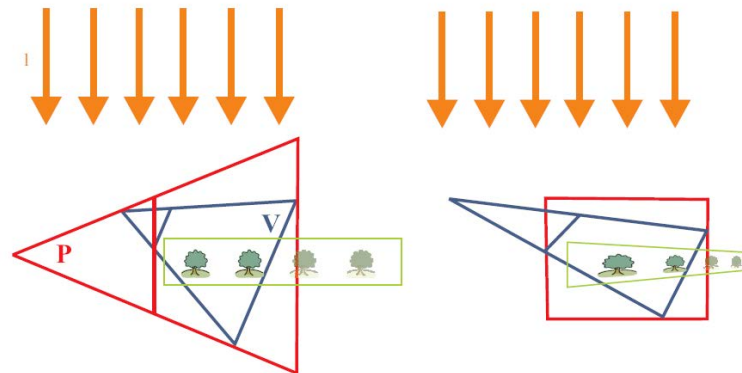
Sampling Problems

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



Solutions

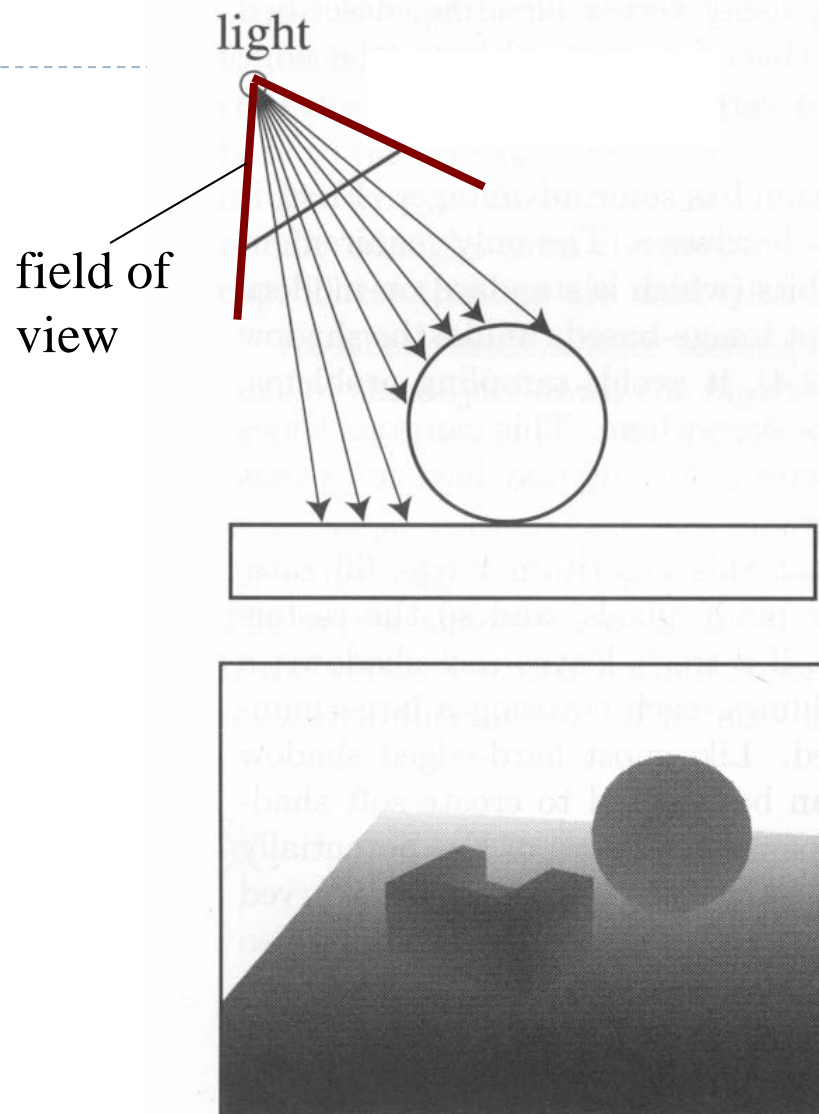
- ▶ Increase resolution of shadow map
 - ▶ Not always sufficient
- ▶ Split shadow map into several tiles
- ▶ Tweak projection for shadow map rendering
 - ▶ Light space perspective shadow maps (LiSPSM)
<http://www.cg.tuwien.ac.at/research/vr/lispsm/>



- ▶ Combination of splitting and LiSPSM
 - ▶ Basis for most commercial implementations

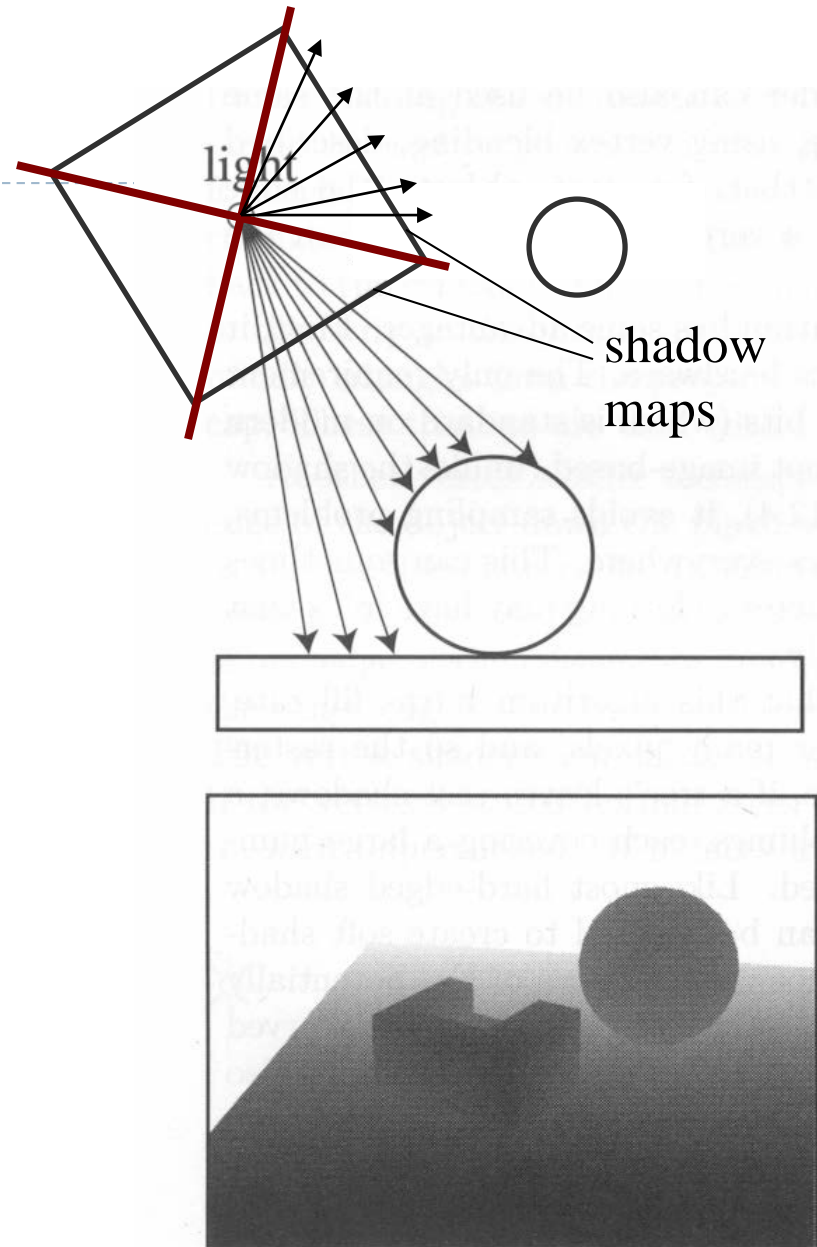
Limited Field of View

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



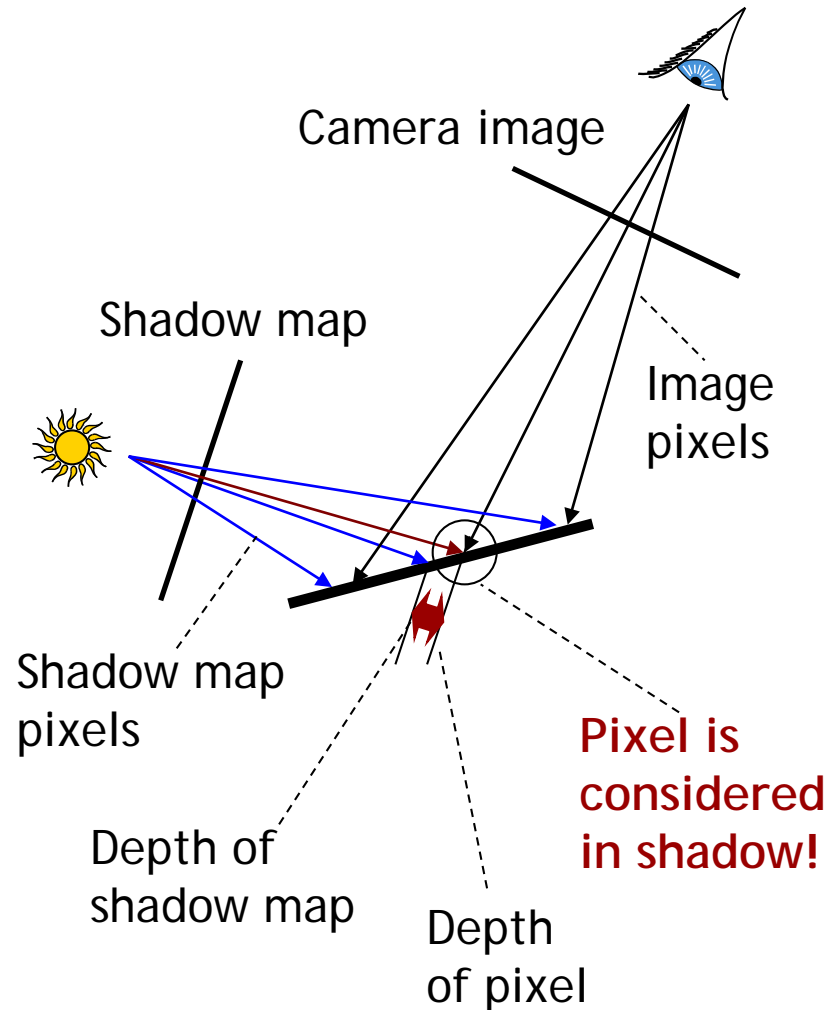
Limited Field of View

- ▶ 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



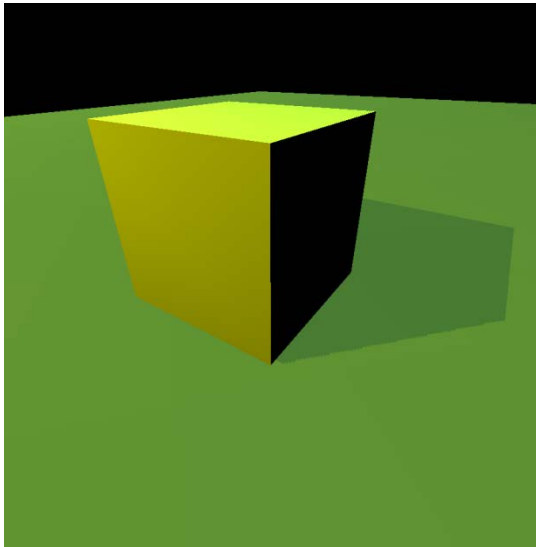
Z-Fighting

- ▶ 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

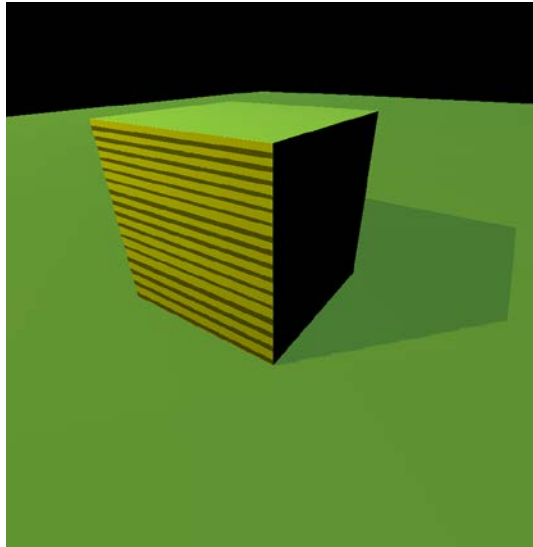


Solution: Bias

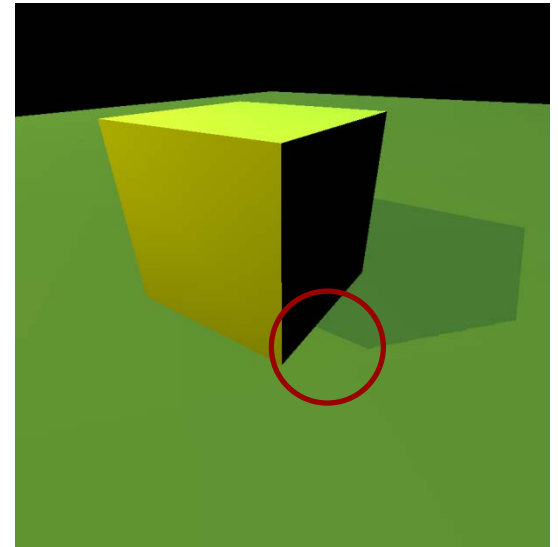
- ▶ Add **bias** when rendering shadow map
 - ▶ Move geometry away from light by small amount
- ▶ Finding correct amount of bias is tricky



Correct bias



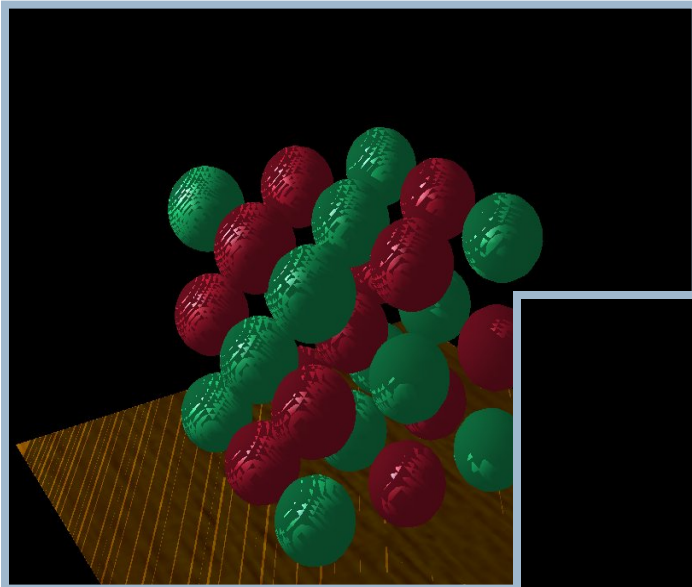
Not enough bias



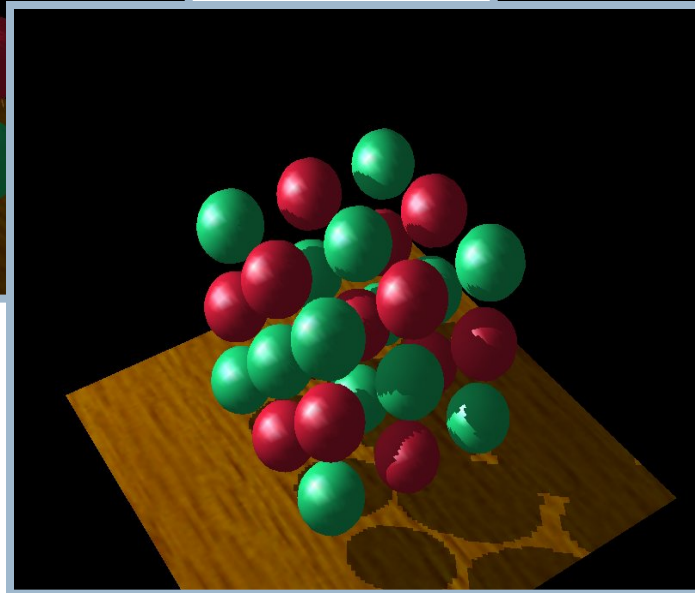
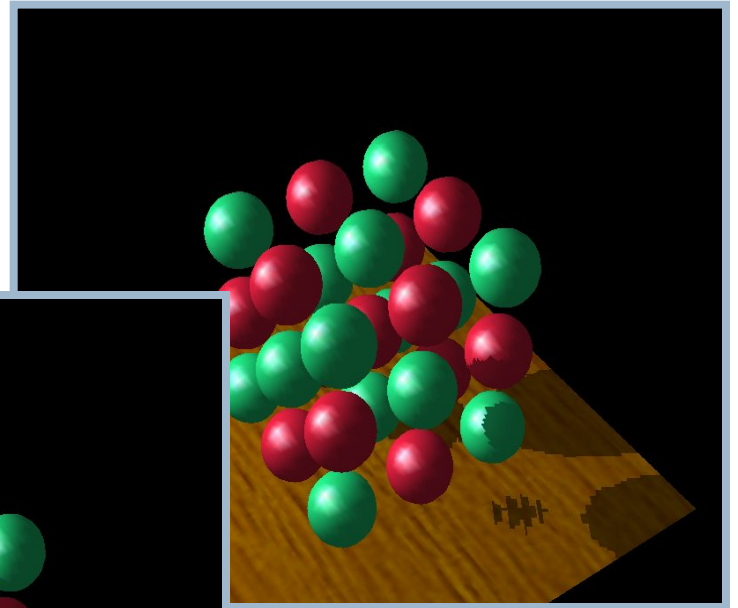
Too much bias

Bias Adjustment

Not enough



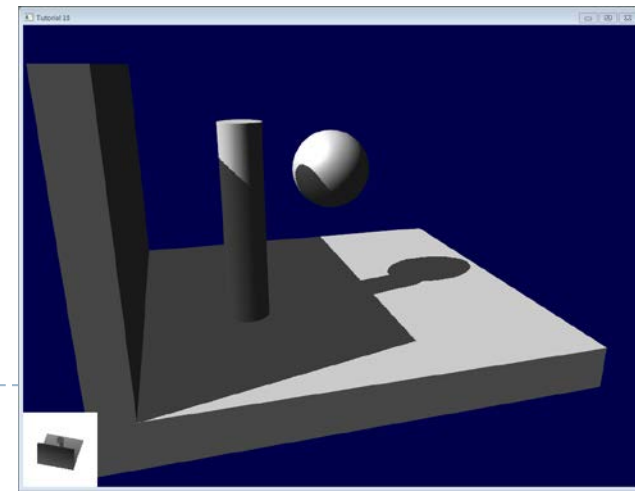
Too much



Just right

Shadow Mapping for Directional Lights

- ▶ Shadow mapping for directional light sources can be done with orthographic projection in step one when creating the shadow map.
- ▶ More information at:
 - ▶ <http://www.opengl-tutorial.org/intermediate-tutorials/tutorial-16-shadow-mapping/>
 - ▶ <http://www.scratchapixel.com/lessons/3d-basic-rendering/perspective-and-orthographic-projection-matrix/orthographic-projection-matrix>



Resources for Shadow Rendering

- ▶ Overview, lots of links

<http://www.realtimerendering.com/>

- ▶ Basic shadow maps

http://en.wikipedia.org/wiki/Shadow_mapping

- ▶ Faking soft shadows with shadow maps

<http://people.csail.mit.edu/ericchan/papers/smoothie/>