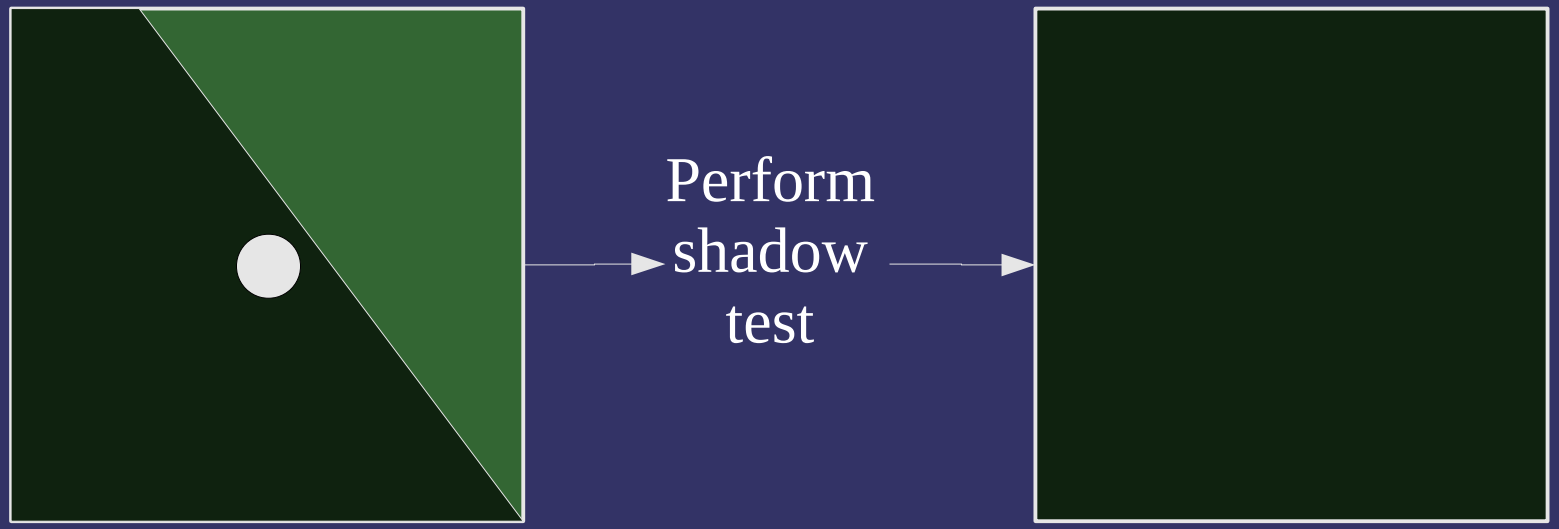
VGP353 – Week 3

- Agenda:
  - Quiz #1
  - More shadow maps:
    - Fixing some aliasing w/percentage closer filtering (PCF)
    - *Plausible* soft shadows w/percentage closer soft shadows (PCSS)

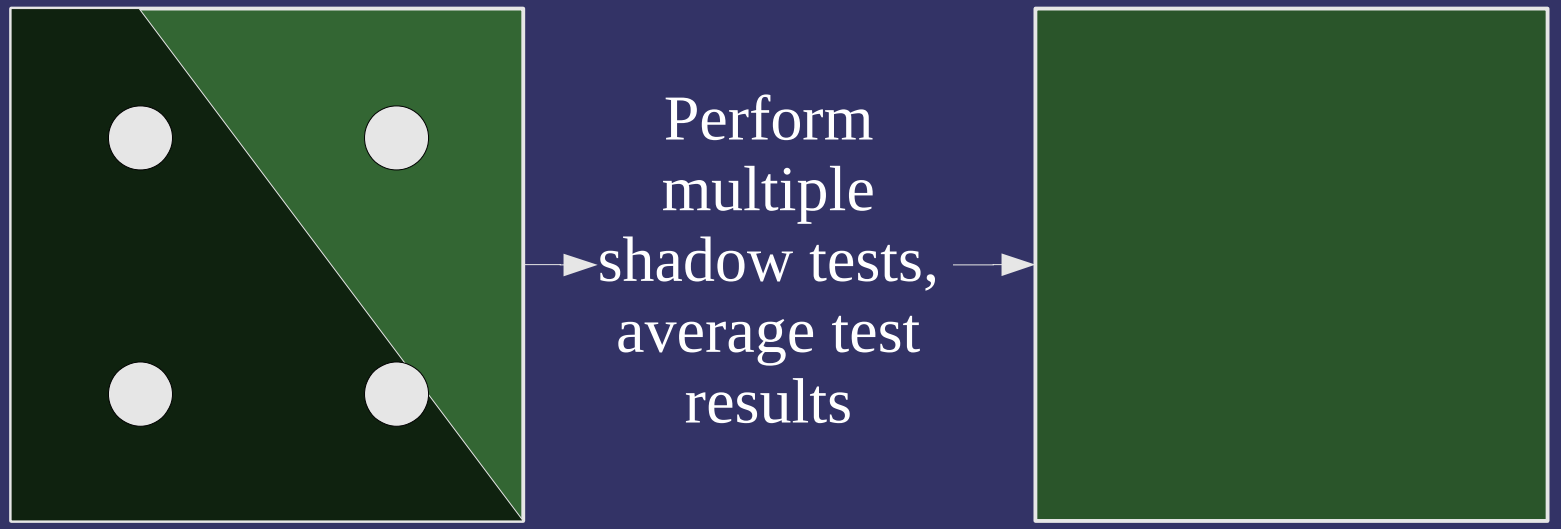
Shadow Map Texture Filtering

- Shadow test samples shadow map once per fragment
  - Results in two possible light levels: fully lit or fully shadowed



Shadow Map Texture Filtering

- Percentage closer filtering (PCF) reads multiple samples, performs one test per sample, averages test results
  - Results in n+1 possible light levels, where n is the number of samples



Shadow Map Texture Filtering

- Straight forward implementation in GLSL:

```
uniform vec2 bias;
uniform sampler2DShadow map;

void main()
{
    vec3 proj = coord.xyz / coord.w;
    vec3 p0 = proj - vec3((0.5 * bias.xy), 0.0);

    float s = texture(map, p0);
    s += texture(map, p0 + vec3(bias.x, 0.0, 0.0));
    s += texture(map, p0 + vec3(0.0, bias.y, 0.0));
    s += texture(map, p0 + vec3(bias.x, bias.y, 0.0));

    shadow /= 4.0;

    ...
}
```

Shadow Map Texture Filtering

- More concise in GLSL 1.30:

```
#version 130
uniform sampler2DShadow map;

void main()
{
    float s = textureProj(map, coord);
    s += textureProjOffset(map, coord, vec2(0, 1));
    s += textureProjOffset(map, coord, vec2(1, 0));
    s += textureProjOffset(map, coord, vec2(1, 1));

    shadow /= 4.0;

    ...
}
```

Shadow Map Texture Filtering

- Even better with GL_ARB_texture_gather
  - But you have to open-code the shadow comparison

```
#version 130
#extension GL_ARB_texture_gather: require

uniform sampler2D map;

void main()
{
    vec4 depth = textureGather(map, coord);
    vec4 s = vec4(lessThan(vec4(gl_FragDepth), depth);

    shadow = dot(s, vec4(0.25));

    ...
}
```

Percentage Closer Filtering

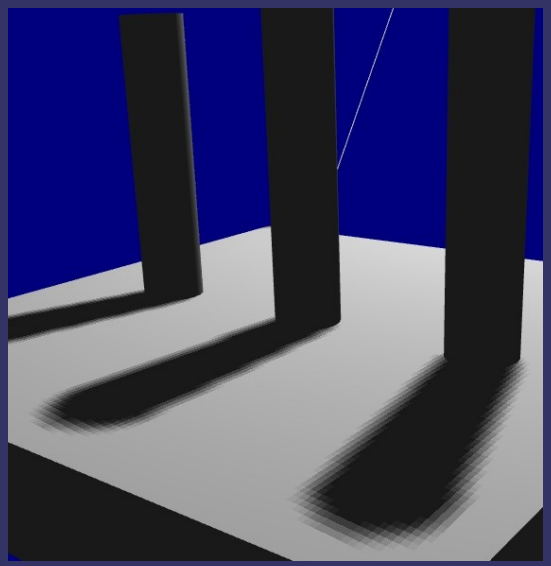
- The good news:
  - Improves quality
  - Larger filter kernels can be used to enable soft shadows
  - Some hardware can do 2x2 PCF nearly for free
    - Just enable `GL_LINEAR` filter on NVIDIA or Intel hardware

Percentage Closer Filtering

- The good news:
  - Improves quality
  - Larger filter kernels can be used to enable soft shadows
  - Some hardware can do 2x2 PCF nearly for free
    - Just enable `GL_LINEAR` filter on NVIDIA or Intel hardware
- The bad news:
  - Larger filter kernels are expensive
  - Grid-based sampling has artifacts

Grid-Based Sampling

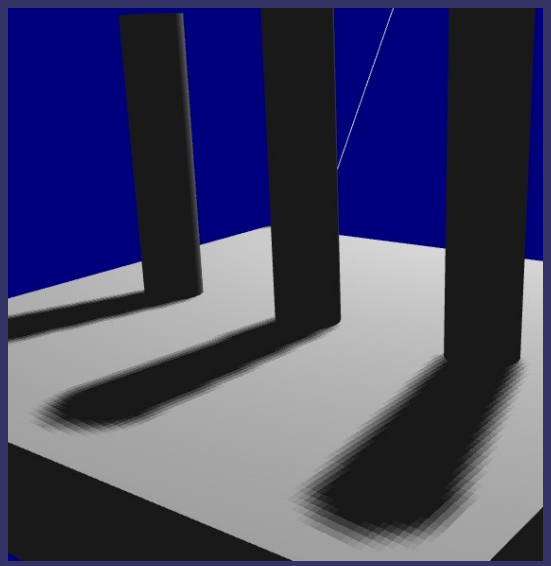
- Grid-based sampling artifacts have regular shape and are easily noticed by the eye

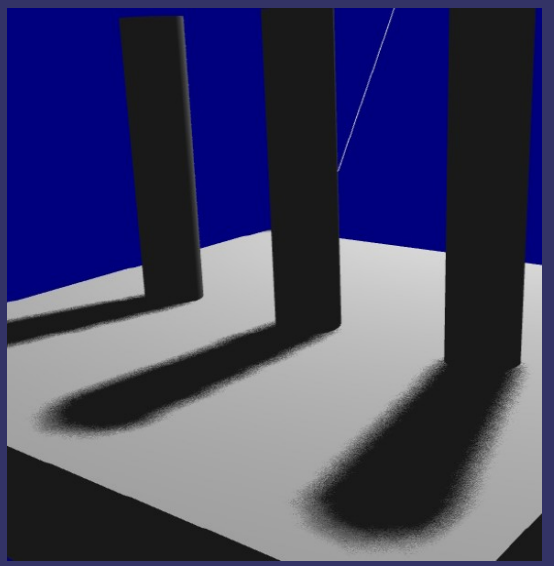


Images from http://ati.amd.com/developer/SIGGRAPH05/ShadingCourse_ATI.pdf

Grid-Based Sampling

- Grid-based sampling artifacts have regular shape and are easily noticed by the eye
- Irregular sample patterns are more easily accepted by the eye
  - Can even use fewer samples in the same size area

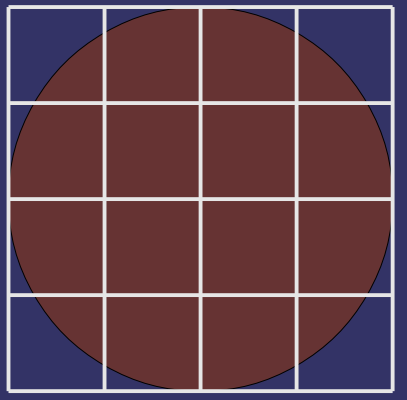




Images from http://ati.amd.com/developer/SIGGRAPH05/ShadingCourse_ATI.pdf

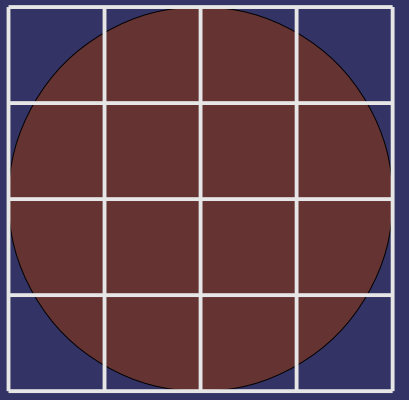
Irregular Sampling

- Select filter area

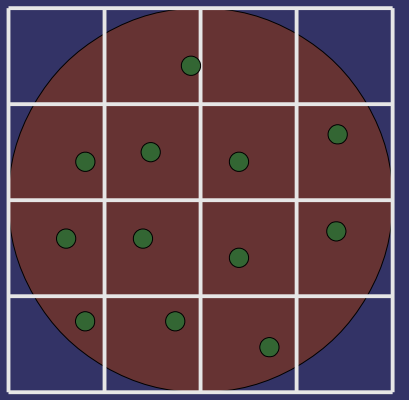


Irregular Sampling

- Select filter area

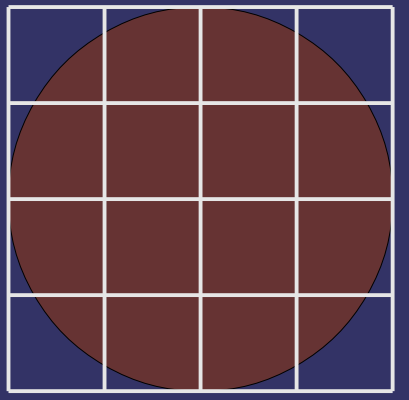


- Select random sample locations within area

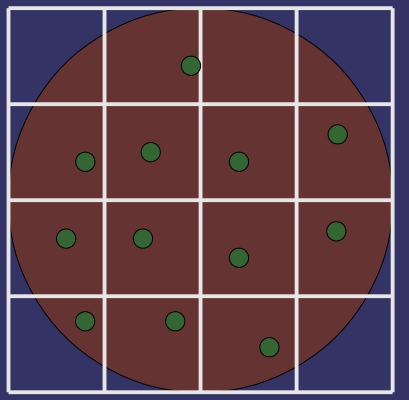


Irregular Sampling

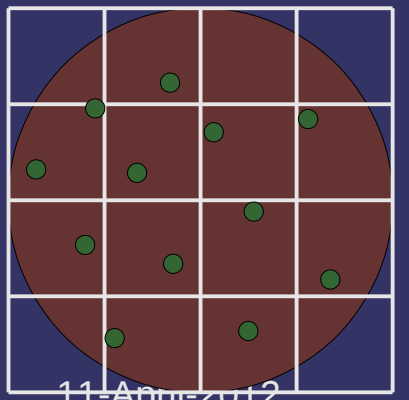
- Select filter area



- Select random sample locations within area



- Randomly rotate sample locations
  - Rotation based on screen location



Irregular Sampling

```
uniform vec2 sample_locations[12];
uniform mat2x2 sample_rotations[7];

float pcf_shadow_filter(sampler2DShadow map, vec4 coord)
{
    // Select rotation matrix based on pixel location
    int k = int(mod(gl_FragPosition.x + gl_FragPosition.y, 7.0));
    vec3 proj = coord.xyz / coord.w;
    float sum = 0.0;

    for (int i = 0; i < sample_locations.length(); i++) {
         vec2 offset = sample_rotations[k] * sample_locations[i]
         sum += texture(map, proj + vec3(offset, 0.0));
    }

    return sum / float(sample_locations.length());
}
```

Filter Cost

- 12 or 16 samples per fragment is expensive

Filter Cost

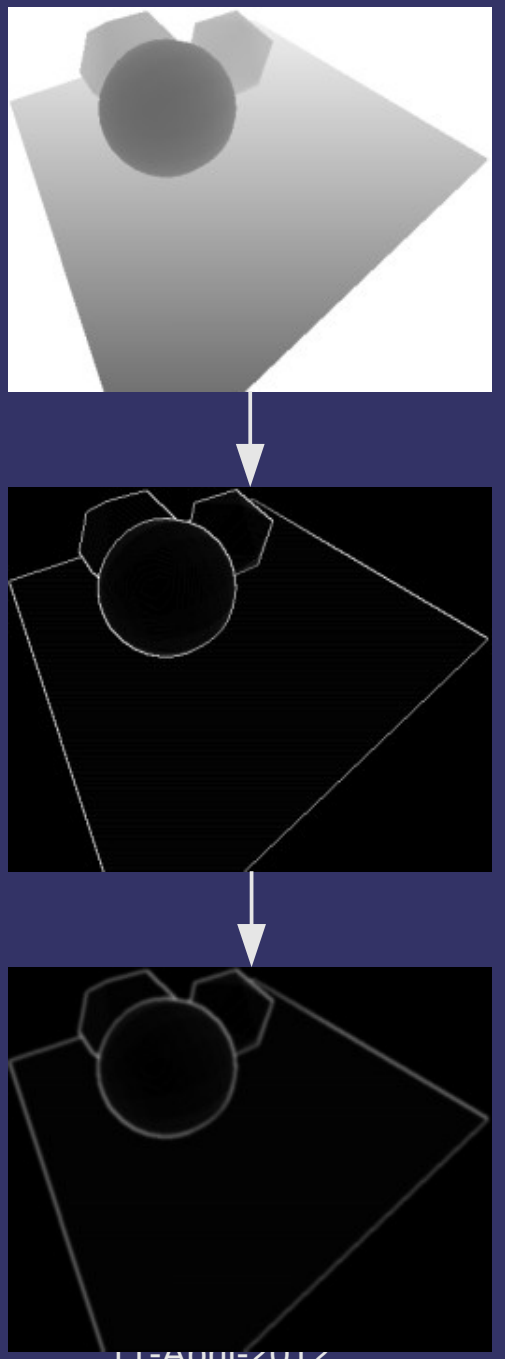
- 12 or 16 samples per fragment is expensive
  - In most of the final image, expensive sampling is unnecessary
  - Nyquist–Shannon sampling theorem tells us that areas with only low-frequency information need fewer samples than areas with high-frequency information

Filter Cost

- 12 or 16 samples per fragment is expensive
  - In most of the final image, expensive sampling is unnecessary
  - Nyquist–Shannon sampling theorem tells us that areas with only low-frequency information need fewer samples than areas with high-frequency information
    - The only high-frequency information is near the shadow boundaries!

Shadow Boundary Map

- Find boundaries in shadow map using edge detection filter
  - The edges in the map are the regions where the expensive filter should be applied
- Blur edge map using a blur kernel equal in size to the shadow map sample filter
  - This increases the area where the expensive filter will be applied and ensures that it will be applied everywhere that it needs to be



Shadow Boundary Map

- Use the shadow boundary map to determine whether to use one or many shadow samples

```
if (textureProj(boundary_map, proj) > 0.0)
    shadow = pcf_shadow_filter(shadow_map, proj);
else
    shadow = textureProj(shadow_map, proj)
```

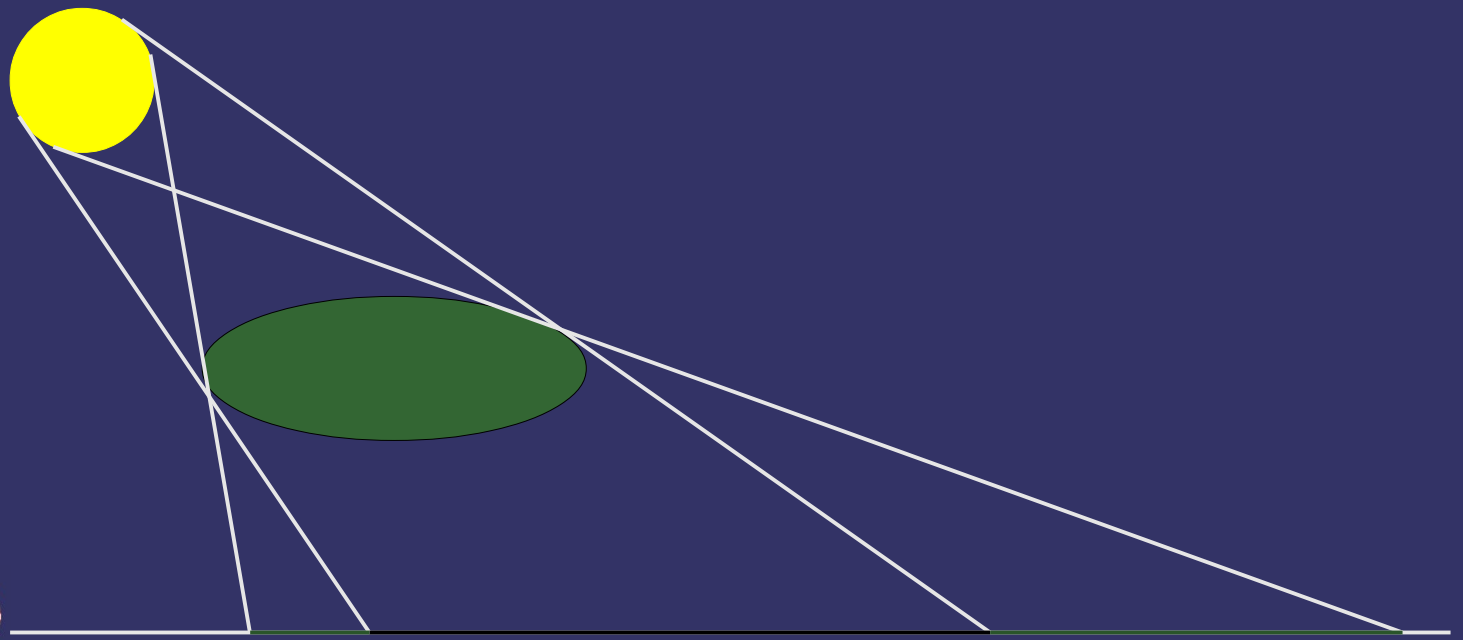
  - On hardware that support dynamic flow control, this can be a *big* win
    - DFC is a required part of DX 9.0c Shader Model 3.0
      - Geforce6 and later
      - Radeon X1xxx (R500) and later
      - Intel GMA X3000 (G965) and later

References

Sander, P. and Isidoro, J. *Explicit Early-Z Culling and Dynamic Flow Control on Graphics Hardware*. ATI Corporation, 2005, accessed 20 April 2008; available from http://ati.amd.com/developer/techpapers.html

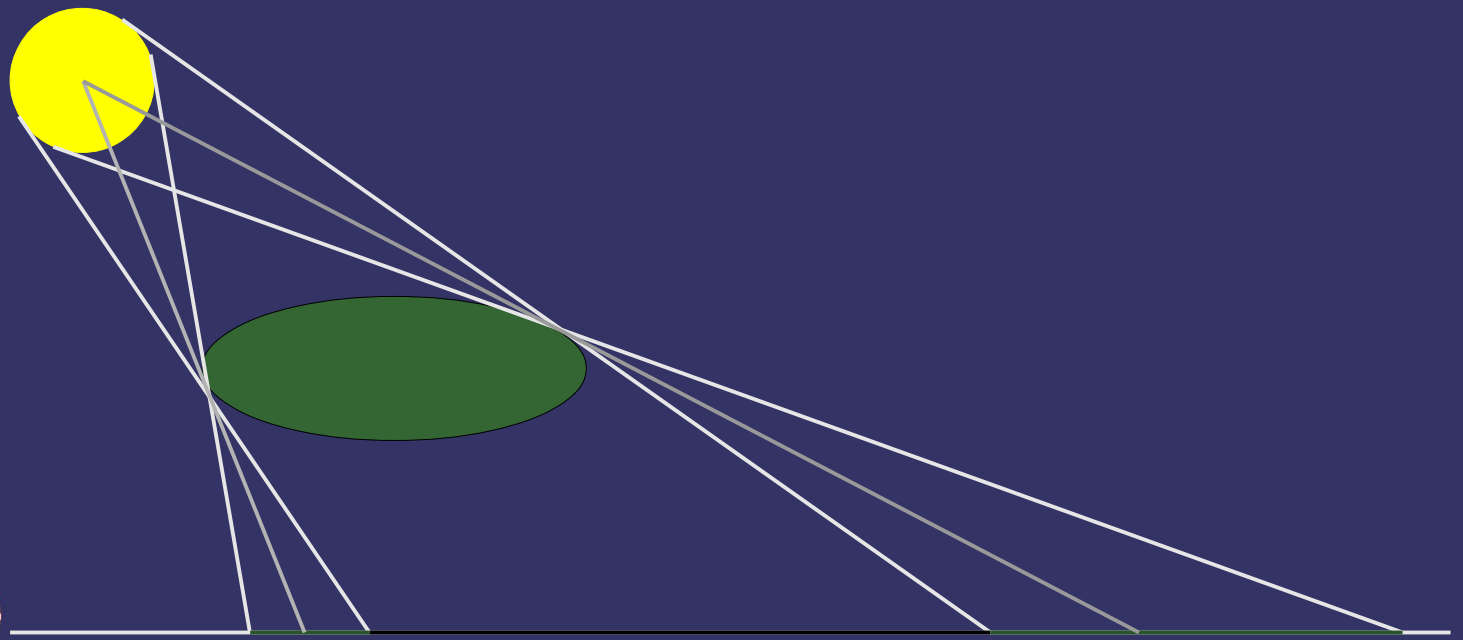
Soft Shadows

- Real lights have area
  - Since the light has area, there are regions where only a portion of the light is occluded...this is the penumbra



Soft Shadows

- Real lights have area
  - Since the light has area, there are regions where only a portion of the light is occluded...this is the penumbra
  - Shadow maps represent part of the penumbra as umbra and part as unoccluded



Soft Shadows

- Size of penumbra region varies with:
  - Size of light
  - Distance between occluder and light
  - Distance between occluder and receiver

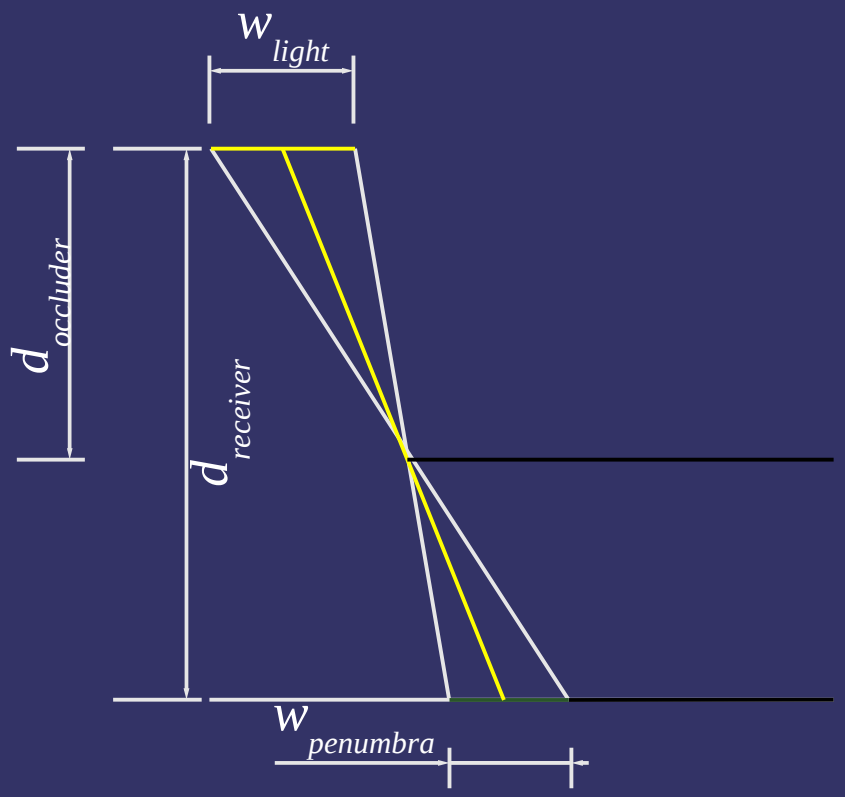
Soft Shadows

- Size of penumbra region varies with:
  - Size of light
  - Distance between occluder and light
  - Distance between occluder and receiver
- Using this information to perform *correct* light visibility calculations is hard
  - Make some simplifying assumptions!
  - Assume that all occluders, receivers, and lights are both flat and parallel to each other

Soft Shadows

- Estimate penumbra size using:

$$w_{penumbra} = \frac{(d_{receiver} - d_{occluder}) \times w_{light}}{d_{occluder}}$$

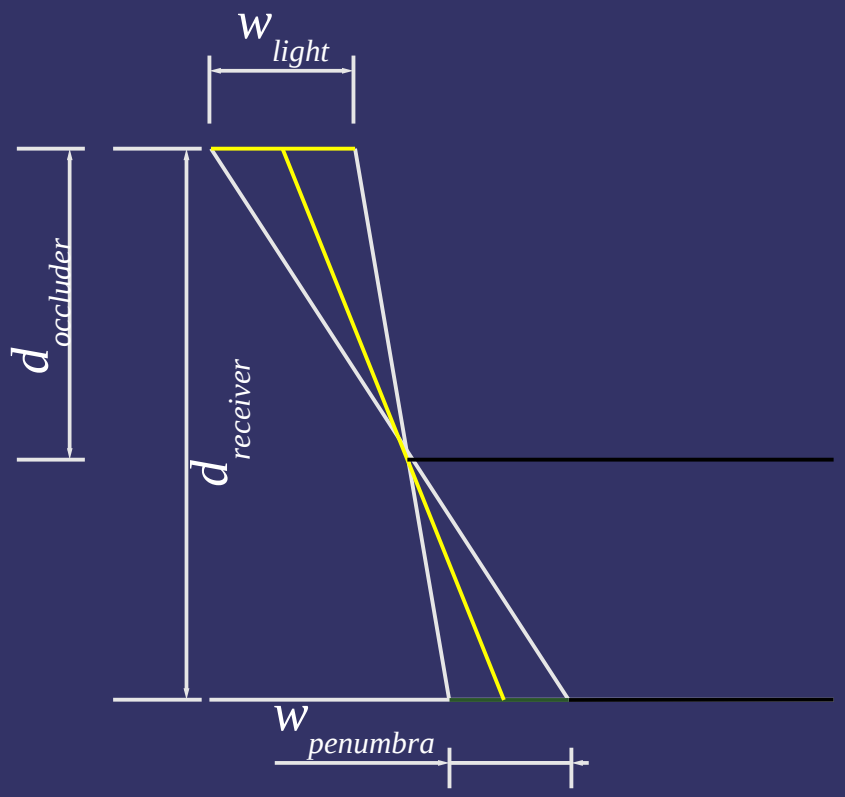


Soft Shadows

- Estimate penumbra size using:

$$w_{penumbra} = \frac{(d_{receiver} - d_{occluder}) \times w_{light}}{d_{occluder}}$$

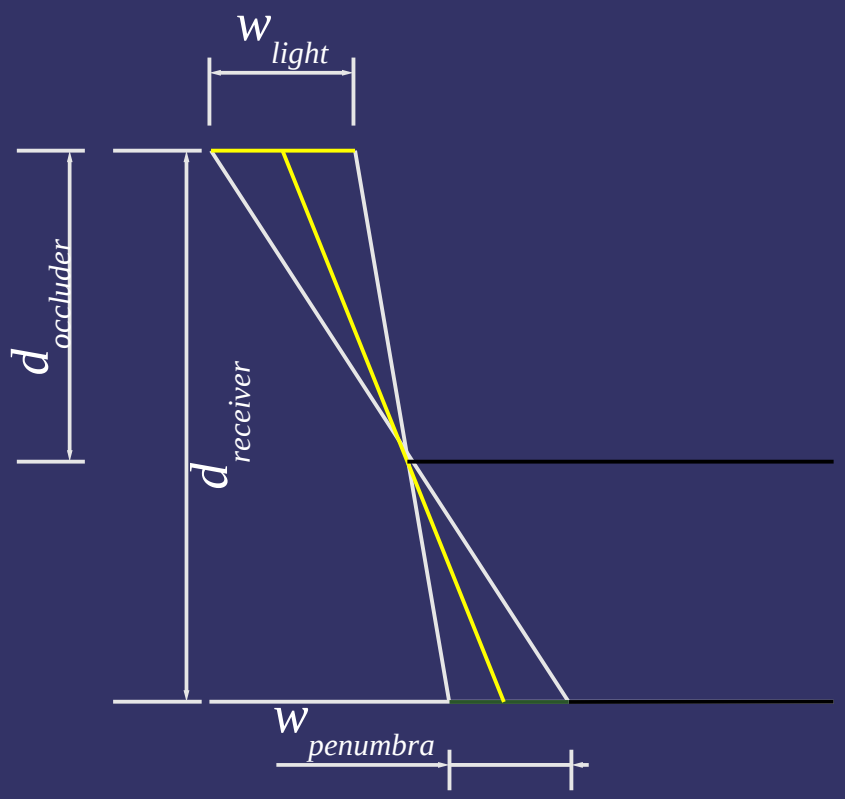
- How do we determine $d_{occluder}$?



Soft Shadows

- Estimate penumbra size using:

$$w_{penumbra} = \frac{(d_{receiver} - d_{occluder}) \times w_{light}}{d_{occluder}}$$



- How do we determine $d_{occluder}$?
  - Search the shadow map for possible occluders

Soft Shadows

- Examine a region around the point in the shadow map
  - Select region size based on light size and rendering budget
  - Sample values and *average* all depths less than the current fragment
    - Very similar to percentage closer filter (PCF)
- Use resulting average as $d_{occluder}$
  - $w_{penumbra}$ is the width of the PCF filter area

Soft Shadows

- Demo

Original image from
http://developer.nvidia.com/object/gdc_2005_presentations.html

References

Randima Fernando. *Percentage-Closer Soft Shadows.* 2005. Game Developer's Conference. http://developer.download.nvidia.com/shaderlibrary/docs/shadow_PCSS.pdf

Next week...

- Shadow volumes

Legal Statement

This work represents the view of the authors and does not necessarily represent the view of Intel or the Art Institute of Portland.

OpenGL is a trademark of Silicon Graphics, Inc. in the United States, other countries, or both.

Khronos and OpenGL ES are trademarks of the Khronos Group.

Other company, product, and service names may be trademarks or service marks of others.