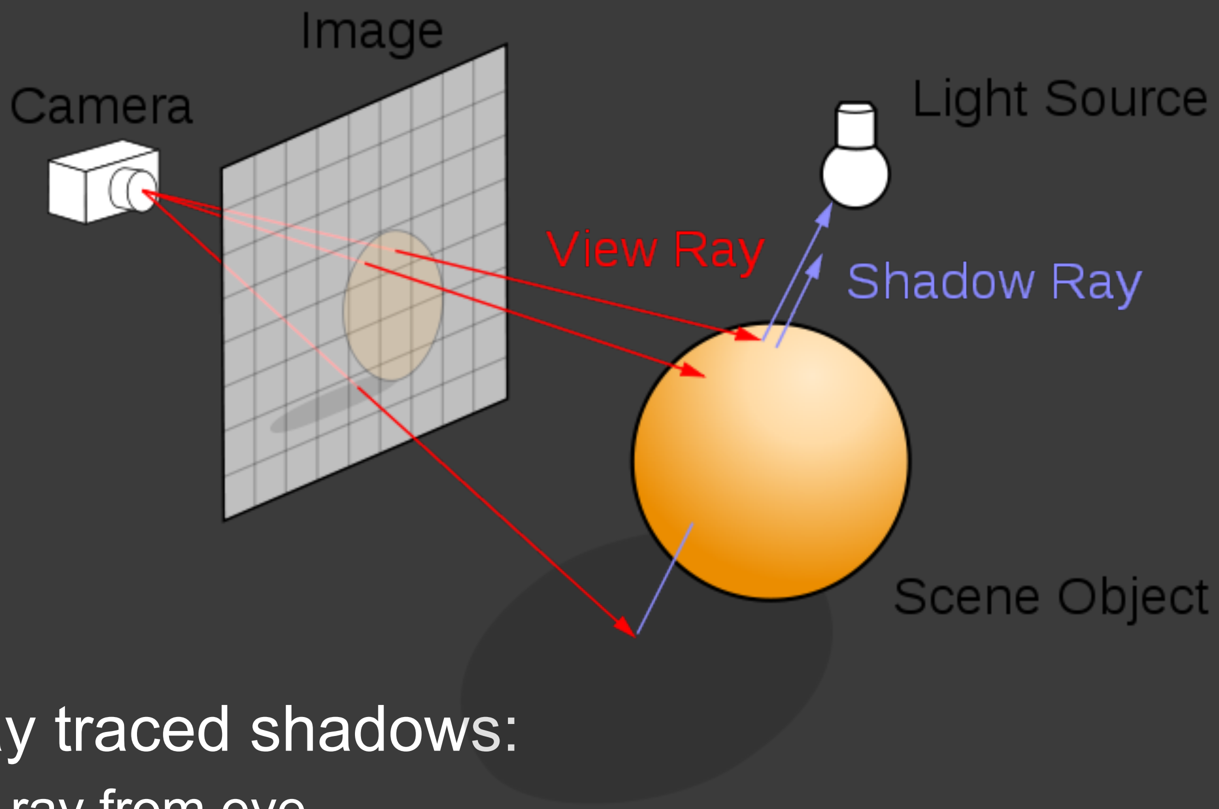
SHADOWS

For real-time rendering

Sean Lilley

Crysis 3 screenshot - http://www.gamingcentrum.com/wpcontent/uploads/2012/04/crysis42412-2.jpg

Hard shadows

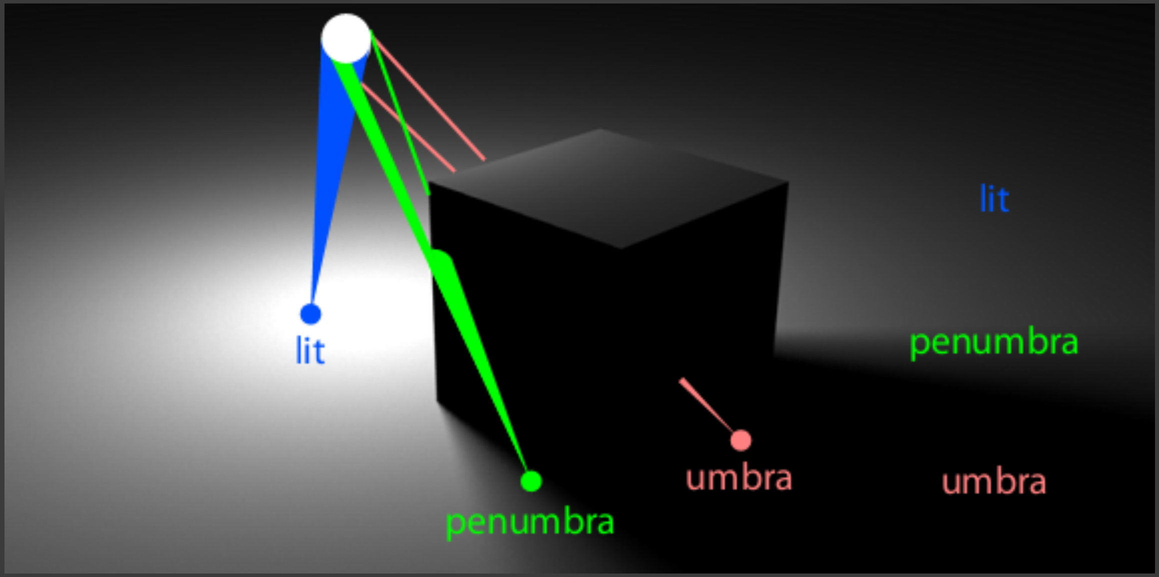


Basic ray traced shadows:

- Shoot ray from eye
- Ray hits a surface
- Send shadow ray out to check if the light reaches this point
- If the shadow ray is obstructed, then the point is in shadow

http://upload.wikimedia.org/wikipedia/commons/8/83/Ray_trace_diagram.svg

Soft shadows



- More realistic than hard shadows
- Light source treated as a physical object
- Shadow rays are now more like cones
- Penumbra: partially shadowed
- Umbra: completely shadowed

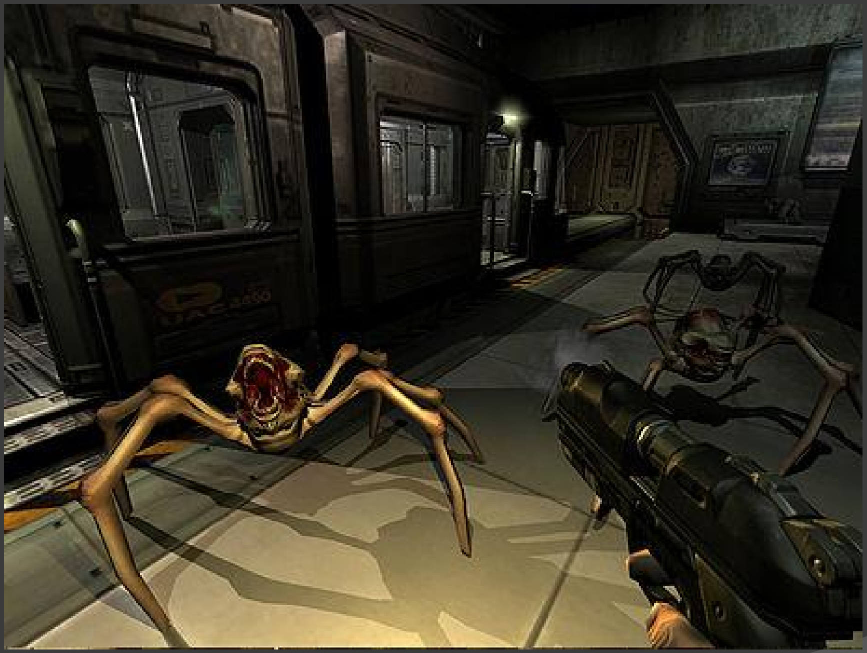
http://www.cse.chalmers.se/~uffe/SIGGRAPH2012CourseNotes.pdf

Common Shadow Techniques

- Shadow Volumes
  - Shadows are represented as polygonal volumes in space
  - Pros: accurate hard shadows
  - Cons: slow, rasterization heavy
- Shadow Maps
  - Shadows are determined through depth buffer comparison tests
  - Pros: fast, support for soft shadows
  - Cons: high memory usage, aliasing

Shadow Volumes

- Invented by Frank Crow in 1977
- Popularized by Doom 3 in 2005

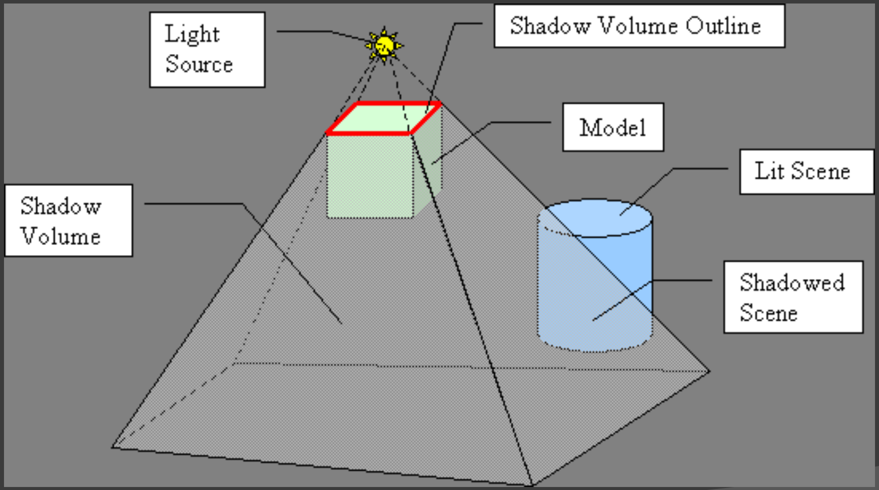


Doom 3 screenshot - http://en.wikipedia.org/wiki/File:Doom3shadows.jpg

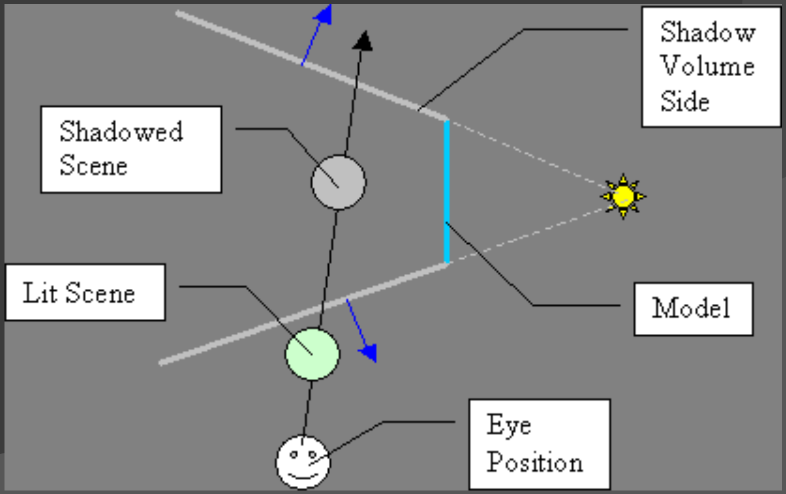
Shadow Volumes: Basic Approach

- For each triangle in the scene, project its edges to infinity along the direction of the light.
- This truncated pyramid is the shadow volume.
- Any part of the scene that lies inside the shadow volume is shadowed.

3D view



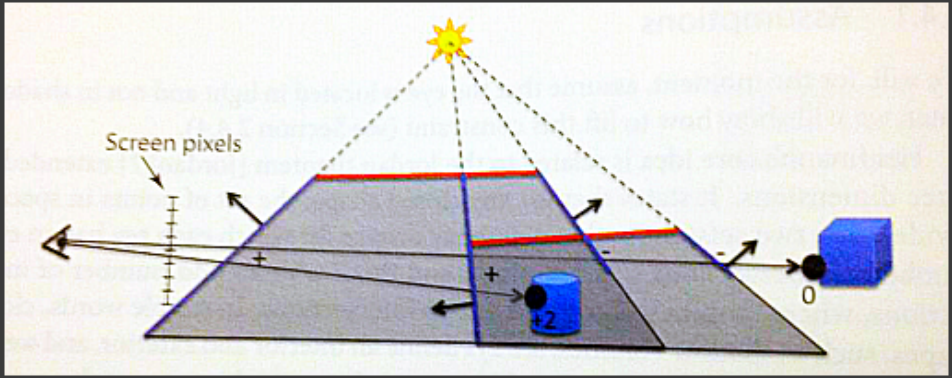
2D view



http://www.sonic.net/~surdules/articles/cg_shadowvolumes/index.html

Shadow Volumes: Implementation

- Depth pre-pass: render the scene into the depth buffer.
- Render all shadow volumes
  - If the shadow volume fragment passes the depth test:
    - If the triangle is front-facing, increment the stencil buffer.
    - If the triangle is back-facing, decrement the stencil buffer.
  - If the value of the stencil buffer at a particular pixel is 0, then that pixel is not in shadow.



- Now render the scene again, enabling the color buffer.
  - Use the stencil buffer as a mask. If the stencil buffer is greater than 0 for a particular fragment, discard the fragment
  - Else, compute diffuse and specular lighting like usual

Image: Real-Time Shadows by Eisemann et al.

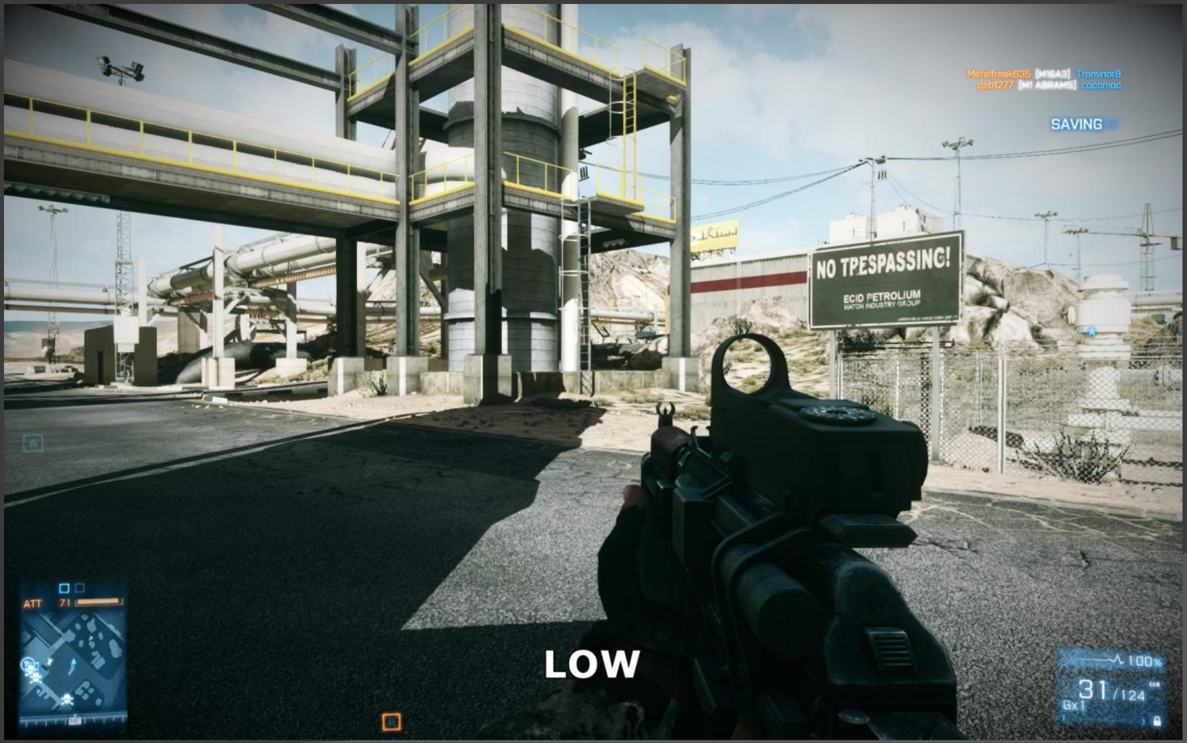
Shadow Volumes: Improvements

- What if camera originates in a shadow volume? Won't the stencil values be wrong?
  - Use z-fail (Carmack's Reverse): If a shadow volume *fails* the depth test, *increment* stencil value if back-facing and *decrement* stencil value if front-facing.
- Use the geometry shader to create shadow volumes easily on the GPU.
  - Emit triangle primitives for the shadow volume front cap, sides, and back cap.
- Create shadow volumes from simplified meshes
- Use occlusion culling to reduce the number of shadow volumes that need to be created.

http://http.developer.nvidia.com/GPUGems3/gpugems3_ch11.html

Shadow Maps

- Invented by Lance Williams in 1978
- Very popular. The dominant technique in today's video games.

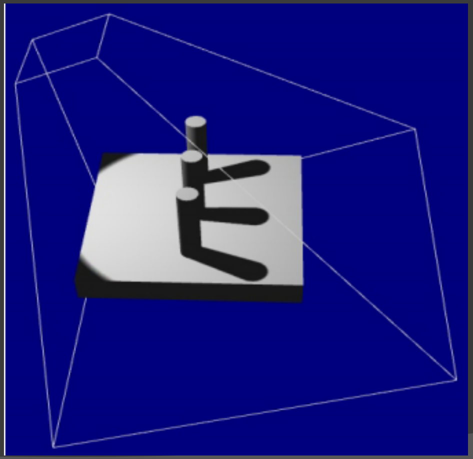


Battlefield 3 screenshot: http://www.geforce.com/Active/en_US/shared/images/guides/bf3-tweak/41_Shadows_Low.jpg

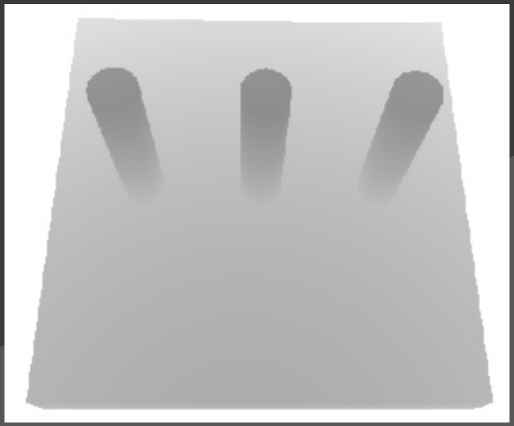
Basic Approach

- Render the scene from the light's point of view.
  - Treat the light like a camera.
  - Render to a depth texture to create the shadow map.
- Render the scene from the camera's point of view.
  - Transform each vertex from world space to light space in vertex shader.
  - Send light space position to fragment shader.
  - Compare the depth of the fragment to the depth stored in the shadow map. If the depth is greater, it is shadowed.

Light frustum



Shadow Map



http://developer.amd.com/media/gpu_assets/Isidoro-ShadowMapping.pdf

Light Types

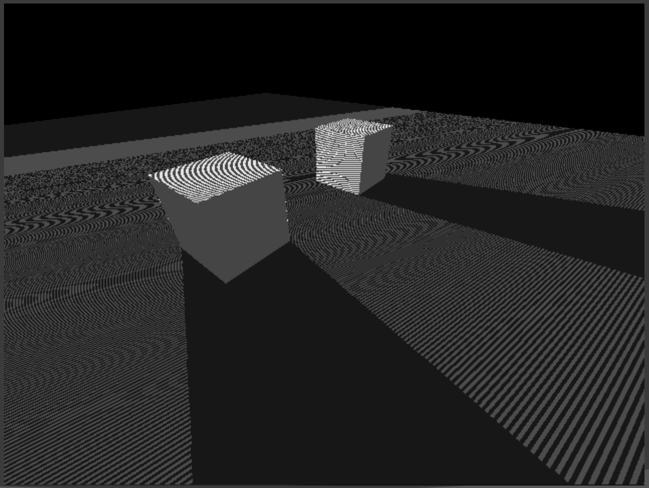
- Directional light (sun) – use orthographic projection
- Spot light – use perspective projection
- Point light – like spot light, but requires an omnidirectional shadow map.
  - Create six light frustums and render to a cube map inside the geometry shader.
- Shadow mapping is expensive; usually only one light source casts shadows in a standard video game.

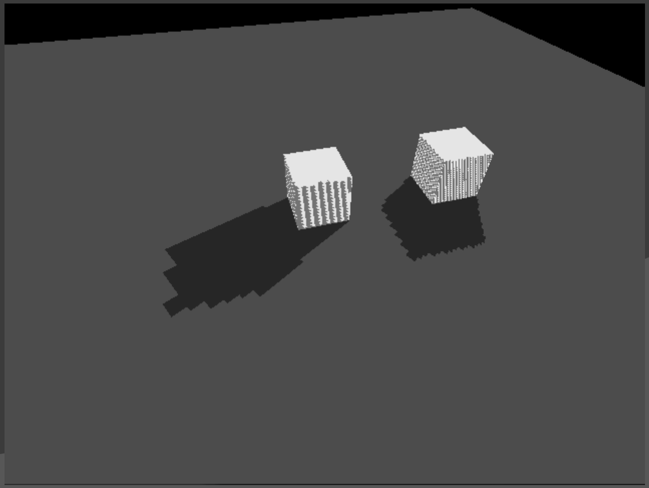
Basic Shadow Mapping Problems

- Projective Aliasing
- Perspective Aliasing
- Texture Resolution Limits

Projective Aliasing

- Occurs when the slope of the geometry is parallel to the light direction
  - Best case: overhead light, flat floor
  - Worst case: overhead light, straight walls
- Depth buffer precision, shadow map resolution, and float comparisons cause problems even for best case.
  - Called z-fighting

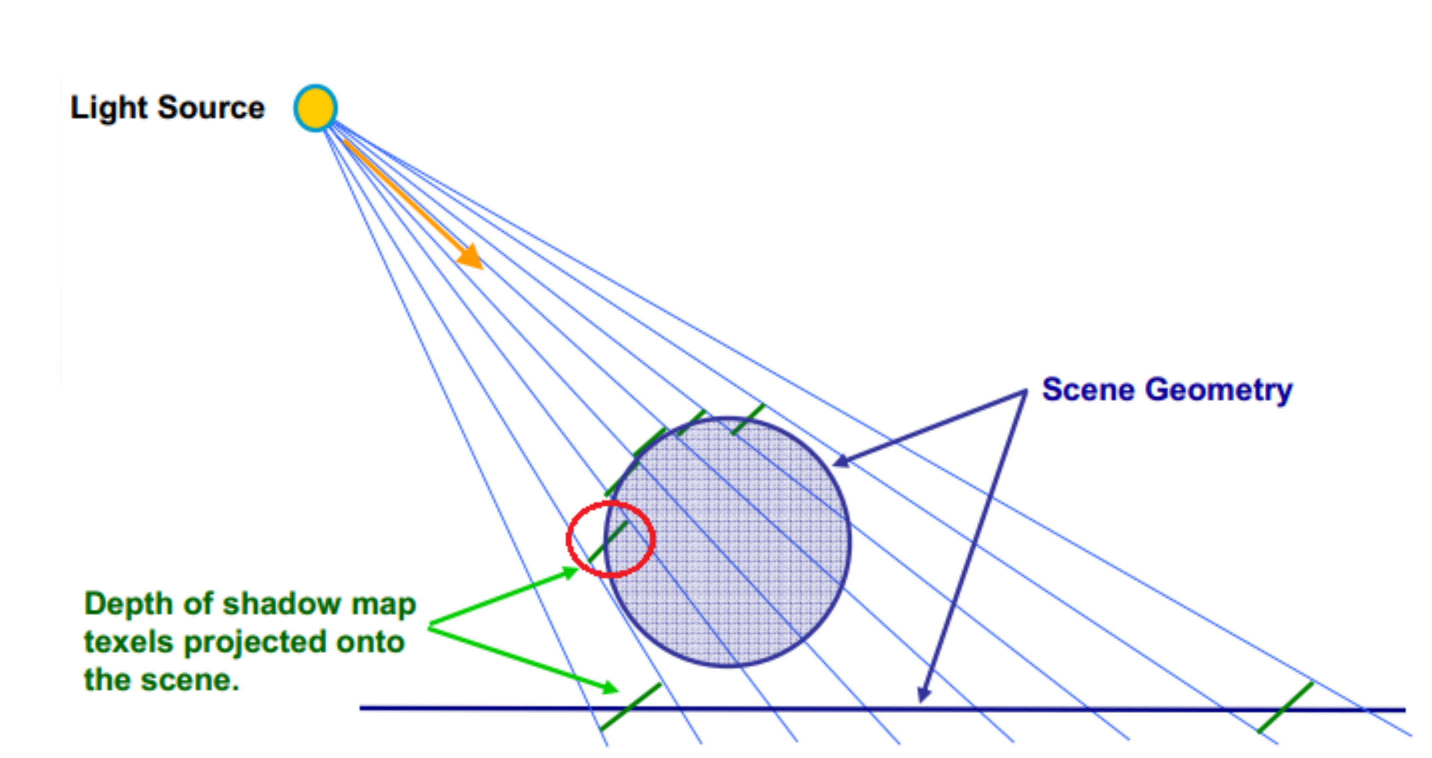




http://fabiensanglard.net/shadowmapping/index.php

Projective Aliasing

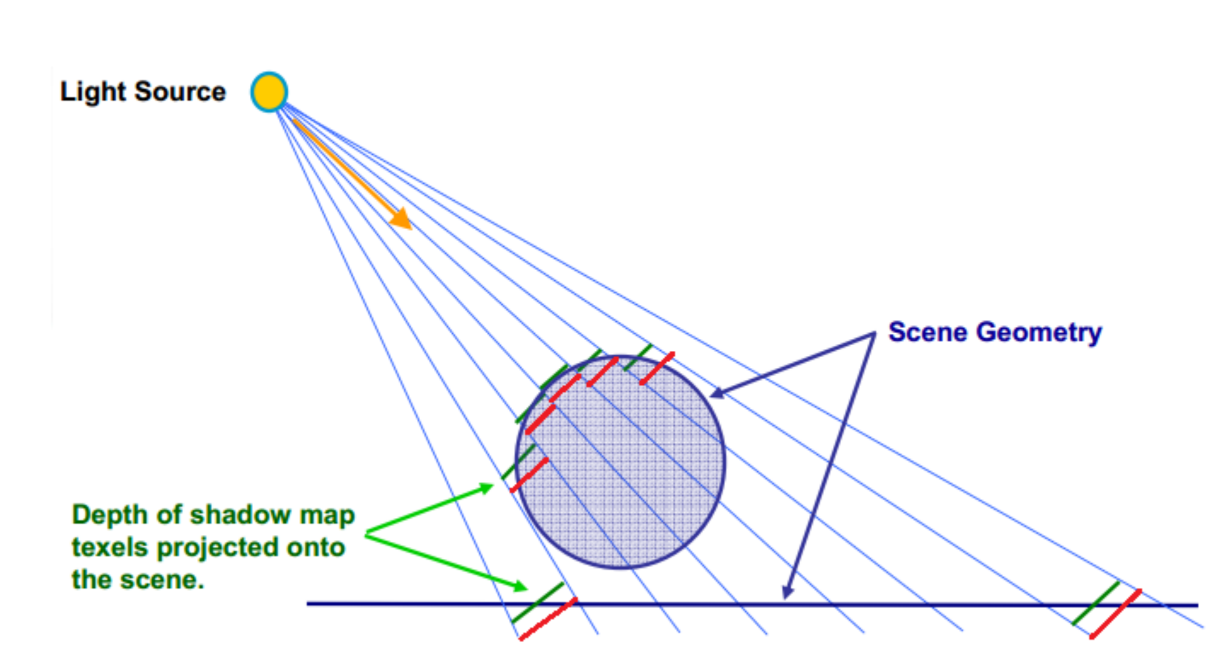
- Seriously affects the side of the circle



Modified from: http://developer.amd.com/media/gpu_assets/Isidoro-ShadowMapping.pdf

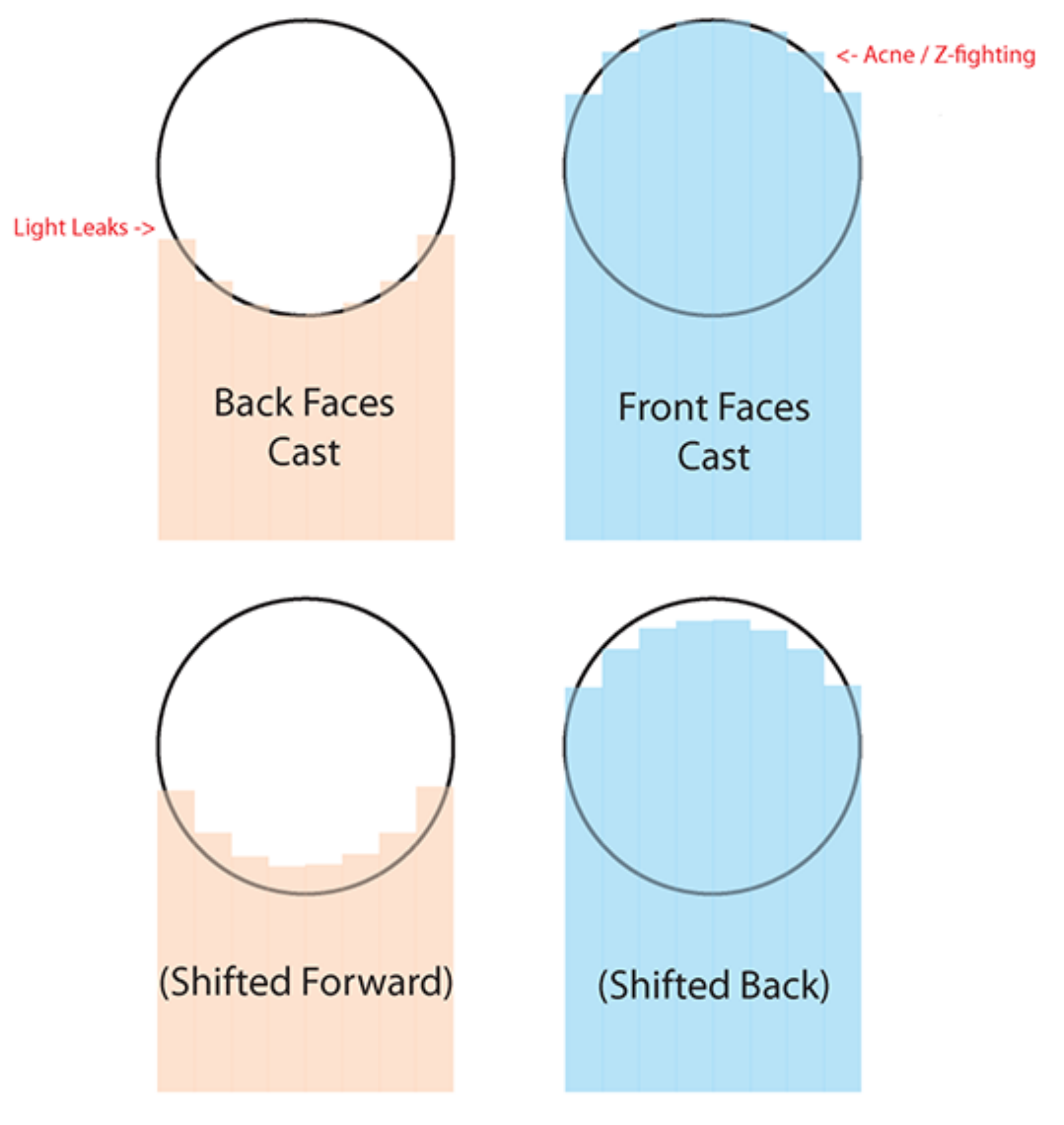
Depth Bias

- Apply a constant depth bias
  - During light pass, push depths slightly deeper
  - Now depth comparison test will succeed in best case
  - Still problematic in worst case



Modified from: http://developer.amd.com/media/gpu_assets/Isidoro-ShadowMapping.pdf

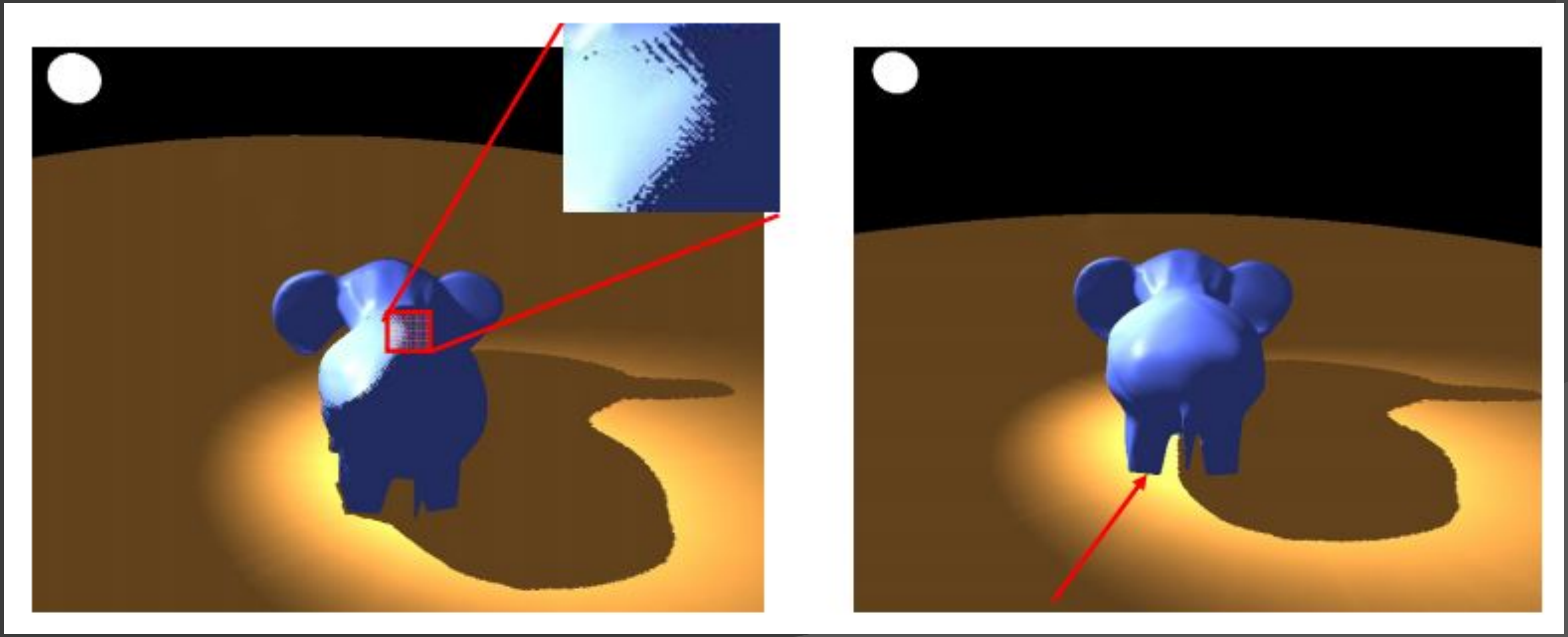
Depth Bias



http://i.imgur.com/8kEZ1En.png

Depth Bias

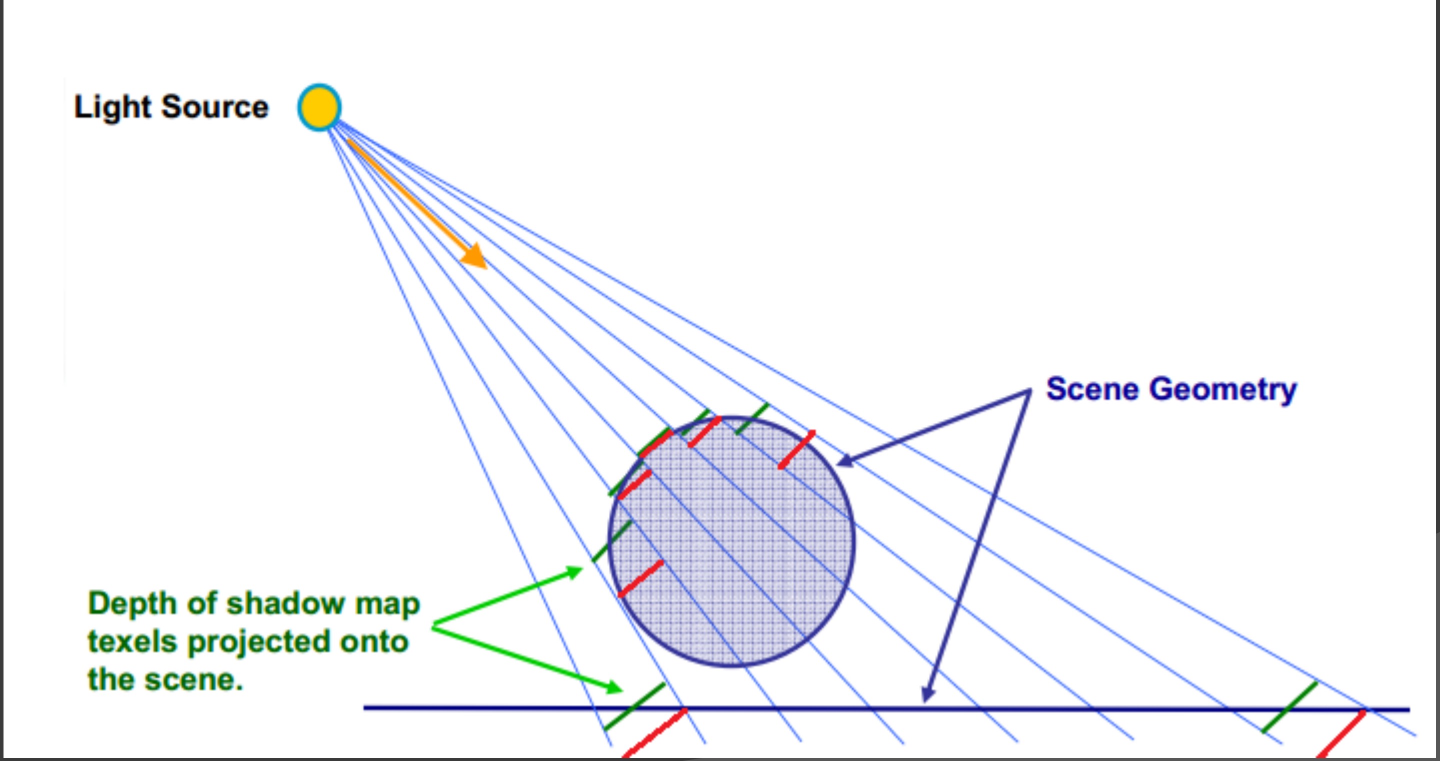
- Too little depth bias causes z-fighting
- Too much depth bias causes light leaking



http://developer.amd.com/media/gpu_assets/Isidoro-ShadowMapping.pdf

Depth Bias

- Bias should be dependent on triangle slope



Modified from: http://developer.amd.com/media/gpu_assets/Isidoro-ShadowMapping.pdf

Depth Bias

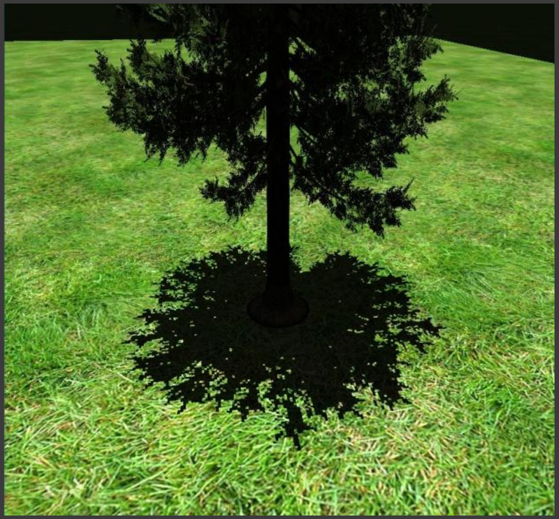
- Use screen space derivatives, which are calculated by hardware.
  - glPolygonOffset automatically calculates bias using screen space derivatives.
    - Takes a constant parameter and a slope-scaling parameter
    - Still need to tweak parameters for particular scenes
  - Use GLSL commands dFdx and dFdy to convert screen space neighbor pixels to light-space slopes
    - More involved and computationally expensive

Other Improvements

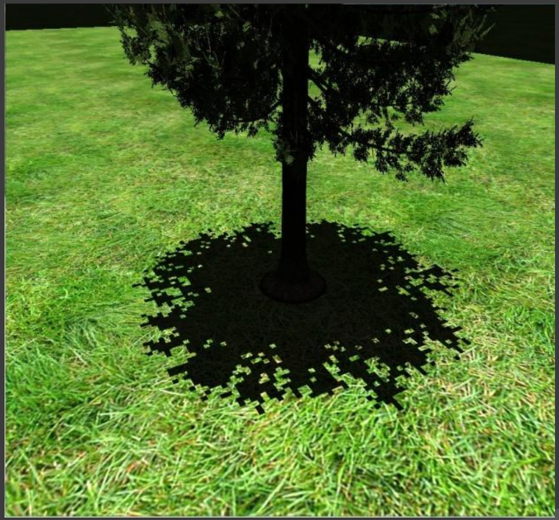
- Increase precision of depth buffer.
  - GL_DEPTH_COMPONENT16
  - GL_DEPTH_COMPONENT24
  - GL_DEPTH_COMPONENT32F
- Fit near and far plane of light frustum to fit the scene
- Increase shadow map resolution if possible
- Linearize depth-buffer.
  - The camera may be looking at part of the scene that is far away from the light, so we want the same amount of detail here as close to the light.

Texture Resolution

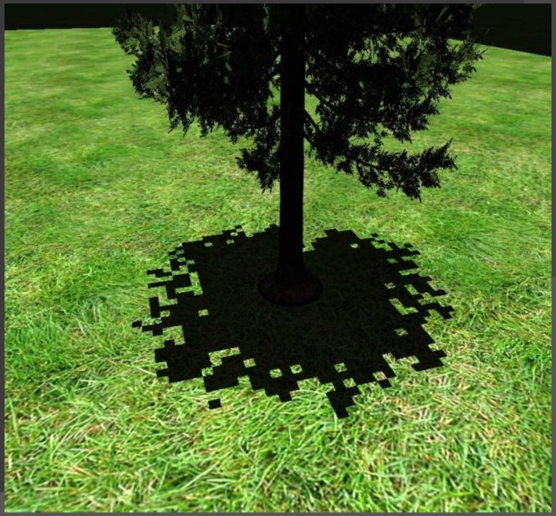
- Quality dependent on texture resolution
  - Low res shadow maps produce blocky results
  - High res shadow maps look better, but take up a lot of memory
  - What if the scene is really large?
    - A single texture cannot stretch across the whole world



2048 x 2048



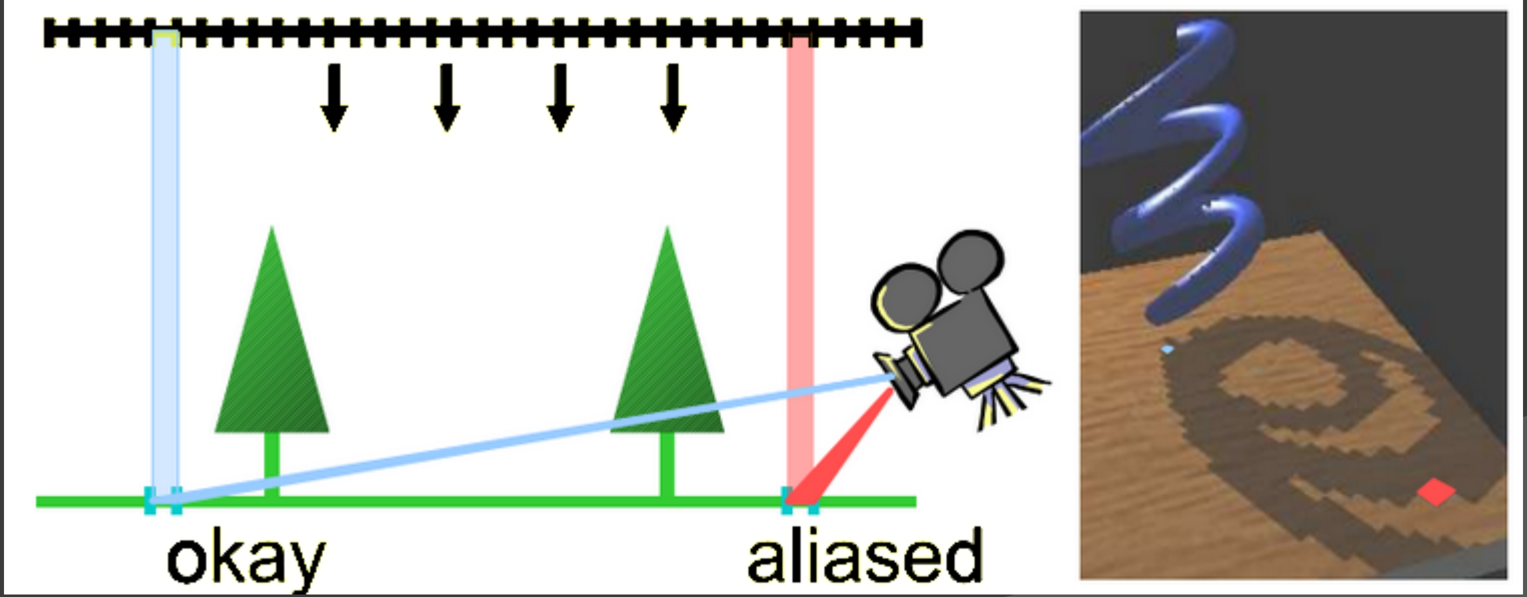
1024 x 1024



512 x 512

Perspective Aliasing

- Size of pixels in view space doesn’t match size of texels in shadow map.



http://www.cg.tuwien.ac.at/~scherzer/files/papers/LispSM_survey.pdf

Advanced Shadow Mapping

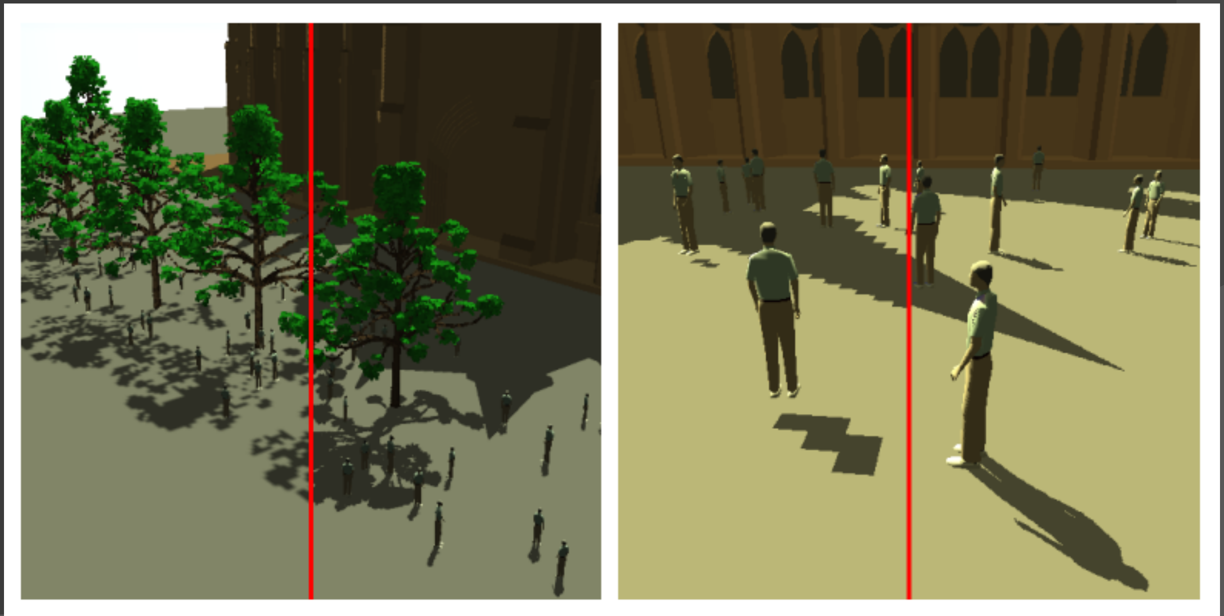
- Want to fix perspective aliasing
  - Need more detail near the eye, and less detail away from the eye.
- Want to handle texture resolution limits
  - Maintain constant texture resolution, independent of scene size
  - Center shadow map around eye
  - Shadow map should not cover areas that are out of view

Advanced Shadow Mapping

- Solutions:
  - Warping techniques
    - Perspective Shadow Maps (PSM)
    - Light Space Perspective Shadow Maps (LiSPSM)
    - Logarithmic Perspective Shadow Maps (LogPSM)
  - Frustum partitioning techniques
    - Cascaded Shadow Maps
      - aka Z-partitioning, parallel split maps
    - Sample Distribution Shadow Maps

Perspective Shadow Maps

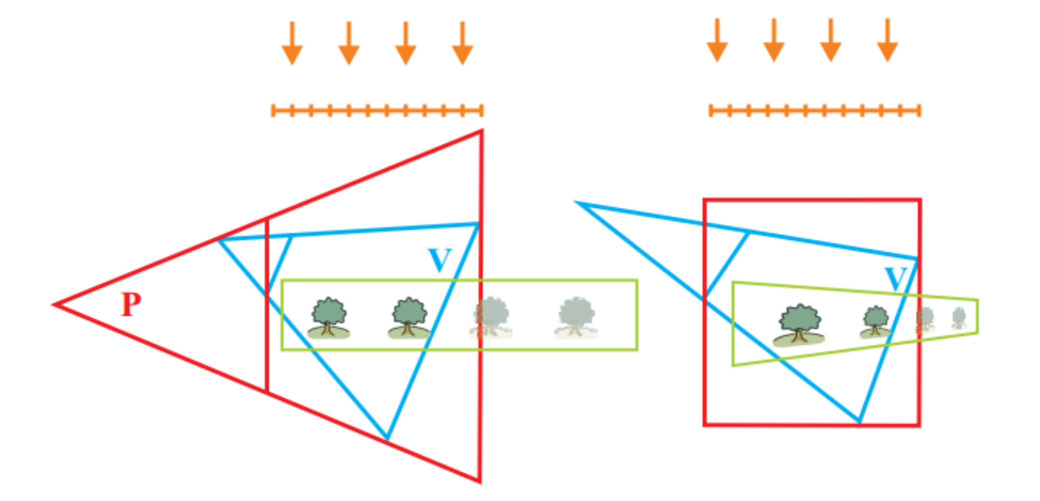
- Apply perspective transformation to scene before rendering into shadow map
  - Simply replace the standard view-projection matrix
- Skews shadow map so that there is more density near the eye.
- Still uses a single shadow map of the same resolution, but gets more out of it.



http://www-sop.inria.fr/reves/Basilic/2002/SD02/PerspectiveShadowMaps.pdf

Light Space Perspective Shadow Maps

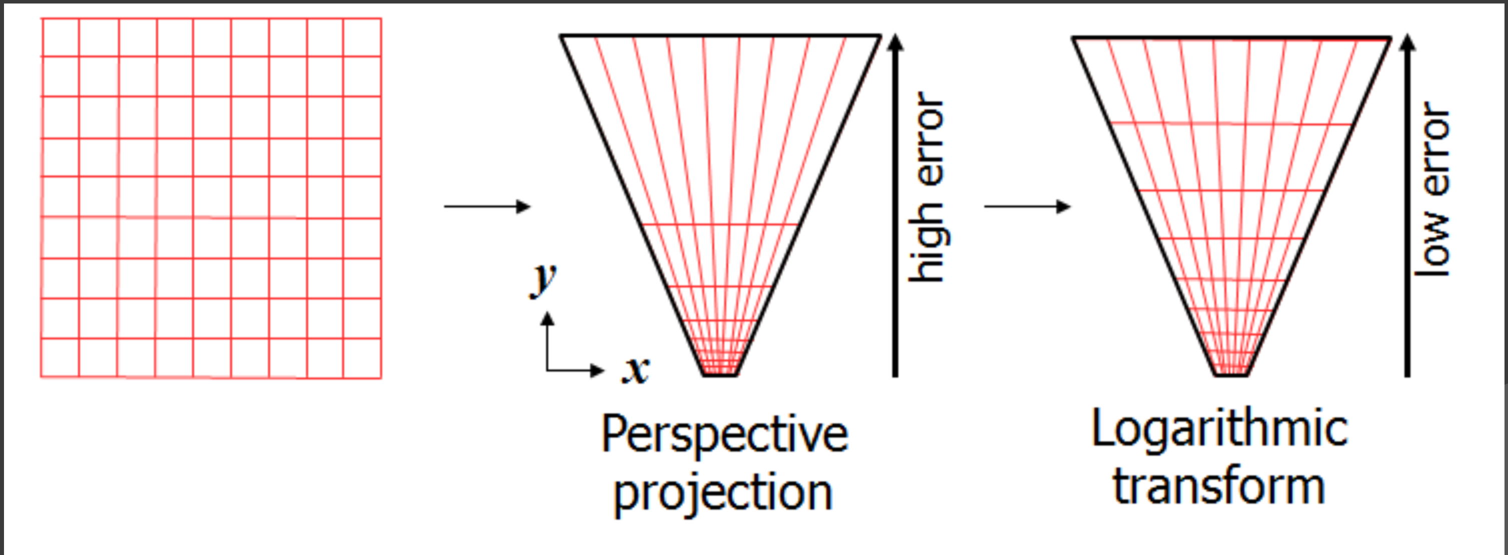
- Fixes limitations of Perspective Shadow Maps
  - Perspective transformation applied to light view-projection matrix rather than eye view-projection matrix.
    - Handles shadow casters that are behind the viewer
    - Lights do not change their type (PSM may convert directional lights to point lights, incorrectly)
    - Overall more stable and better-distributed error



http://www.cse.chalmers.se/~uffe/SIGGRAPH2012CourseNotes.pdf

Logarithmic Perspective Shadow Maps

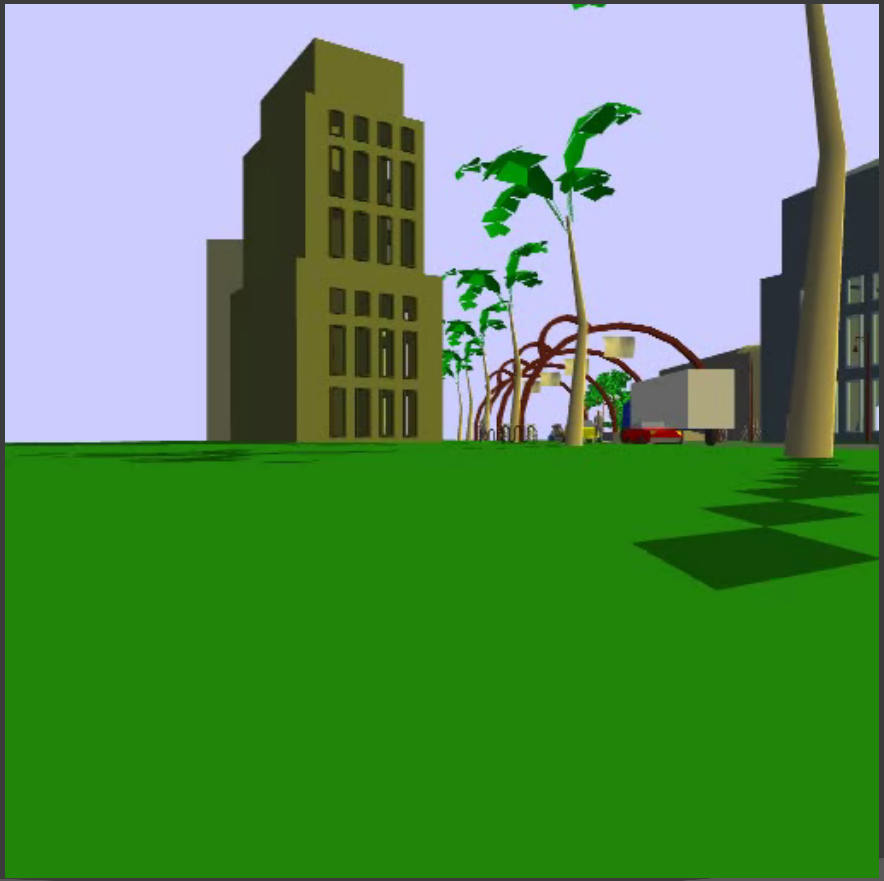
- Perspective projection + logarithmic transformation.
- Optimal constant error
- Requires logarithmic rasterization, which is not supported by current GPU hardware

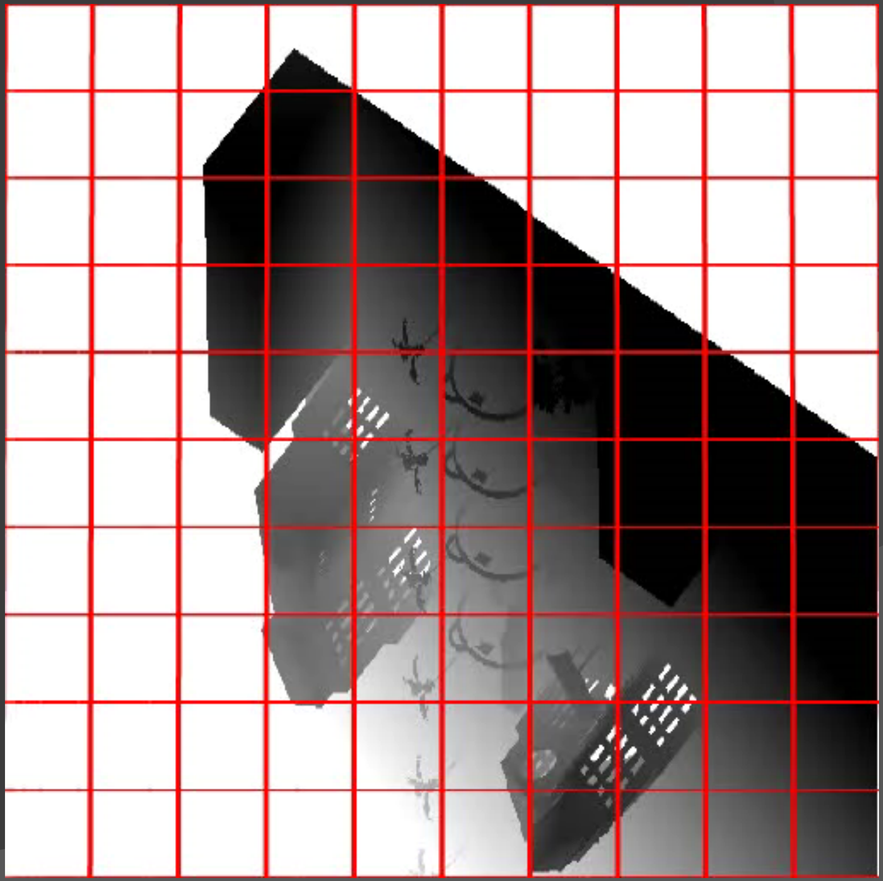


http://gamma.cs.unc.edu/LOGPSM/

Results

- Standard shadow map

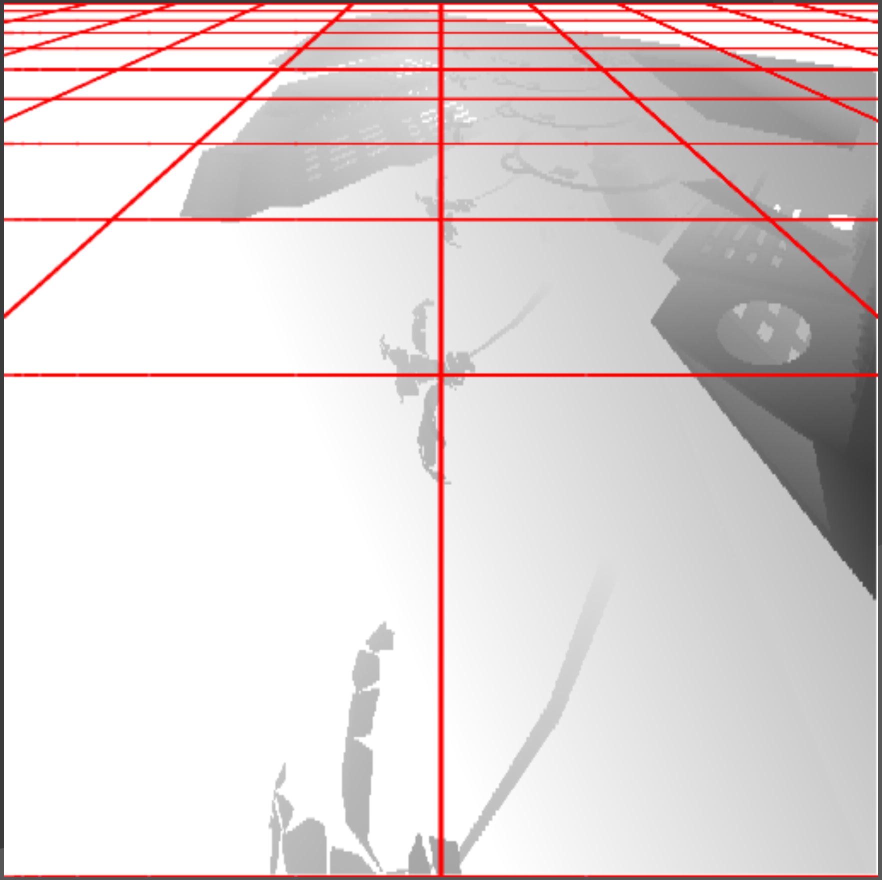




http://gamma.cs.unc.edu/LOGPSM/

Results

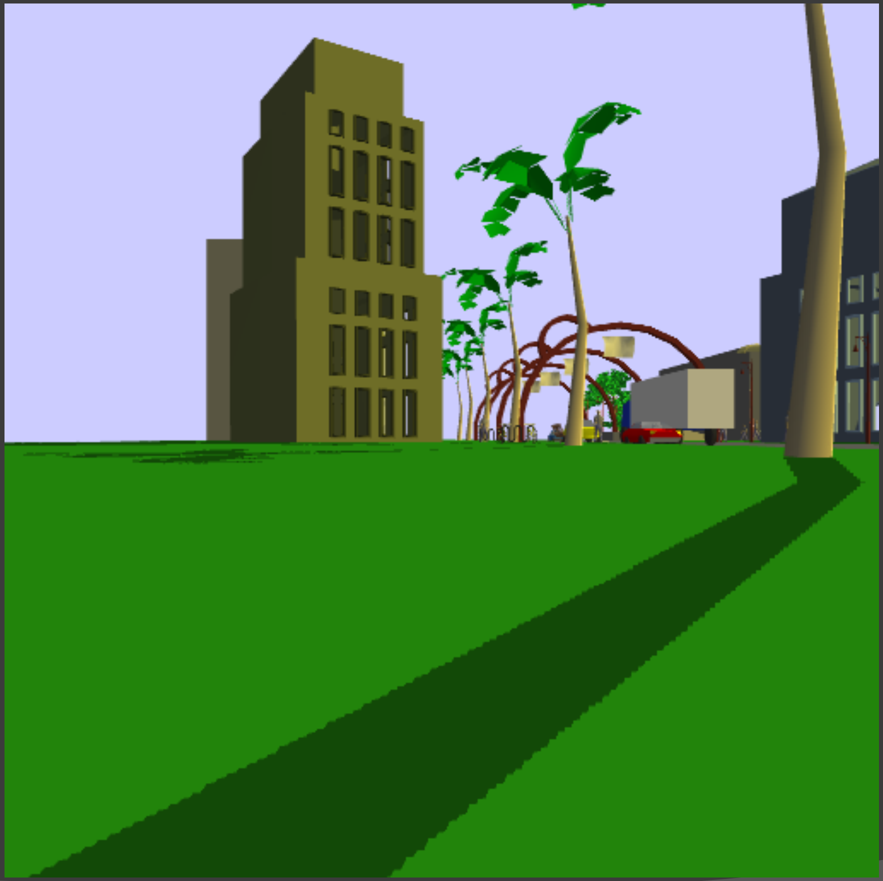
- Perspective Warping (LiPSM)

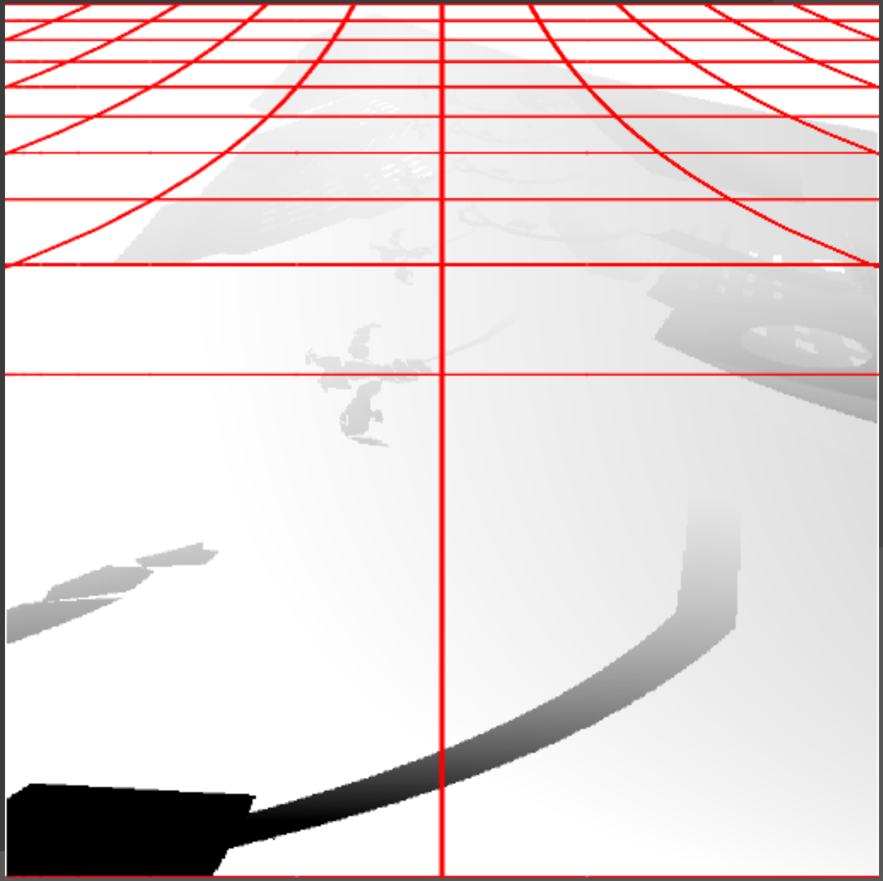


http://gamma.cs.unc.edu/LOGPSM/

Results

- Logarithmic Perspective Warping

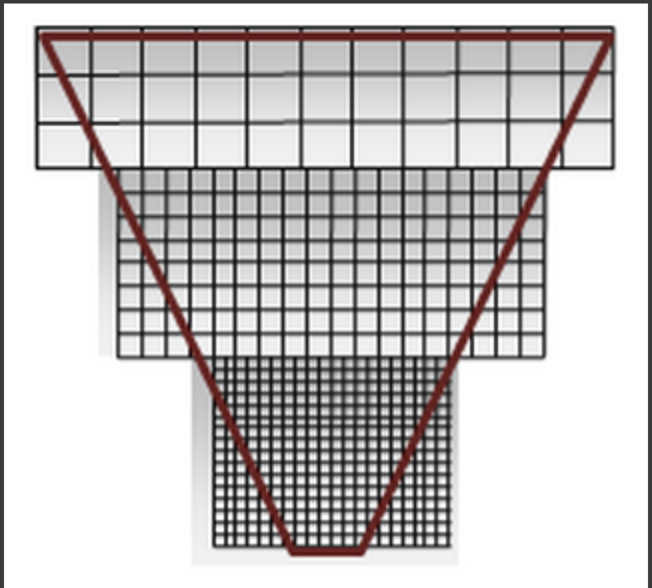


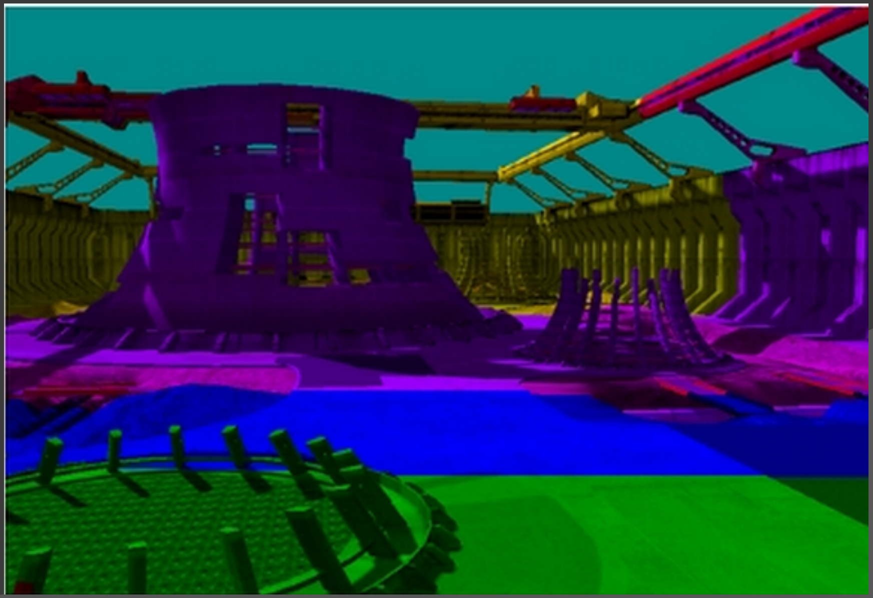


http://gamma.cs.unc.edu/LOGPSM/

Cascaded Shadow Maps

- Partition light frustum into multiple frusta
- Higher density near the eye, lower density away from the eye
- Each subfrustum gets its own shadow map. They are all the same size.
- Fragment shader samples the appropriate shadow map
- May use warping methods within each subfrustum

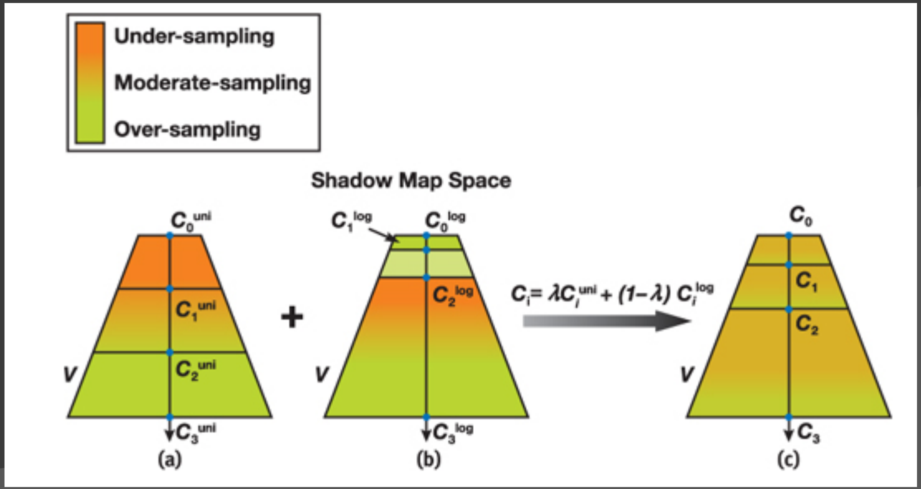




http://msdn.microsoft.com/en-us/library/windows/desktop/ee416307(v=vs.85).aspx

How to choose the partitions?

- Chose static partitions based on specific views
  - Birds eye view requires only a few cascades
  - Standard walking view requires multiple cascades when the scene extends far.
  - Annoying to always tweak parameters
- Split types
  - Artist determined
  - Uniform
  - Logarithmic
  - Find midpoint



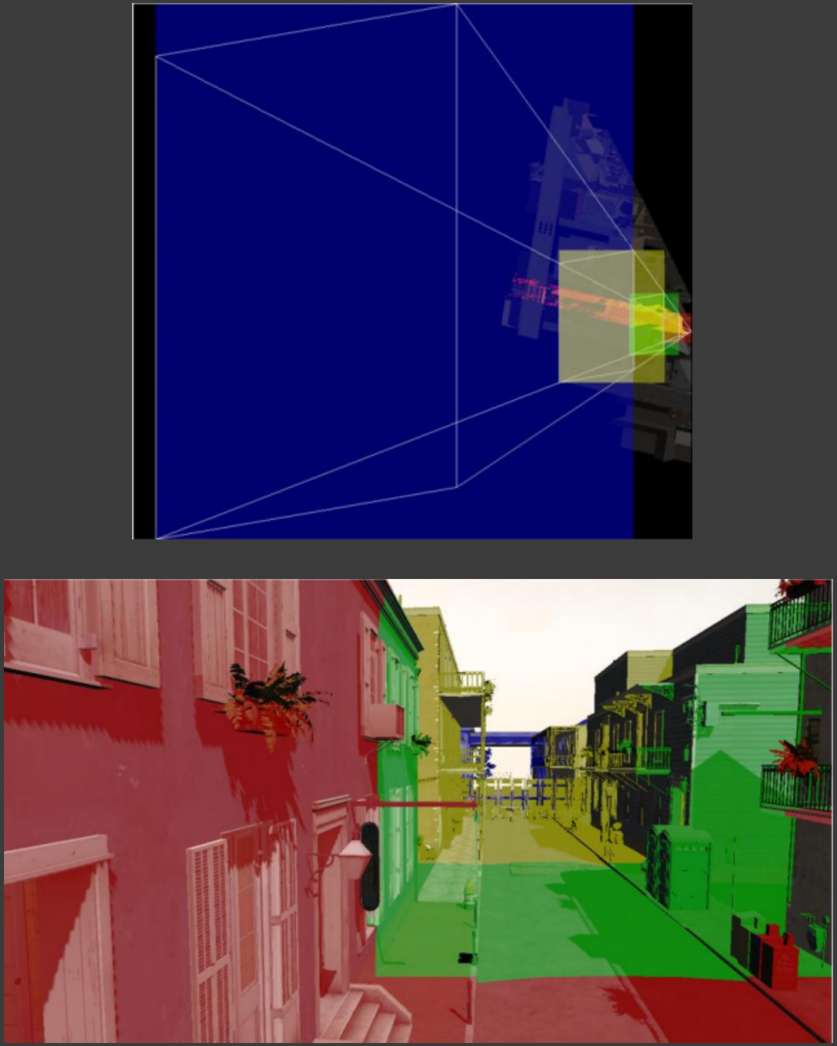
http://http.developer.nvidia.com/GPUGems3/gpugems3_ch10.html

How to choose the partitions?

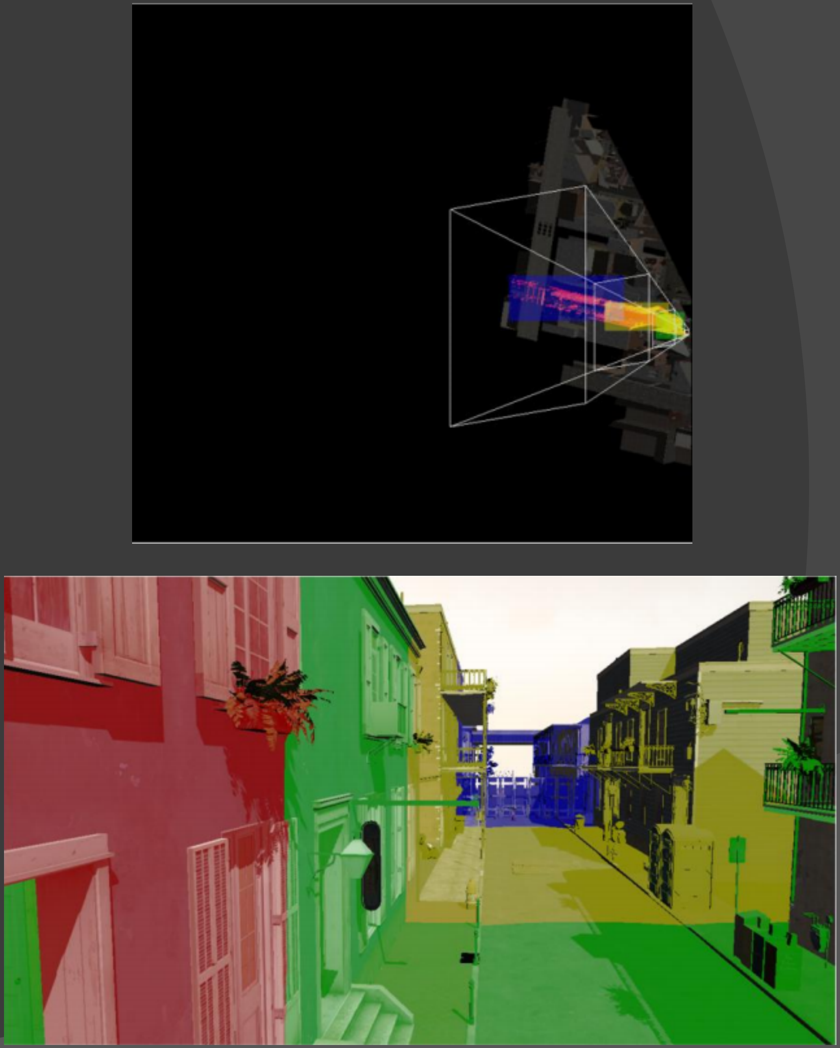
- Better to have a dynamic approach
- Sample Distribution Shadow Maps
  - Use geometry information and occlusion tests to create tightly bound frusta.
  - Approximate logarithmic splits
  - A more complicated approach involves analyzing the z distribution of samples (from the camera’s point of view) in a compute shader.

Results

Standard Cascaded Shadow Maps



Sample Distribution Shadow Maps



http://visual-computing.intel-research.net/art/publications/sdsm/sampleDistributionShadowMaps_SIGGRAPH2010_notes.pdf

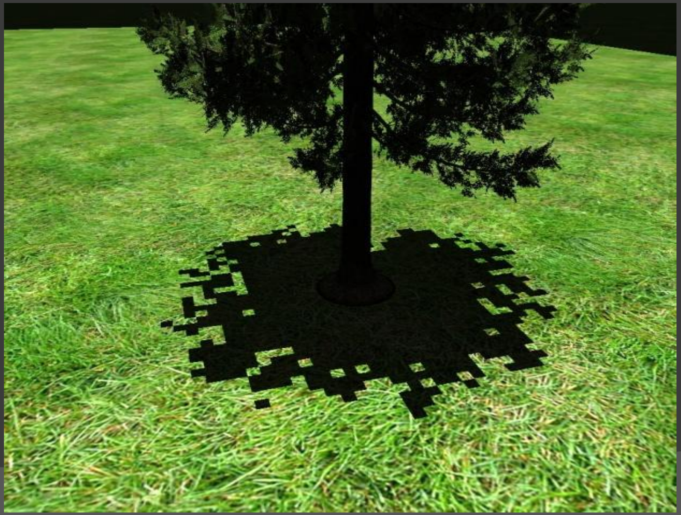
Smoothing Hard Shadows

- Linear Filtering
- Percentage Closer Filtering
- Variance Shadow Maps

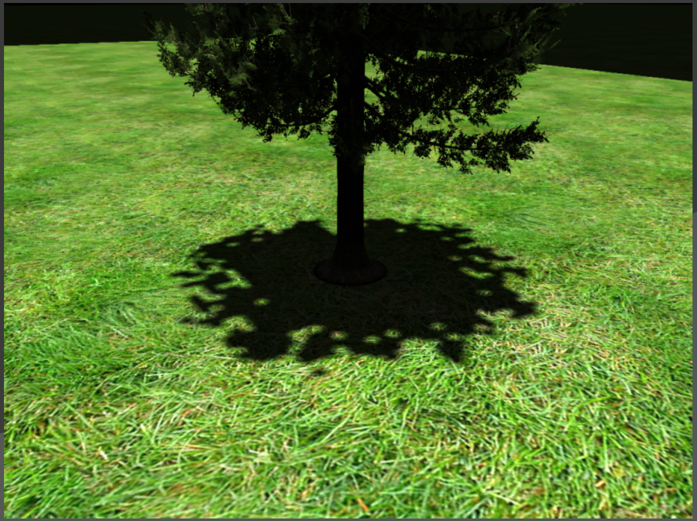
Linear Filtering

- Enable linear filtering on shadow map texture
  - Interpolates depth between 2x2 region of pixels instead of just choosing the depth of the closest pixel
- Really, really simple. But not exactly correct.

512x512 with GL_NEAREST

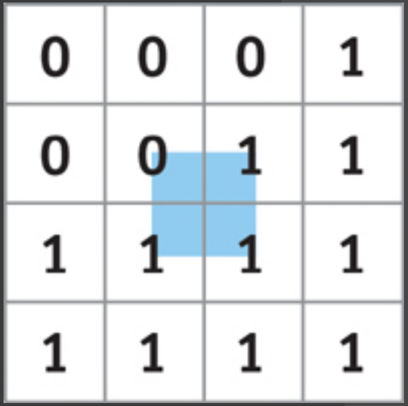


512x512 with GL_LINEAR



Percentage Closer Filtering

- Simulate soft shadows by looking at neighboring shadow texels.
- Take 4 nearest samples in shadow map
  - Use GLSL command *textureGather*
- Compare the surface’s depth with each of these samples.
  - Supply surface depth to *textureGather* call
- Bilinearly interpolate the results
- Different than linear filtering, which interpolates the depth values and not the results of the comparisons.
- To get a larger penumbra, sample 16 nearest texels with 4 textureGather calls.

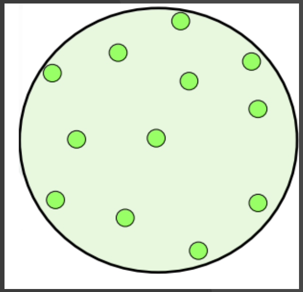


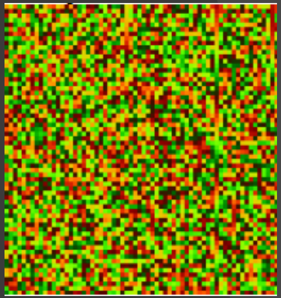
0	0	0	1
0	0	1	1
1	1	1	1
1	1	1	1

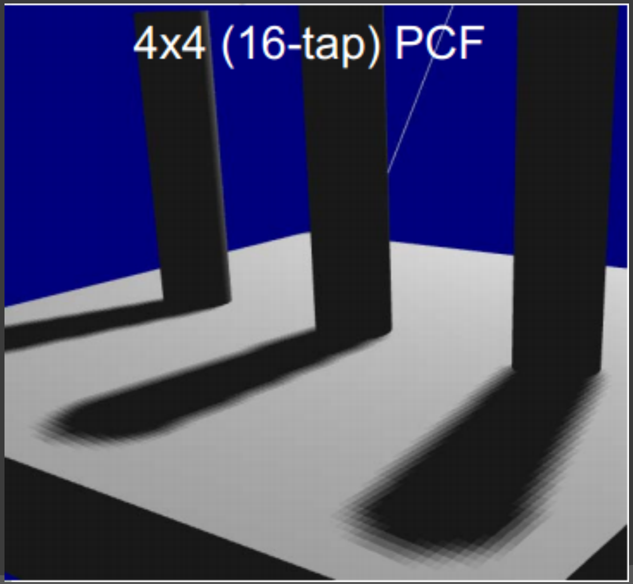
http://http.developer.nvidia.com/GPUGems/gpugems_ch11.html

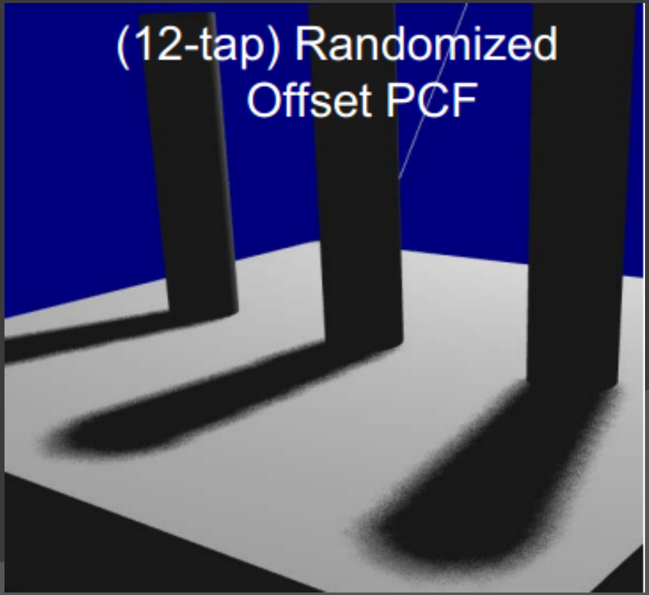
Percentage Closer Filtering

- Sample random neighbors to get a less patterned look
  - Use a non-uniform disk. Rotate the disk using random rotations stored in a texture.





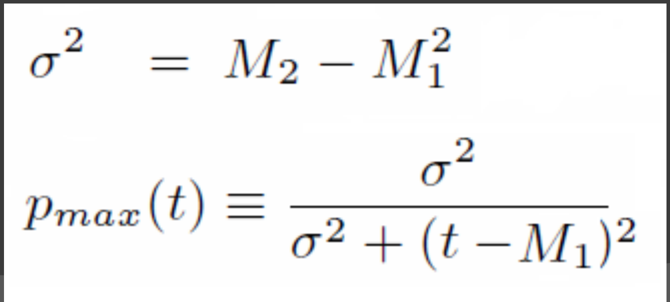




http://developer.amd.com/media/gpu_assets/Isidoro-ShadowMapping.pdf

Variance Shadow Maps

- Store depth in one map, and depth² in another
- Filter these maps to your liking
  - Mip-map
  - Gaussian blur
  - Summed area tables
- Determine fragment's shadow strength through a probability function



$$\sigma^2 \quad = \quad M_2 - M_1^2$$

$$p_{max}(t) \equiv \frac{\sigma^2}{\sigma^2 + (t - M_1)^2}$$

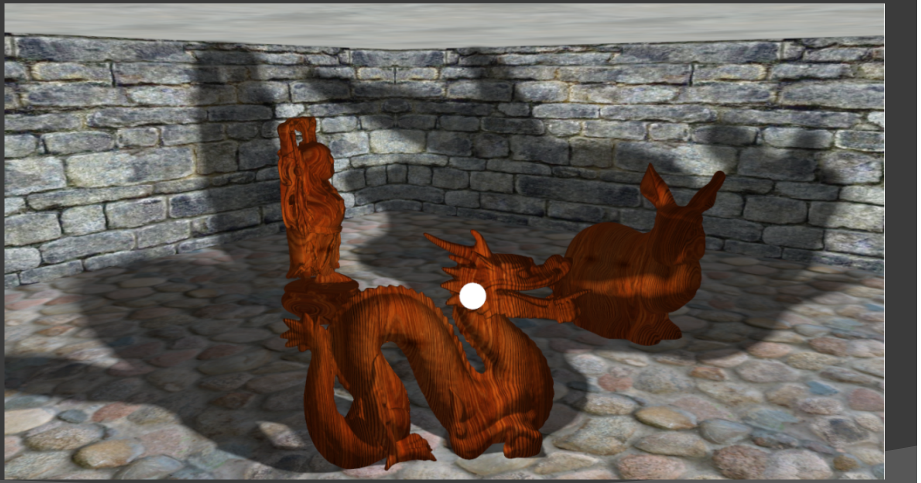
Where M_1 = shadow map sample
M_2 = depth² shadow map sample
t = fragment depth
σ^2 = variance
p_{max} = max % of samples in light

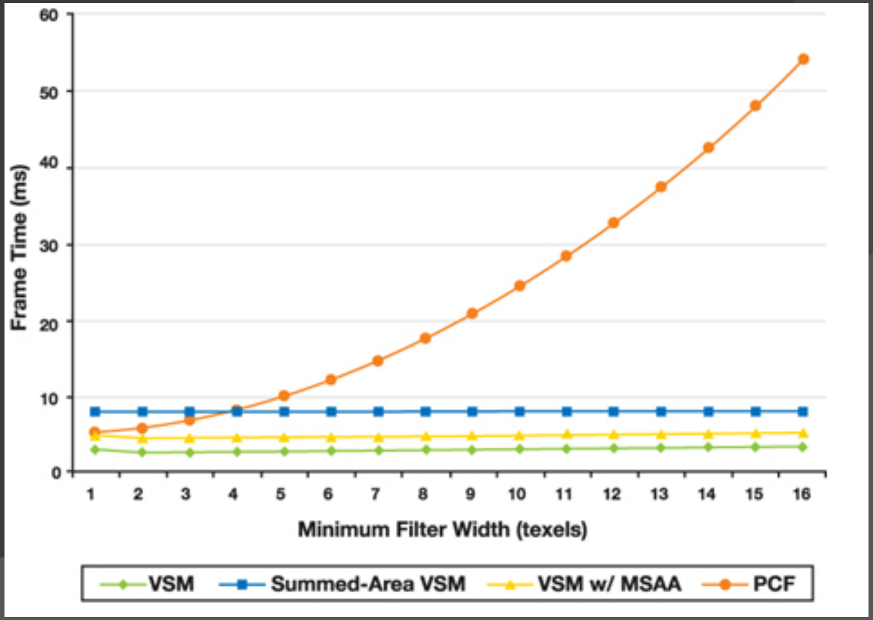
http://graphics.stanford.edu/~mdfisher/Shadows.html

Variance Shadow Maps

- Advantages:
  - Able to capture large penumbra at a much smaller cost than PCF
  - Does a great job of blurring low res shadow maps

21 x 21 blur filter kernel with Six 512x512 shadow maps
Running at 141 fps

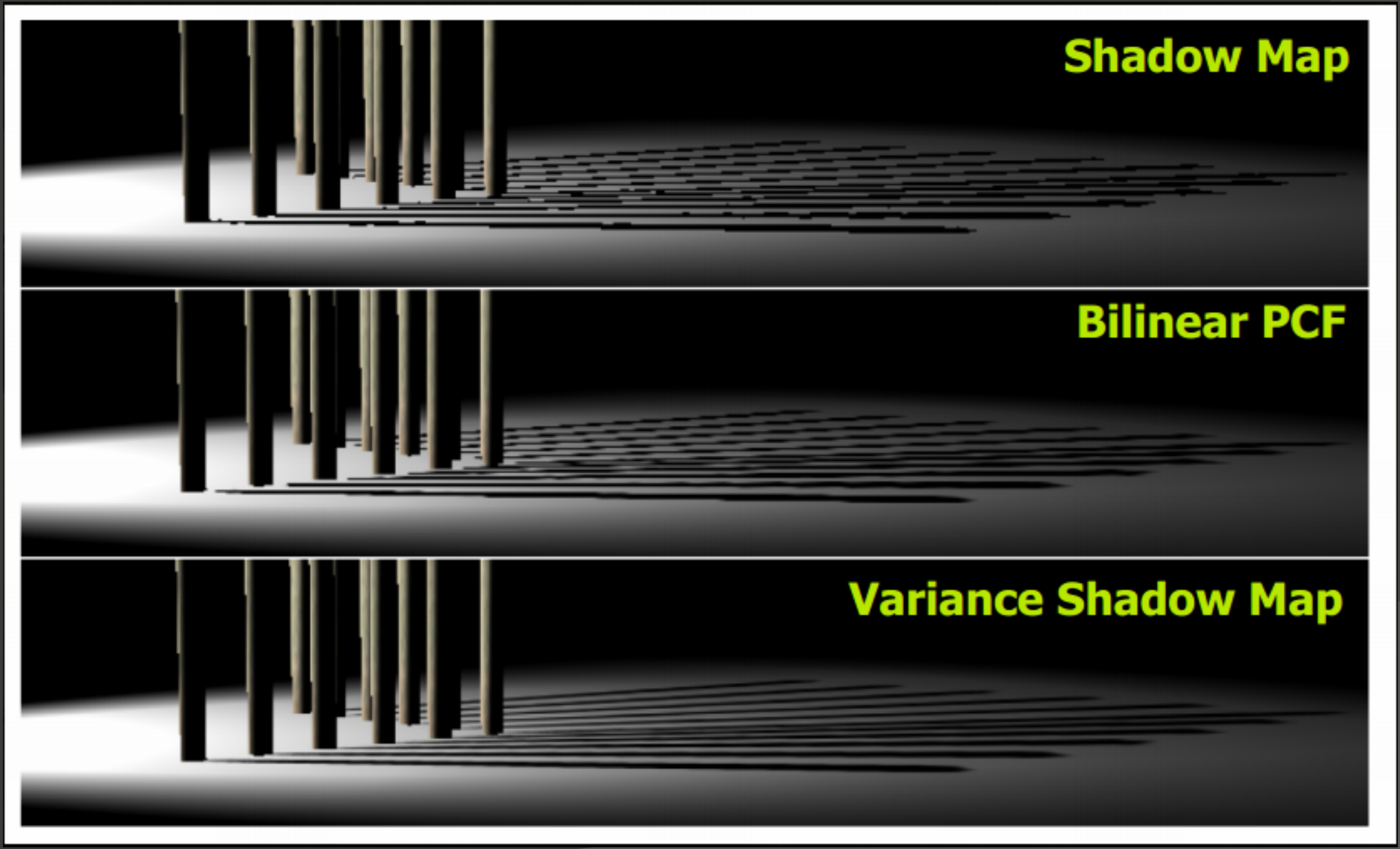




http://graphics.stanford.edu/~mdfisher/Shadows.html

http://http.developer.nvidia.com/GPUGems3/gpugems3_ch08.html

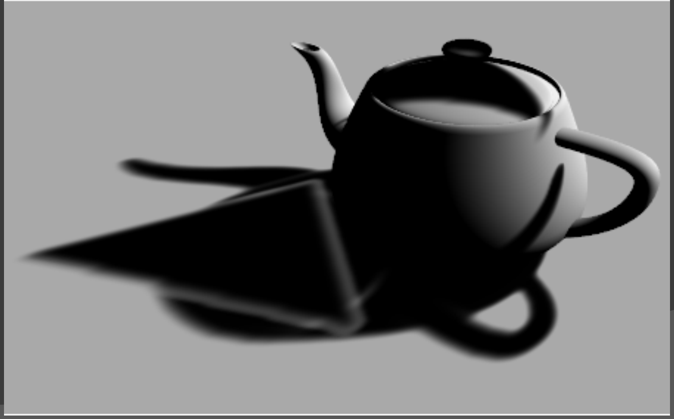
Variance Shadow Maps



http://http.download.nvidia.com/developer/presentations/2006/gdc/2006-GDC-Variance-Shadow-Maps.pdf

Variance Shadow Maps

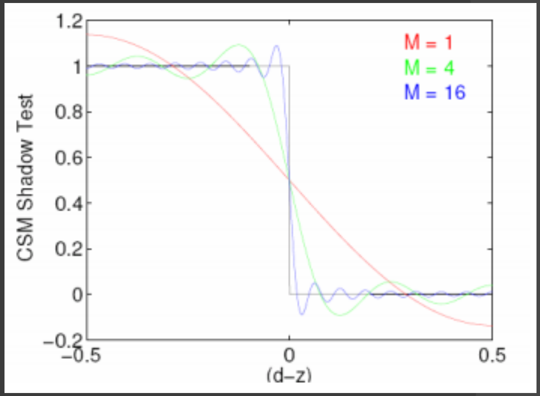
- Main problem: light bleeding
  - Happens when more than two occluders that are far apart shadow the same region.
  - At the lit edges, the probability function uses the depth of the triangle to estimate the shadow strength.
    - The depth map can only store one sample per pixel, so there is no understanding of a second, closer occluder (the teapot)
  - Somehow we need to know the depth distribution …

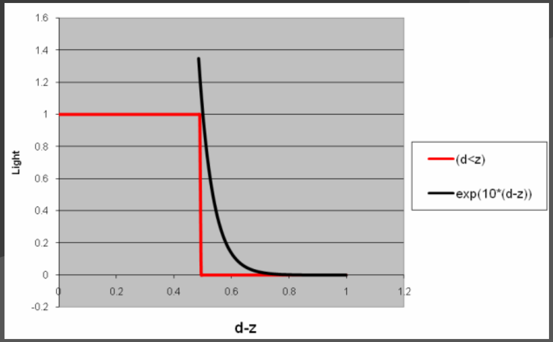


http://pixelstoomany.wordpress.com/category/shadows/light-bleeding/

Variance Shadow Maps

- Light Bleeding solutions
  - Add variance threshold so that you never get very light areas
    - But makes the shadow falloff too strong
  - Represent discrete depth steps with smooth functions
    - Convolution shadow maps
      - Approximate depth steps with 1D Fourier expansion
    - Exponential shadow maps
      - Approximate step function with exponentia function





http://developer.download.nvidia.com/presentations/2008/GDC/GDC08_SoftShadowMapping.pdf

Which techniques to use?

- What shadow technique family should I use?
  - Shadow Volumes or Shadow Mapping?
  - Shadow Mapping
- Which technique should I use to combat perspective aliasing and shadow map resolution issues?
  - Warped perspective shadow maps or Cascaded Shadow Maps?
  - Cascaded Shadow Maps
- Which soft shadow technique should I use?
  - Percentage Closer Filtering for hard shadows with soft edges
  - Variance Shadow Maps for very soft shadows
- Which technique should I use to fix light bleeding?
  - Exponential Shadow Maps
- What more can I do to make my shadows better?
  - Use ambient occlusion to approximate global illumination
  - Screen space effects like depth of field and motion blur can help smooth shadow artifacts.

WebGL demo

http://codeflow.org/entries/2013/feb/15/soft-shadow-mapping/

Resources

- Every image credit leads to a valuable resource on that topic.
- I referenced the following books:
  - *Real-Time Rendering*, Third Edition by Akenine-Moller, Haines, and Hoffman
  - *Real-Time Shadows* by Eisemann, Schwarz, Assarsson, and Wimmer