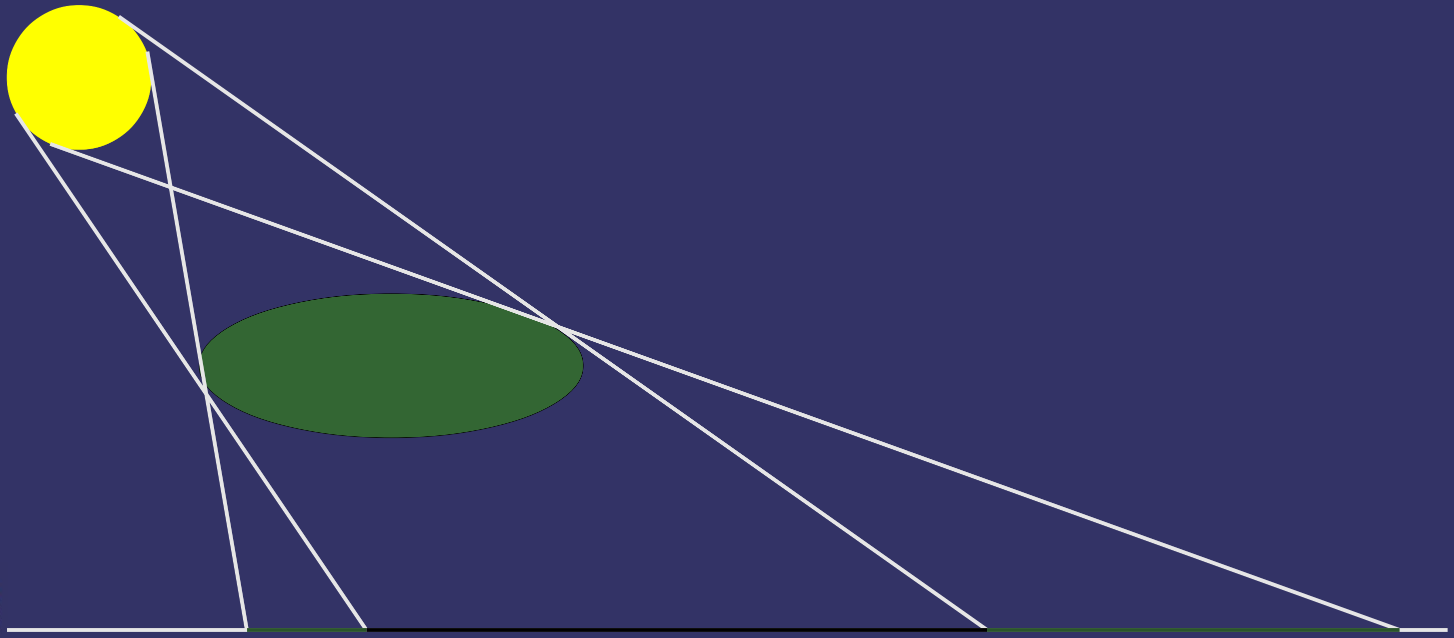
VGP353 – Week 3

- Agenda:
  - Quiz #1
  - More shadow maps:
    - Percentage closer soft shadows (PCSS)
    - Parallel-split shadow maps (PSSMs)
  - Assignment #2 (shadow maps) started

© Copyright Ian D. Romanick 2008, 2011 BY-NC-SA

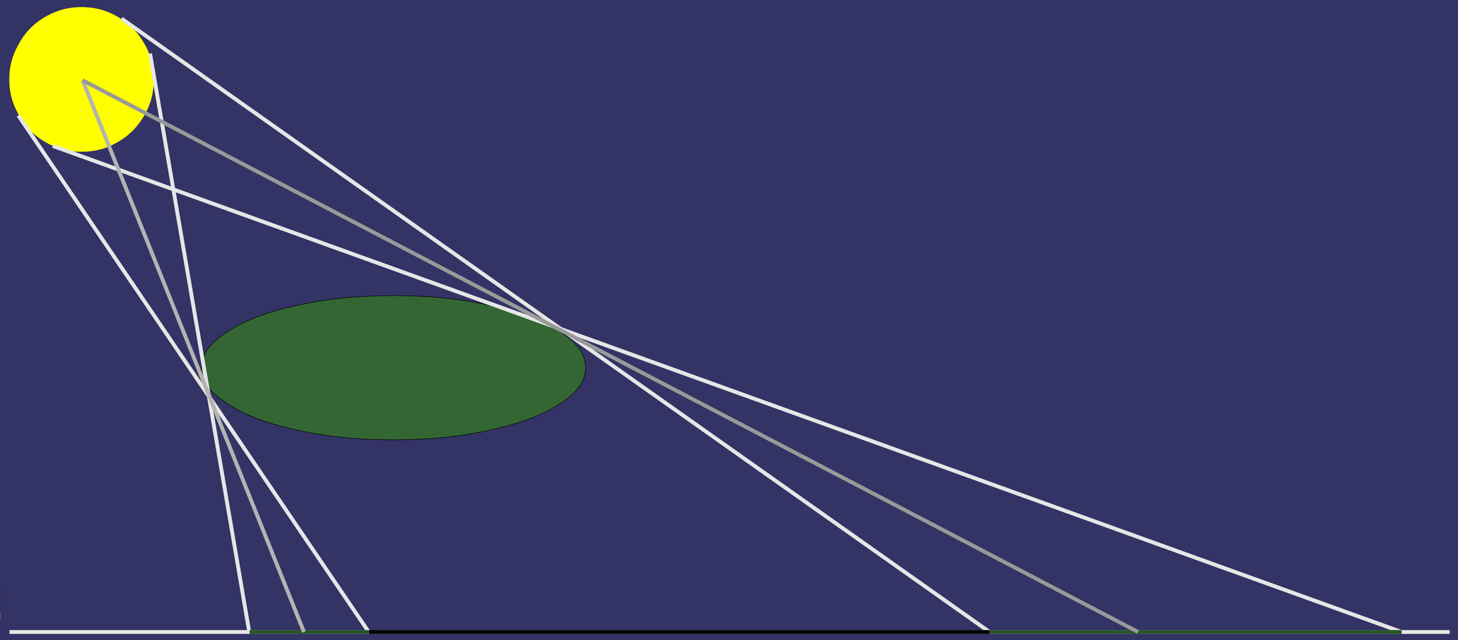
Soft Shadows

- 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



© Copyright Ian D. Romanick 2008, 2011 BY-NC-SA

Soft Shadows

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

© Copyright Ian D. Romanick 2008, 2011 (cc) BY-NC-SA

Soft Shadows

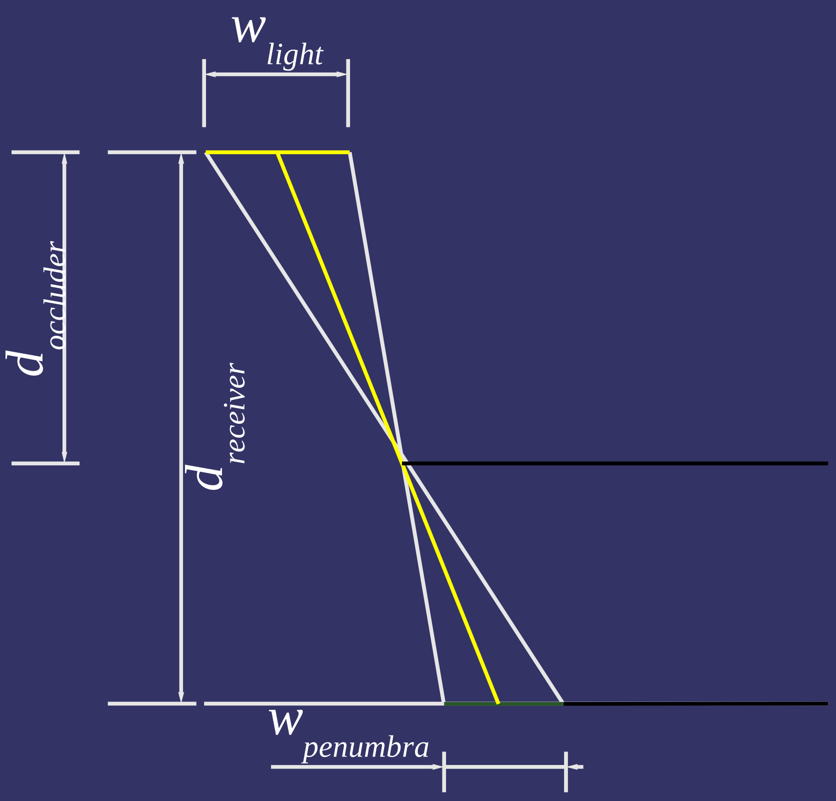
- Size of penumbra region varies with:
  - Size of light
  - Distance between occluder and light
  - Distance between occluder and receiver
- Using this information to perform *correct* light visibility calculations is hard
  - Make some simplifying assumptions!
  - Assume that all occluders, receivers,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

© Copyright Ian D. Romanick 2008, 2011 (cc) BY-NC-SA

Soft Shadows

- Estimate penumbra size using:

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



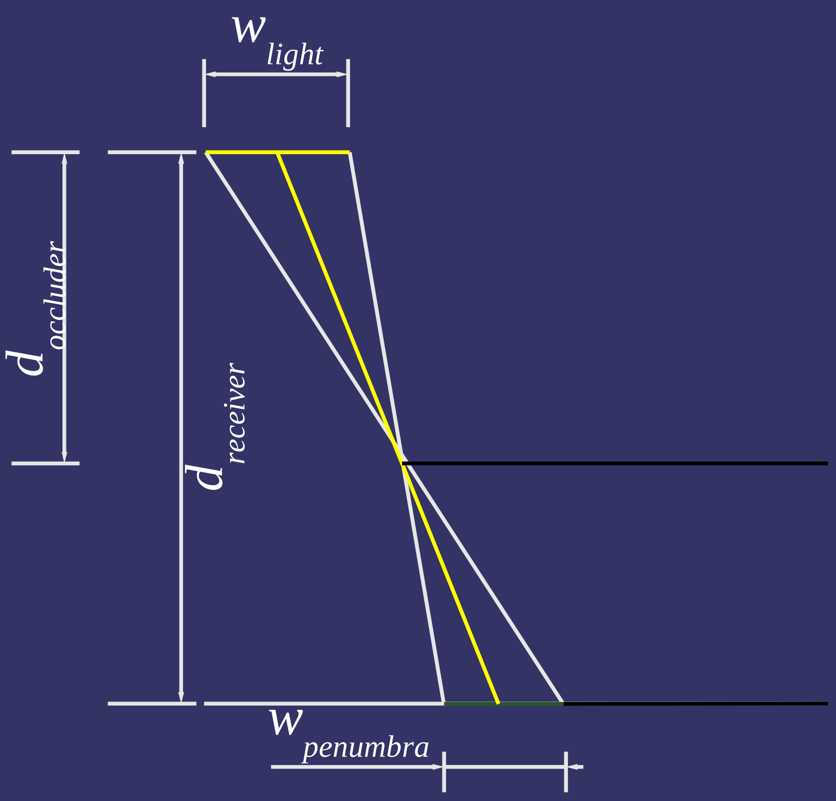
© Copyright Ian D. Romanick 2008, 2011 (cc) BY-NC-SA

Soft Shadows

- Estimate penumbra size using:

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

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



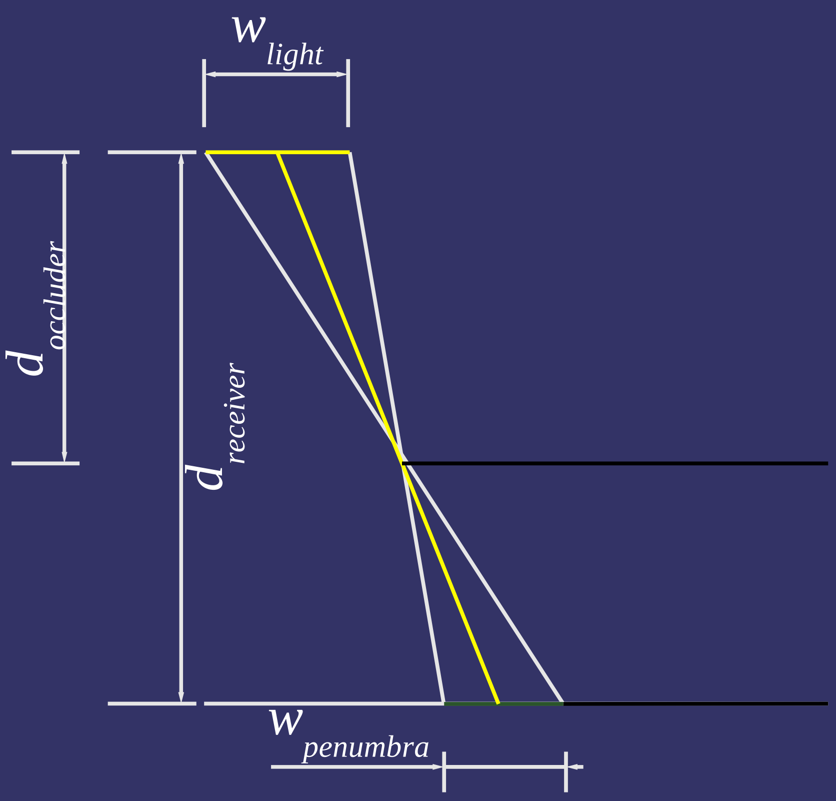
© Copyright Ian D. Romanick 2008, 2011 (cc) BY-NC-SA

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



© Copyright Ian D. Romanick 2008, 2011 (cc) BY-NC-SA

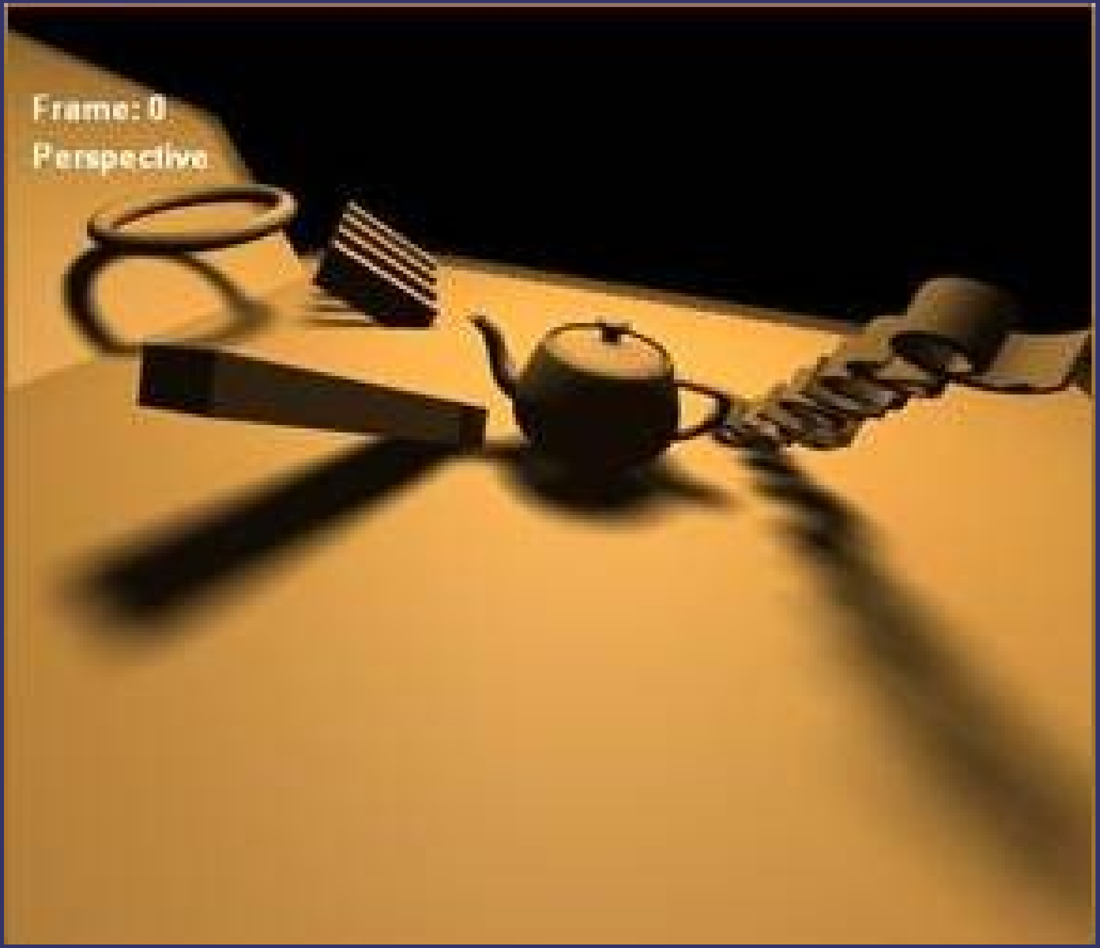
Soft Shadows

- 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

© Copyright Ian D. Romanick 2008, 2011 (cc) BY-NC-SA

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

© Copyright Ian D. Romanick 2008, 2011 (cc) BY-NC-SA

Shadow Map Aliasing

- A shadow map texel represents an area $d_s \times d_s$
  - d_s is the reciprocal of the shadow map resolution
    - As shadow map resolution increases, d_s decreases
  - As r_s increases, each texel covers more of the surface
  - The projected size of a surface at distance r_s is approximately:

$$\frac{d_s r_s}{N \cdot L}$$

Shadow Map Aliasing

- An image pixel represents an area $d_i \times d_i$
  - d_i is the reciprocal of the image resolution
    - As image resolution increases, d_i decreases
  - As r_i increases, each pixel covers more of the surface
  - The projected size of a surface at distance r_i is approximately:

$$\frac{d_i r_i}{N \cdot V}$$

© Copyright Ian D. Romanick 2008, 2011 (cc) BY-NC-SA

Shadow Map Aliasing

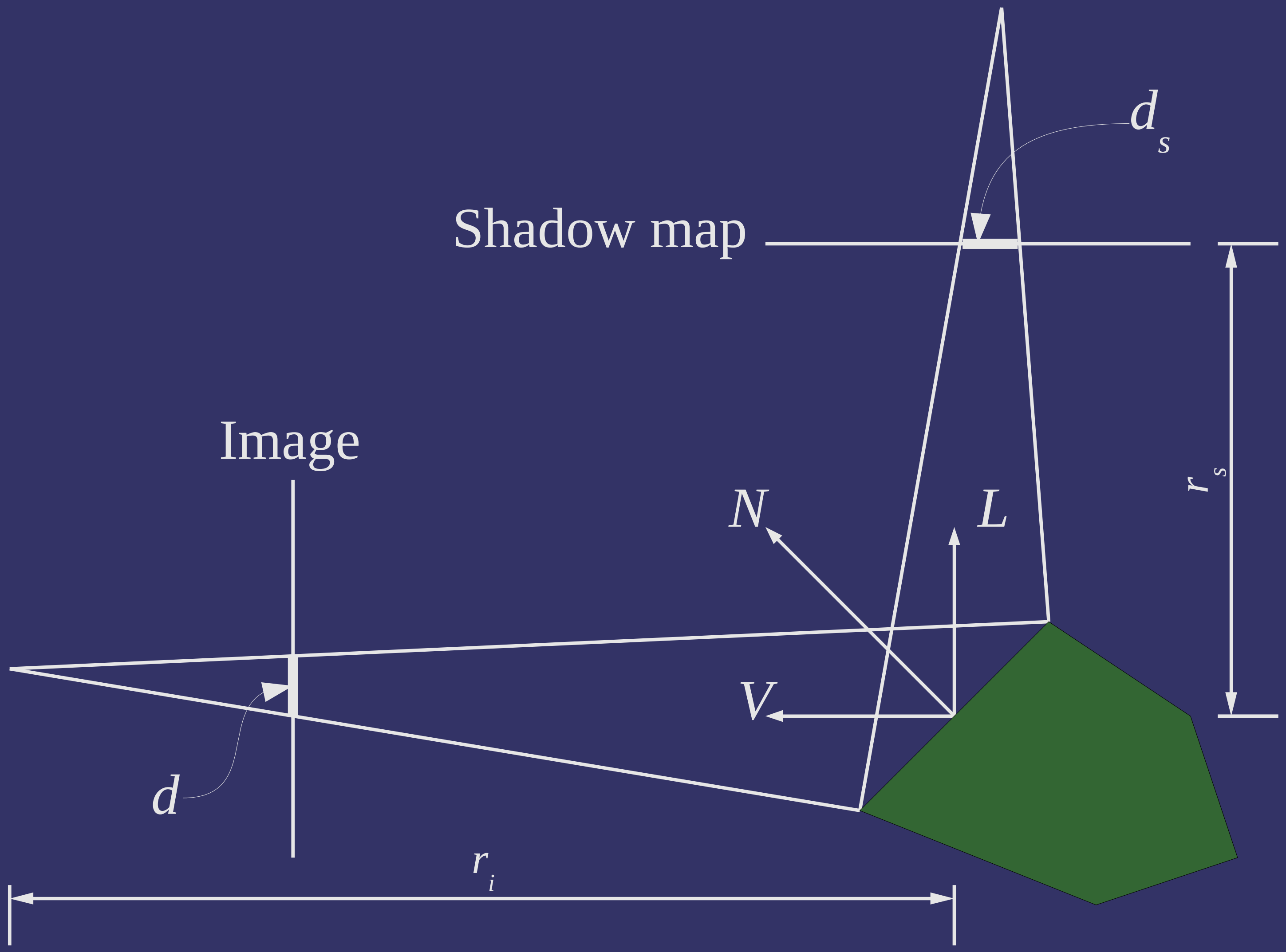
- The size of the projection of the shadow texel in the final image is:

$$d = d_s \frac{r_s}{r_i} \frac{N \cdot V}{N \cdot L}$$

  - Aliasing occurs when $d > d_i$

© Copyright Ian D. Romanick 2008, 2011 (cc) BY-NC-SA

Shadow Map Aliasing



© Copyright Ian D. Romanick 2008, 2011 (cc) BY-NC-SA

Shadow Map Aliasing

- The size of the projection of the shadow texel in the final image is:

$$d = d_s \frac{r_s}{r_i} \frac{N \cdot V}{N \cdot L}$$

  - Aliasing occurs when $d > d_i$
  - Matches intuition:
    - If the shadow area is small in the shadow map but large in the final image, there will be aliasing.

Shadow Map Aliasing

$$d_s \frac{r_s}{r_i} \frac{N \cdot V}{N \cdot L}$$

Large when light rays are nearly tangent to surface geometry, but surface geometry faces towards the viewer

- This is called *projection aliasing*
- Dependent on orientation of scene geometry
- Can change even when light and viewer are stationary
- Difficult to fix!

© Copyright Ian D. Romanick 2008, 2011 BY-NC-SA

Shadow Map Aliasing

$$d_s \frac{r_s}{r_i} \frac{N \cdot V}{N \cdot L}$$

Occurs when the view is close to individual texels of the shadow map

- This is called *perspective aliasing*
- Occurs if the shadow map is too small (i.e., d_s is large)
- Can only increase shadow map size so much!
- Also occurs if $r_s \gg r_i$

Large when light rays are nearly tangent to surface geometry, but surface geometry faces towards the viewer

- This is called *projection aliasing*
- Dependent on orientation of scene geometry
- Can change even when light and viewer are stationary
- Difficult to fix!

© Copyright Ian D. Romanick 2008, 2011 (cc) BY-NC-SA

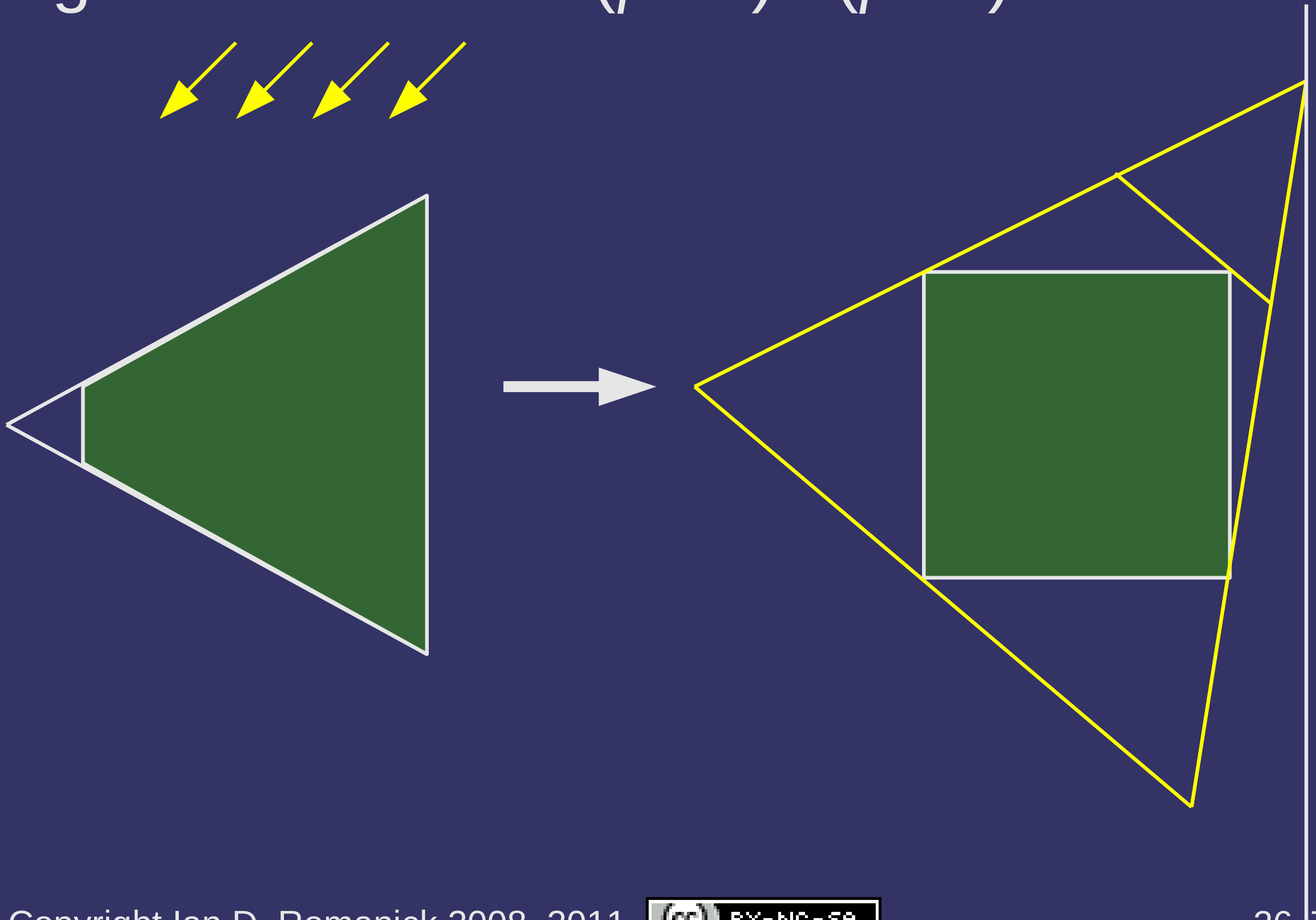
Perspective Shadow Maps

- If the problem stems from the relationship between the camera frustum and light frustum, then the solution make take both frusta into account
  - Perform shadow map calculations in post-projection camera space *instead of* world space
  - The projection remaps the frustum volume to a cube, this cube is then sampled to create the shadow map
    - Applying this to the world before applying the light's view effectively changes the “shape” of the light

© Copyright Ian D. Romanick 2008, 2011 (cc) BY-NC-SA

Perspective Shadow Maps

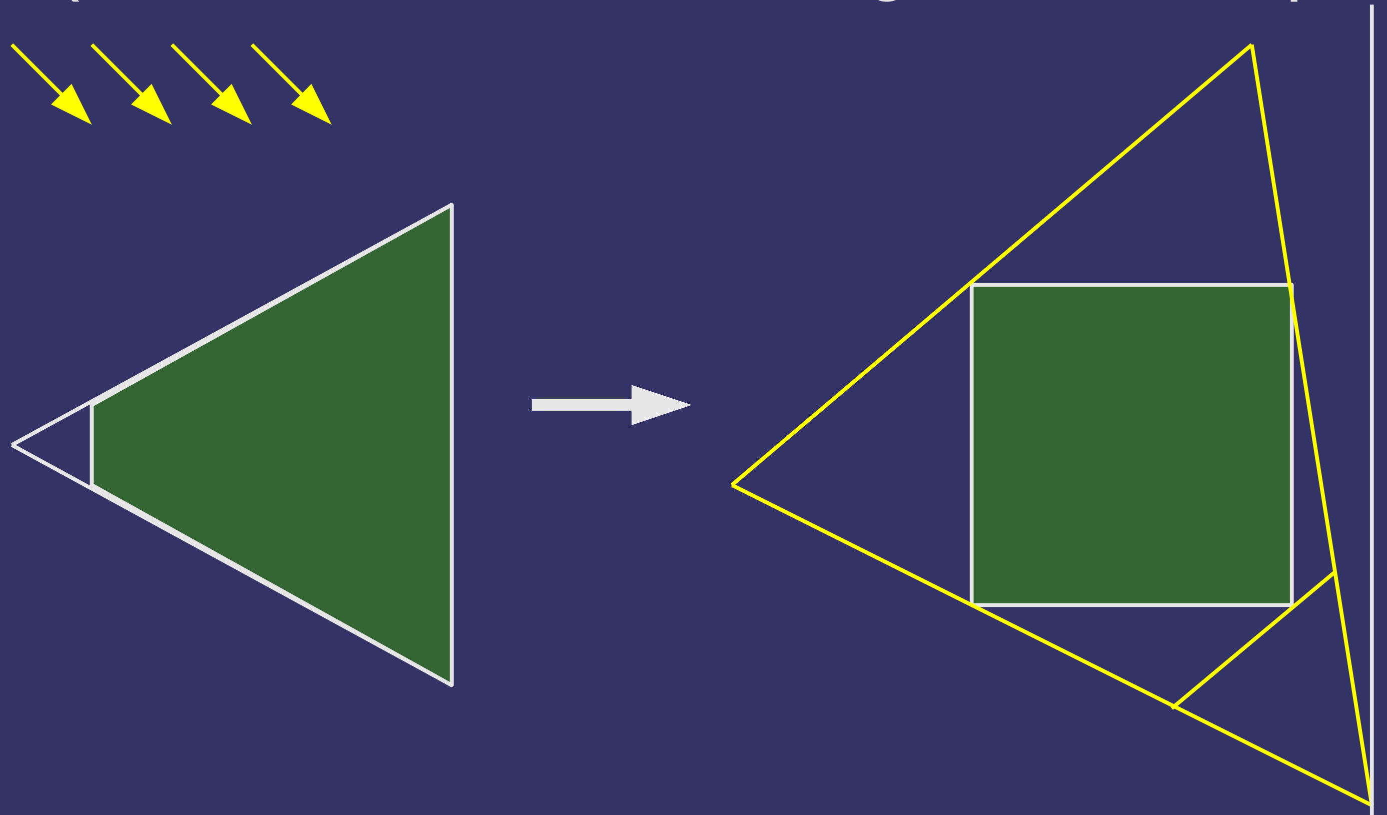
- Directional lights become point lights "on the infinity plane"
  - The light's Z becomes $(f + n) / (f - n)$



© Copyright Ian D. Romanick 2008, 2011 BY-NC-SA

Perspective Shadow Maps

- Directional front-lights become inverted
  - Reverse the order of the usual depth and shadow tests (i.e., less-than becomes greater-or-equal)



© Copyright Ian D. Romanick 2008, 2011 BY-NC-SA

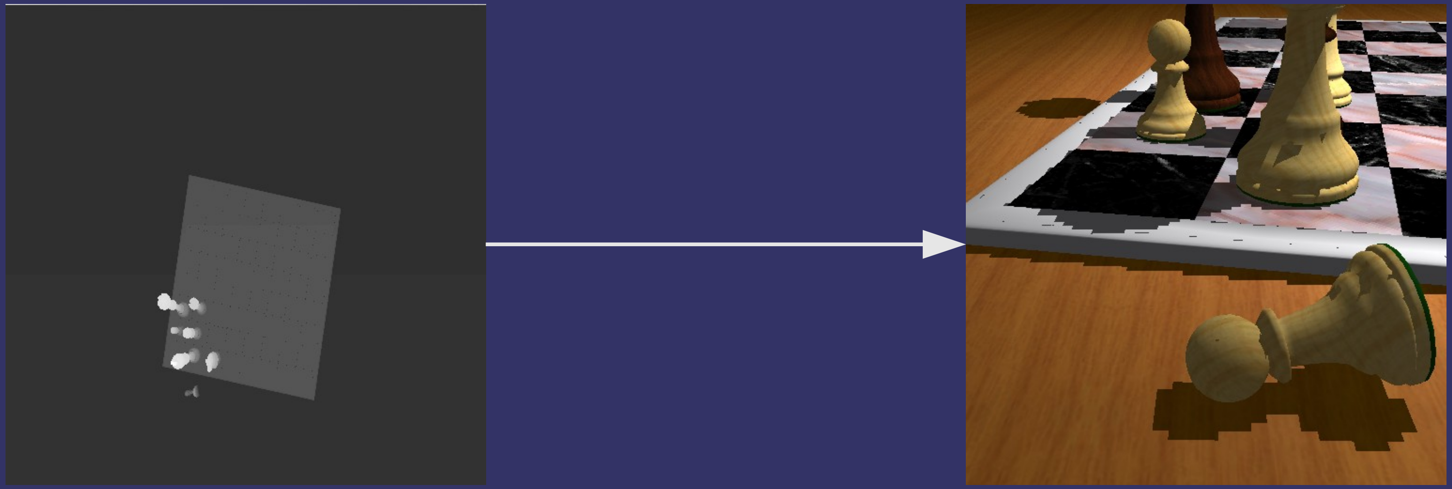
Perspective Shadow Maps

- Directional lights have other quirks
  - The more parallel the light and view direction, the lower the quality
    - A directional light pointing in the exact opposite direction of the view direction degrades back to the classic shadow map case
  - Casters behind the viewer (i.e., negative Z) are inverted and projected past the far plane
    - Several methods to handle this special case
  - Point lights have similar issues

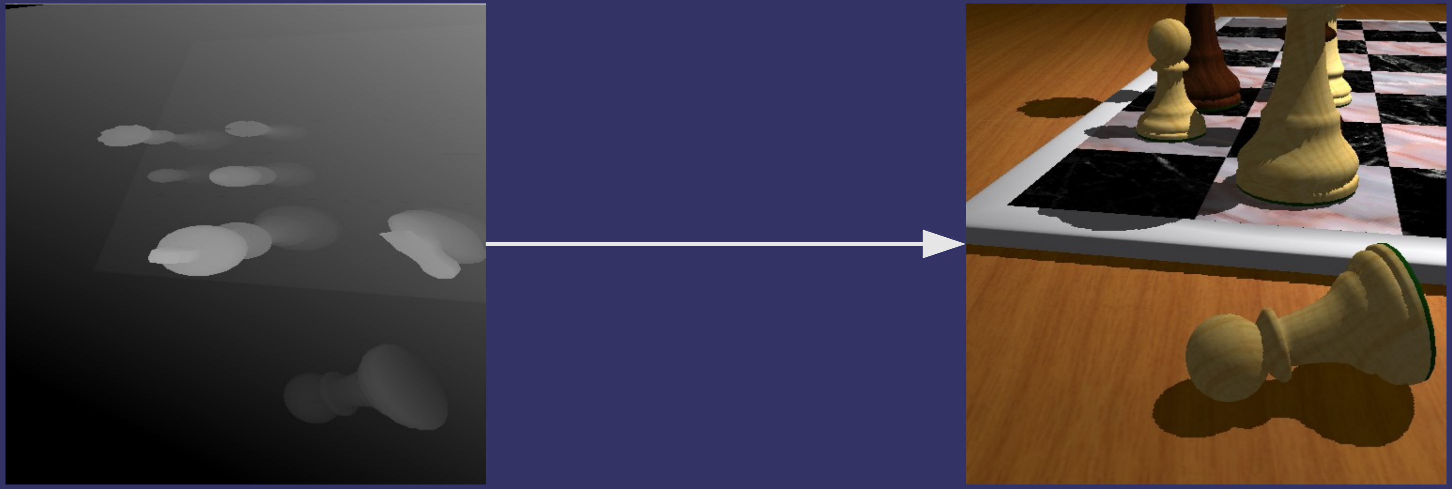
© Copyright Ian D. Romanick 2008, 2011 (cc) BY-NC-SA

Perspective Shadow Map

Standard shadow map



Perspective shadow map



Images from http://www-sop.inria.fr/reves/publications/data/2002/SD02/index.gb.html

© Copyright Ian D. Romanick 2008, 2011 (cc) BY-NC-SA

Perspective Shadow Maps

- Advantages:
  - Improves quality for many common cases
  - Easy to implement for directional light sources
- Disadvantages:
  - Shadow maps are view dependent, and must be regenerated when the camera moves (instead of just when the light or objects move)
  - Dual perspective transforms exaggerate shadow acne
  - As the viewer moves, the quality of the shadow map changes...even if the rest of the scene is static
    - For *most* games, this is the deal breaker

© Copyright Ian D. Romanick 2008, 2011 (cc) BY-NC-SA

References

Stamminger, M. and Drettakis, G. 2002. Perspective shadow maps. In *Proceedings of the 29th Annual Conference on Computer Graphics and interactive Techniques* (San Antonio, Texas, July 23 - 26, 2002). SIGGRAPH '02. ACM, New York, NY, 557-562. http://www-sop.inria.fr/reves/publications/data/2002/SD02/index.gb.html

© Copyright Ian D. Romanick 2008, 2011 (cc) BY-NC-SA

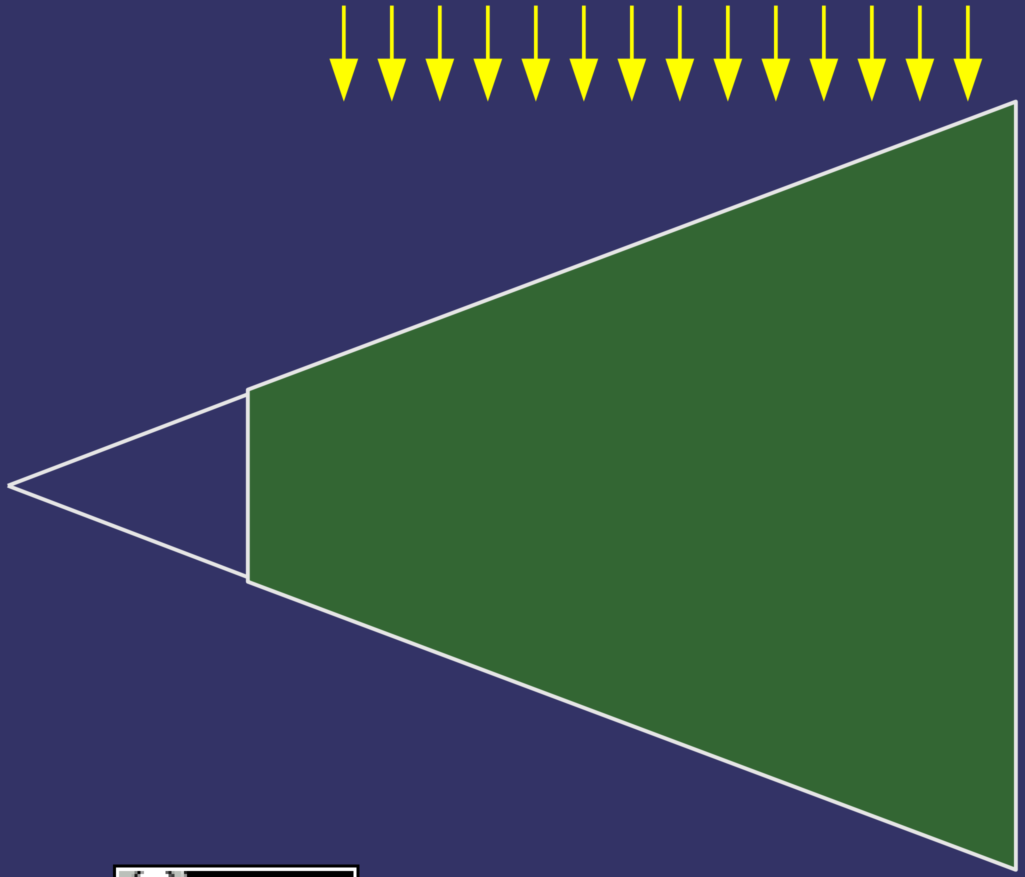
Perspective Shadow Maps

- Some significant problems:
  - Shadow map *quality* is view-dependent
  - Several special cases that must be handled depending on light direction / position
  - Difficulties handling shadow casters behind the camera
- Introduced some good ideas:
  - Re-parameterizing the scene based on the camera / light frusta
  - Quantitatively determining when aliasing will occur

© Copyright Ian D. Romanick 2008, 2011 (cc) BY-NC-SA

Parallel-Split Shadow Maps

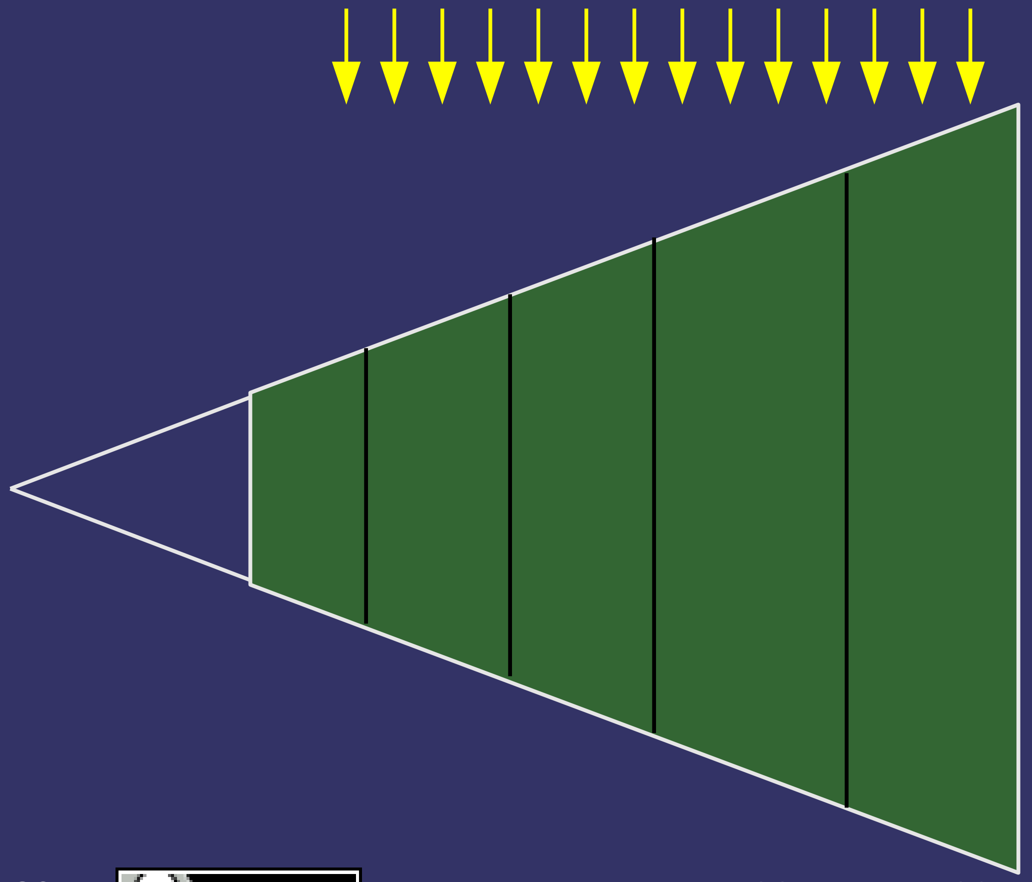
- PSSMs solve most of these problems



© Copyright Ian D. Romanick 2008, 2011 BY-NC-SA

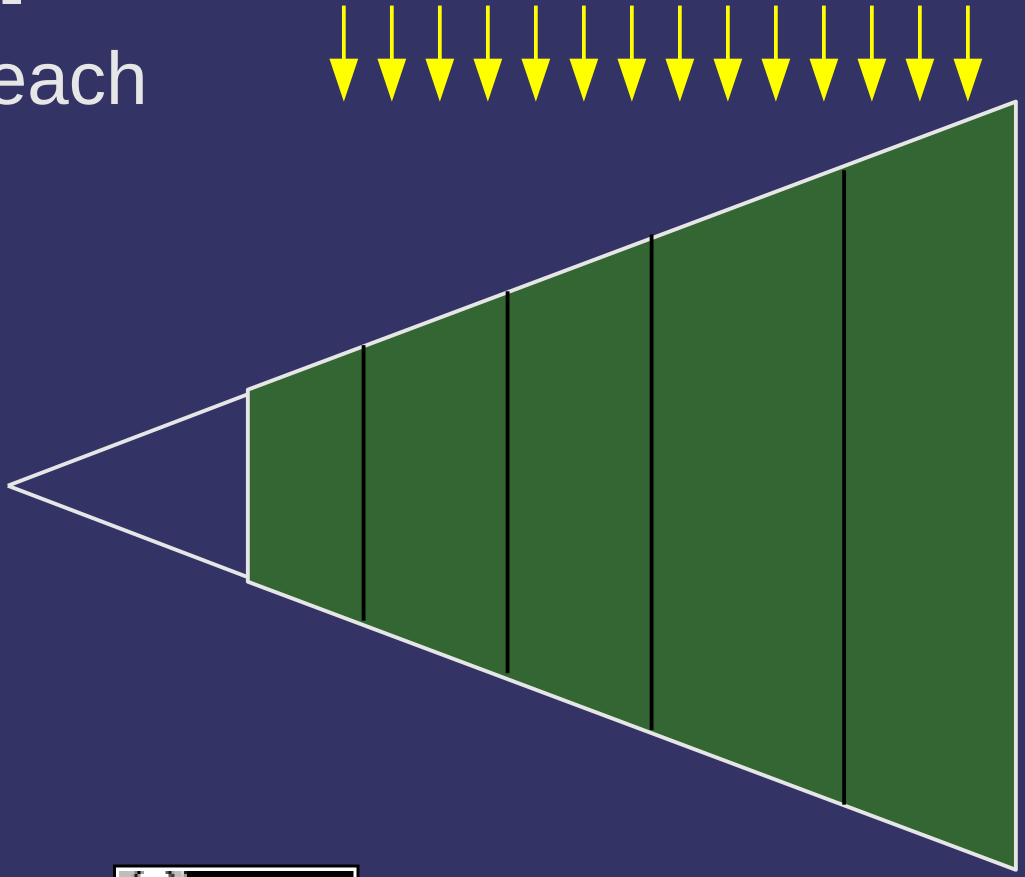
Parallel-Split Shadow Maps

- PSSMs solve most of these problems
  - Split view frustum into m parts with planes parallel to the near / far plane



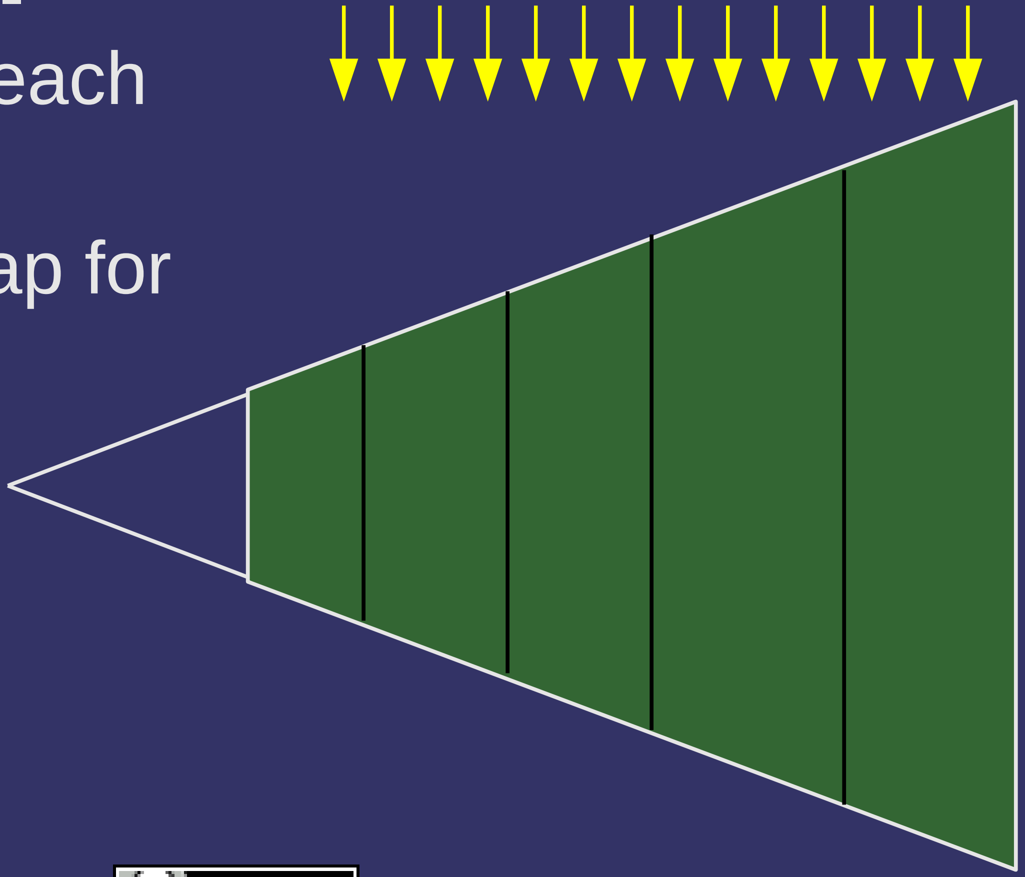
Parallel-Split Shadow Maps

- PSSMs solve most of these problems
  - Split view frustum into m parts with planes parallel to the near / far plane
  - Calculate light's view-projection matrix for each split region



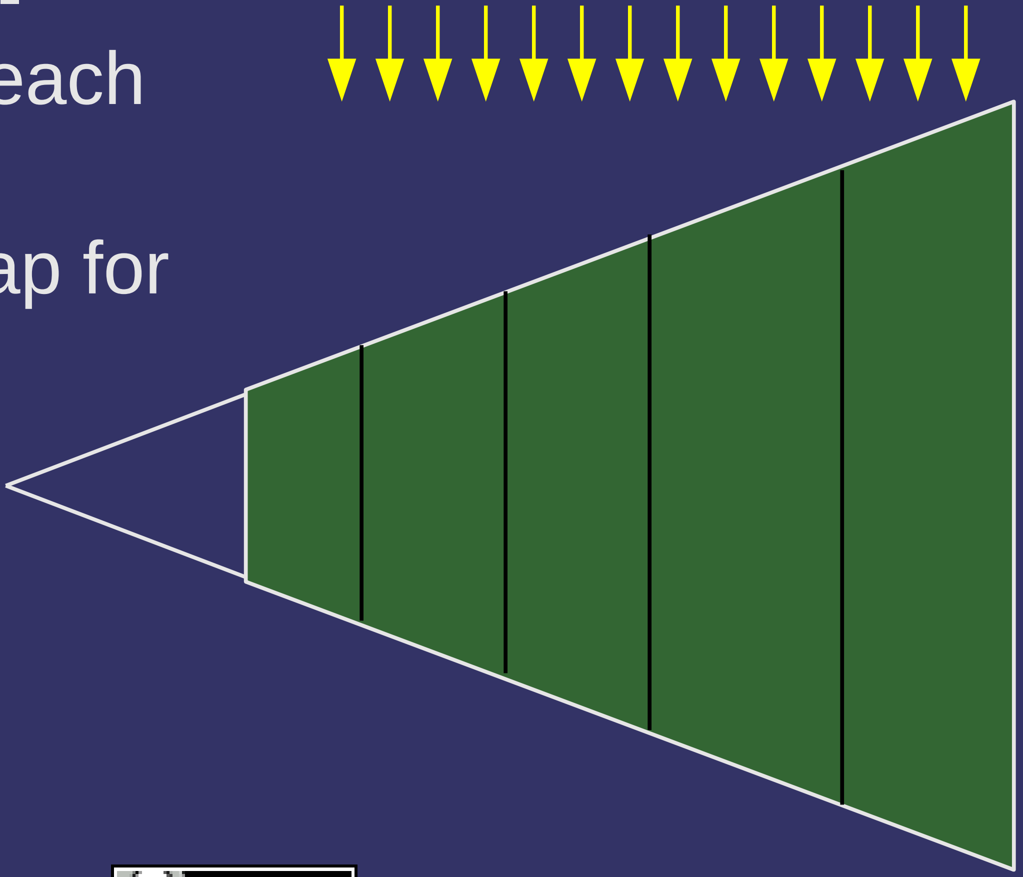
Parallel-Split Shadow Maps

- PSSMs solve most of these problems
  - Split view frustum into m parts with planes parallel to the near / far plane
  - Calculate light's view-projection matrix for each split region
  - Generate shadow map for each split region



Parallel-Split Shadow Maps

- PSSMs solve most of these problems
  - Split view frustum into m parts with planes parallel to the near / far plane
  - Calculate light's view-projection matrix for each split region
  - Generate shadow map for each split region
  - Apply shadow maps to scene



© Copyright Ian D. Romanick 2008, 2011 BY-NC-SA

Parallel-Split Shadow Maps

- Aliasing occurs when $d > d_i$

$$d = d_s \frac{r_s}{r_i} \frac{N \cdot V}{N \cdot L}$$

  - Rename r_i as z, and call dz the change in z relative to one unit in ds

$$d = \frac{dz}{z\,ds} \frac{N \cdot V}{N \cdot L}$$

  - Ignoring perspective aliasing, this means that we want dz / zds to be constant over the entire view
    - Call this constant ρ

© Copyright Ian D. Romanick 2008, 2011 BY-NC-SA

Parallel-Split Shadow Maps

- Optimal shadow map distribution is:

$$\frac{ds}{z\,dz}=\rho\Rightarrow s(z)=\int_0^s ds=\frac{1}{\rho}\int_n^z\frac{1}{z}dz=\frac{1}{\rho}\ln\left(\frac{z}{n}\right)$$

  - Since $s(f)$ = 1, $\rho = \ln(f / n)$

© Copyright Ian D. Romanick 2008, 2011 (cc) BY-NC-SA

Parallel-Split Shadow Maps

- Current hardware can't do this non-linear z transform
  - Discretely perform the mapping in steps at the split planes

$$s_i = \mathrm{s}(C_i^{\log}) = \frac{1}{\ln(f/n)} \ln\left(\frac{C_i^{\log}}{n}\right)$$

  - Each split gets 1 / m of total texture resolution, substituting i / m for s_i

$$C_i^{\log} = n\left(\frac{f}{n}\right)^{i/m}$$

© Copyright Ian D. Romanick 2008, 2011 (cc) BY-NC-SA

Parallel-Split Shadow Maps

- Alternately, the view frustum could be divided into equally sized pieces

$$C_i^{uni} = \frac{(f - n) \times i}{m} + n$$

© Copyright Ian D. Romanick 2008, 2011 (cc) BY-NC-SA

Parallel-Split Shadow Maps

- Neither split strategy works very well
  - Logarithmic splitting groups split-planes too close to the near plane
  - Uniform splitting doesn't group split-planes close enough to the near plane

© Copyright Ian D. Romanick 2008, 2011 (cc) BY-NC-SA

Parallel-Split Shadow Maps

- Neither split strategy works very well
  - Logarithmic splitting groups split-planes too close to the near plane
  - Uniform splitting doesn't group split-planes close enough to the near plane
- Instead, use a hybrid of the two

$$C_i = \lambda C_i^{\log} + (1 - \lambda) C_i^{uni}$$

  - λ is tunable parameter
  - The paper calls this the *practical split scheme*

© Copyright Ian D. Romanick 2008, 2011 (cc) BY-NC-SA

Parallel-Split Shadow Maps

- Light transformation matrices are determined much like before
  - Calculate view-projection matrix for light relative to whole view frustum
  - Transform each split region to light's post-projection space
  - Calculate AABB for transformed split region
  - Use AABB to calculate “crop” transformation to scale and center split region to full view

© Copyright Ian D. Romanick 2008, 2011 (cc) BY-NC-SA

Parallel-Split Shadow Maps

- To apply shadows, the shader must determine which region contains the current fragment
  - Determine the split-plane, C_s, nearest the camera but farther away than the current fragment
  - C_s determines which shadow map to apply
  - The light transforms, C_i distances, and shadow maps (samplers) must be provided to the shader as arrays of uniforms
    - m is a compile-time constant

© Copyright Ian D. Romanick 2008, 2011 (cc) BY-NC-SA

Parallel-Split Shadow Maps

- Only directional lights have been dealt with so far

© Copyright Ian D. Romanick 2008, 2011 BY-NC-SA

Parallel-Split Shadow Maps

- Only directional lights have been dealt with so far
  - Light transformations for each split region are calculated from the light's post-projection space

© Copyright Ian D. Romanick 2008, 2011 BY-NC-SA

Parallel-Split Shadow Maps

- Only directional lights have been dealt with so far
  - Light transformations for each split region are calculated from the light's post-projection space
  - For point lights, transform by the light's view-projection matrix *first*
    - This effectively converts the point-light to a directional light!

© Copyright Ian D. Romanick 2008, 2011 (cc) BY-NC-SA

References

Zhang, F., Sun, H., Nyman, O. “Parallel-Split Shadow Maps on Programmable GPUs,” in *GPU Gems 3,* ed. Hubert Nguyen, pp. 202 – 237. Boston, MA: Addison-Wesley, 2008. http://appsrv.cse.cuhk.edu.hk/~fzhang/pssm_project/

Wimmer, M., Scherzer, D., and Purgathofer, W. “Light Space Perspective Shadow Maps,” in *Proceedings of Eurographics Symposium on Rendering*, pp. 143 - 151. Norrköping, Sweden: Eurographics Association, 2004. http://www.cg.tuwien.ac.at/research/vr/lispsm/

© Copyright Ian D. Romanick 2008, 2011 (cc) BY-NC-SA

Next week...

- More shadow maps
  - Optimal light viewing frustum calculations
  - Resolution matched shadow maps
  - Omni-directional lights
  - Texture atlases for shadow maps
  - Read:

Brabec, Stefan and Annen, Thomas and Seidel, Hans-Peter, "Shadow Mapping for Hemispherical and Omnidirectional Light Sources." In *Advances in Modeling, Animation and Rendering* (Proceedings Computer Graphics International 2002) , pages 397-408. Springer, 2002. http://www.mpi-inf.mpg.de/~brabec/

Aaron E. Lefohn and Shubhabrata Sengupta and John D. Owens, "Resolution Matched Shadow Maps." ACM Transactions on Graphics , vol. 26 , no. 4 , pages 20:1--20:17. ACM, 2007. http://www.idav.ucdavis.edu/publications/print_pub?pub_id=919

© Copyright Ian D. Romanick 2008, 2011 (cc) BY-NC-SA

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.

This work is licensed under the Creative Commons Attribution-NonCommercial-ShareAlike 3.0 United States License. To view a copy of this license, visit http://creativecommons.org/licenses/by-nc-sa/3.0/us/ or send a letter to Creative Commons, 171 Second Street, Suite 300, San Francisco, California, 94105, USA.