Improved Hard shadows

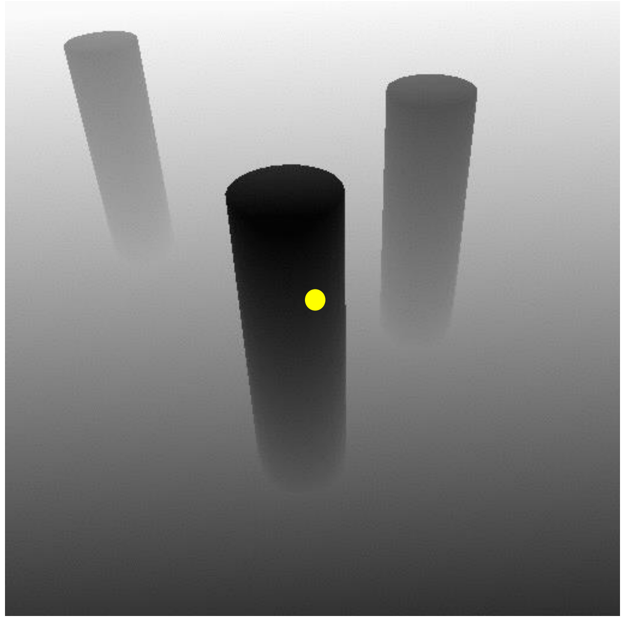
Pere-Pau Vázquez
ViRVIG – UPC

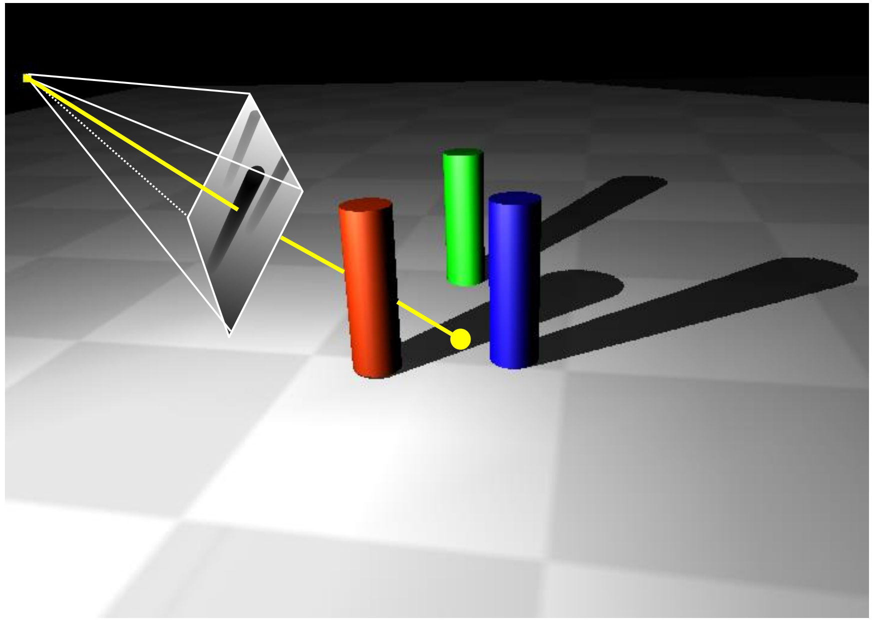
Outline

- Motivation
- Practical Shadow Mapping
- Perspective Shadow Maps
- Parallel-split Shadow Maps

Motivation

- Image-based method
  - Compute shadow map from light view
  - Compute final image
    - compare actual vs. stored depth



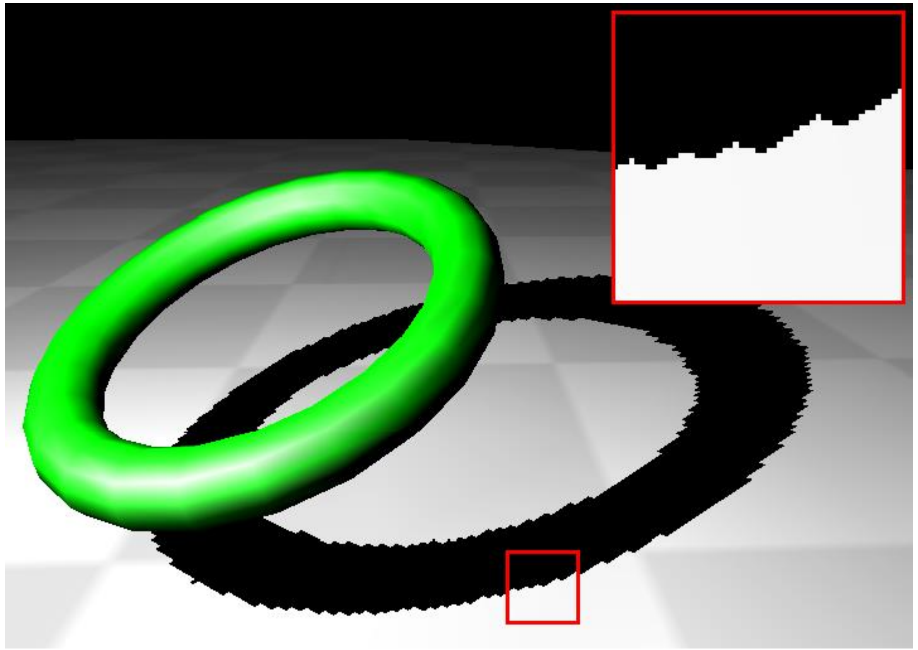


Motivation

- Well suited for hardware-accelerated rendering [Segal92]
  - Like texture mapping but with depth values
    - Depth textures, texture compare mode
  - First realized in SGI's Infinite Reality
    - Now available on all consumer-cards

Motivation

- But: shadow quality may suffer from sampling artifacts
  - More on this later…



Blocky shadow edges due to limited shadow map resolution

Motivation

- Consumer 3D hardware solution
  - Shader-based:
    - Compare with depth map using tex2DShadow or similar
      - Returns 1 if lit and 0 if not lit
    - Completely supported in graphics hardware

Motivation. Issues with shadow maps

- Prone to aliasing artifacts
  - "Percentage closer" filtering helps this
  - Normal color filtering does not work well
- Depth bias is not completely foolproof
- Requires extra shadow map rendering pass and texture loading
- Higher resolution shadow map reduces blockyness
  - But also increase texture copying expense

Motivation. Issues with shadow maps

- Shadows are limited to view frustums
  - Could use six view frustums for omni-directional light
- Objects outside or crossing the near and far clip planes are not properly accounted for by shadowing
  - Move near plane in as close as possible
  - But too close throws away valuable depth map precision when using a projective frustum

Motivation. Issues with shadow maps

- Requires knowing how pixels (samples) in the light's view compare to the size of pixels (samples) in the eye's view
  - A re-sampling problem

Motivation. Issues with shadow maps

- When light source frustum is reasonably well aligned with the eye's view frustum, the ratio of sample sizes is close to 1.0
  - Great match if eye and light frustum's are nearly identical
  - But that implies very few viewable shadows
  - Consider a miner's lamp (i.e., a light attached to your helmet)
  - The chief reason for such a lamp is you don't see shadows from the lamp while wearing it

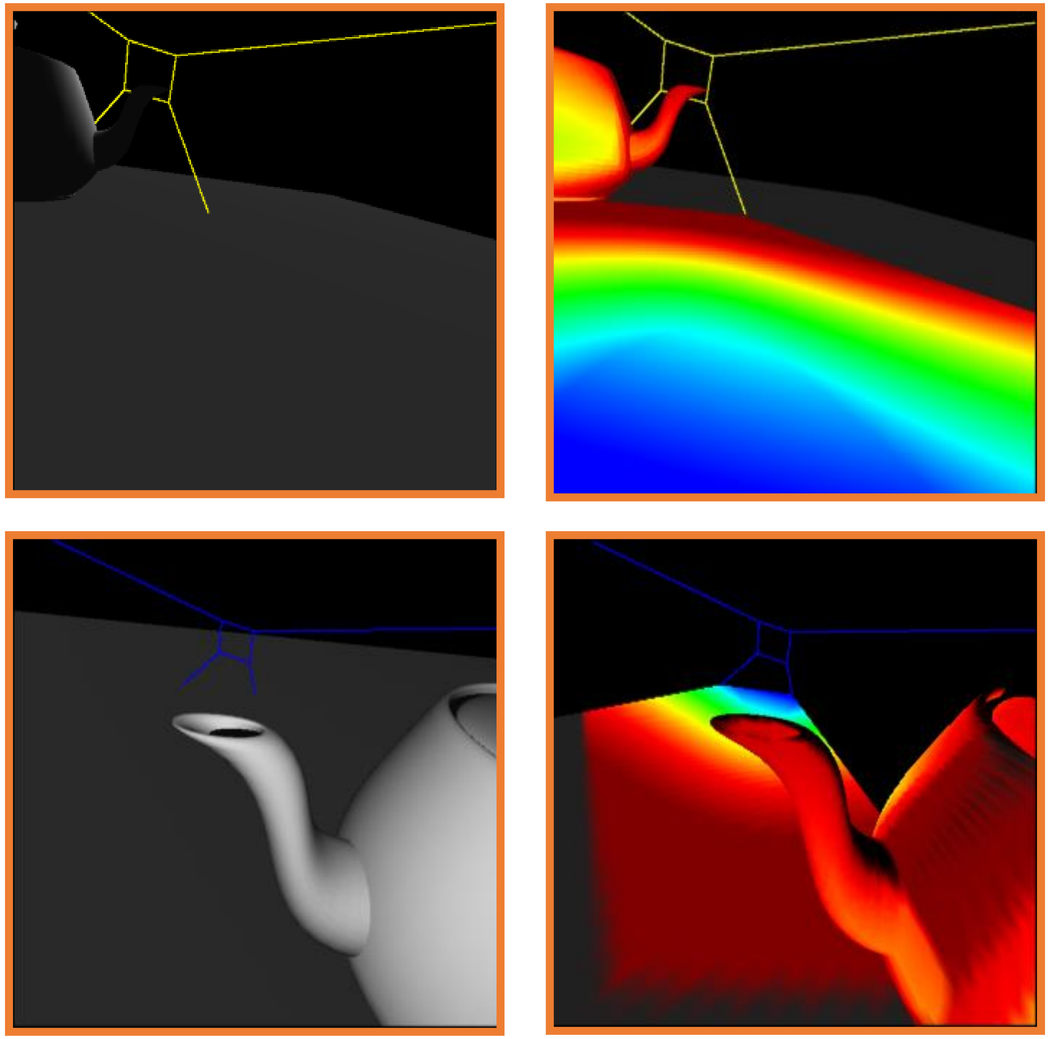
Motivation. Issues with shadow maps

- So best case is miner's lamp
- Worst case is shadows from light shining at the viewer
  - "that deer in the headlights" problem – definitely worst case for the deer
  - Also known as the "dueling frusta" problem (frusta, plural of frustum)
- Let's attempt to visualize what happens…

Motivation. Issues with shadow maps

Eye's View

Light's View

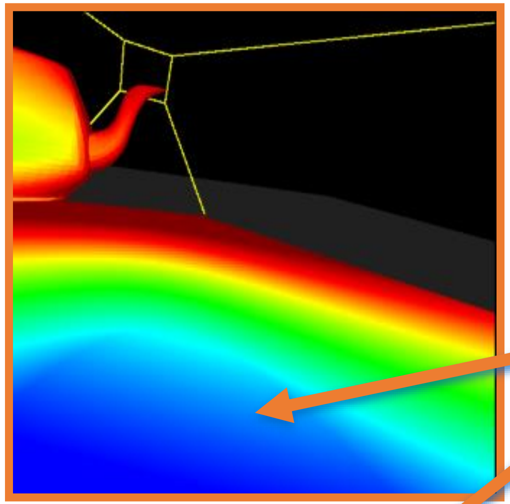


Eye's View with projection of color-coded mipmap levels from light:
Blue = magnification
Red = minification

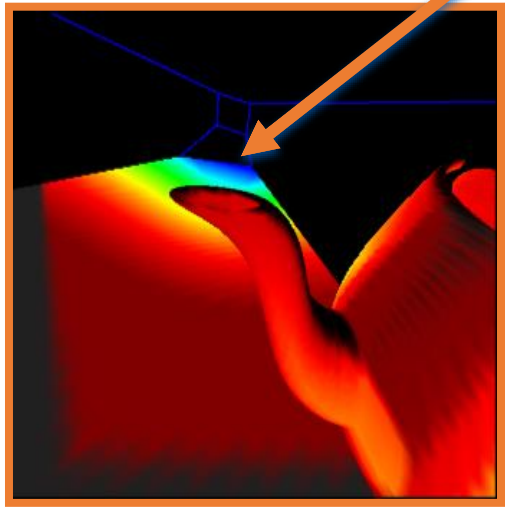
Light's View with re-projection of above image from the eye

Motivation. Issues with shadow maps

Eye's View



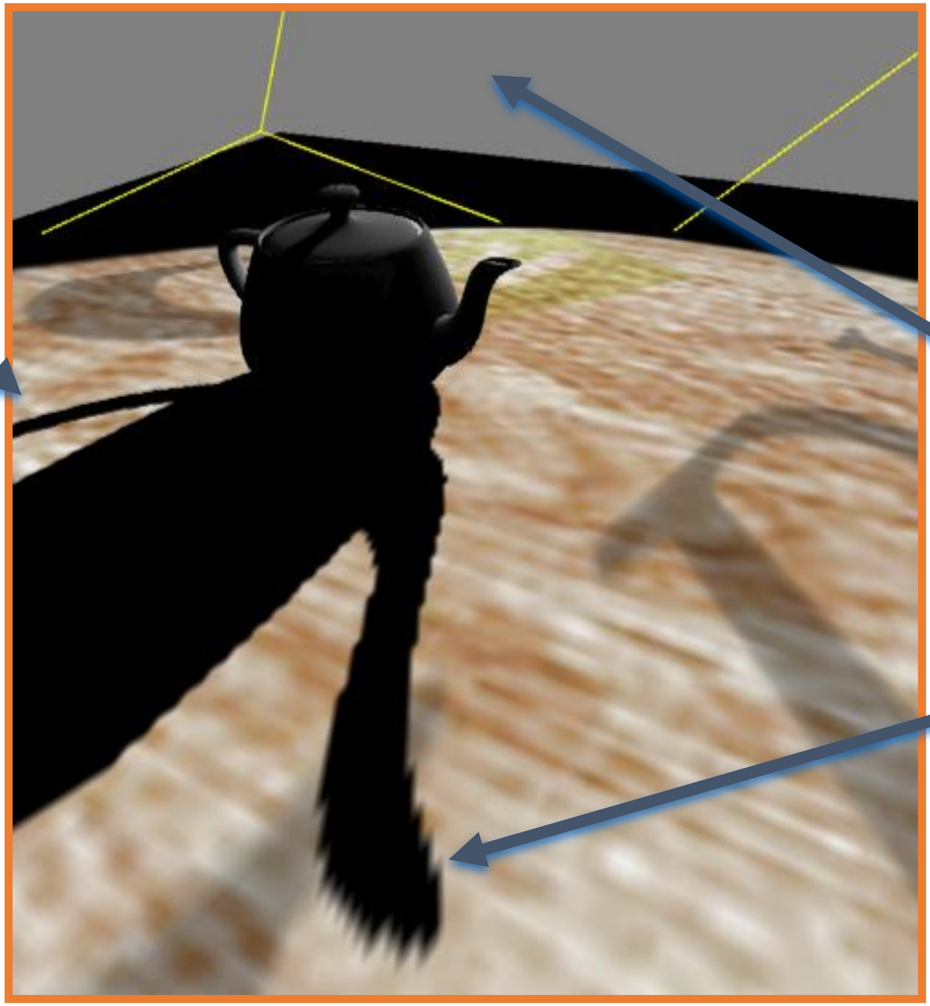
Light's View



Region that is smallest in the light's view is a region that is very large in the eye's view. This implies that it would require a very high-resolution shadow map to avoid obvious blocky shadow edge artifacts.

Motivation. Issues with shadow maps

Notice that shadow edge is well defined in the distance.



Light position out here pointing towards the viewer.

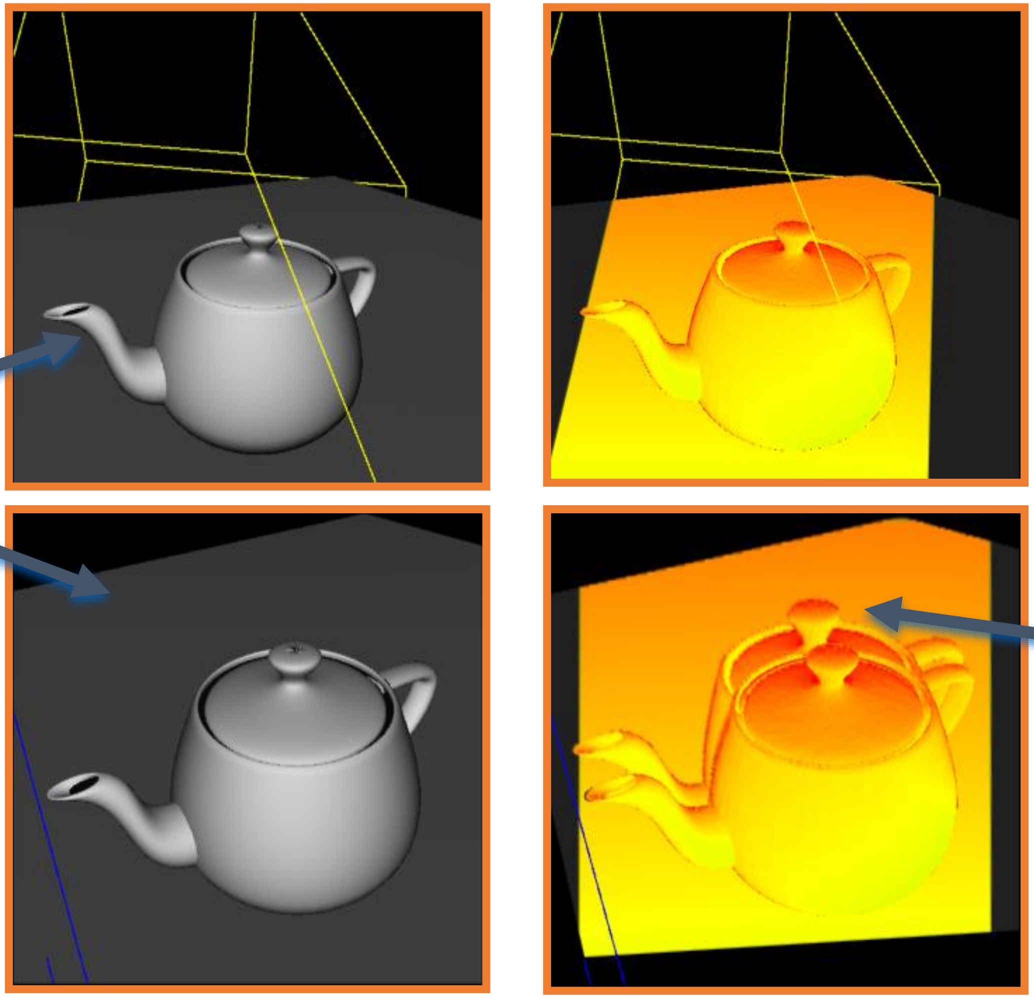
Blocky shadow edge artifacts.

Motivation. Issues with shadow maps

Eye's View

Very similar views

Light's View



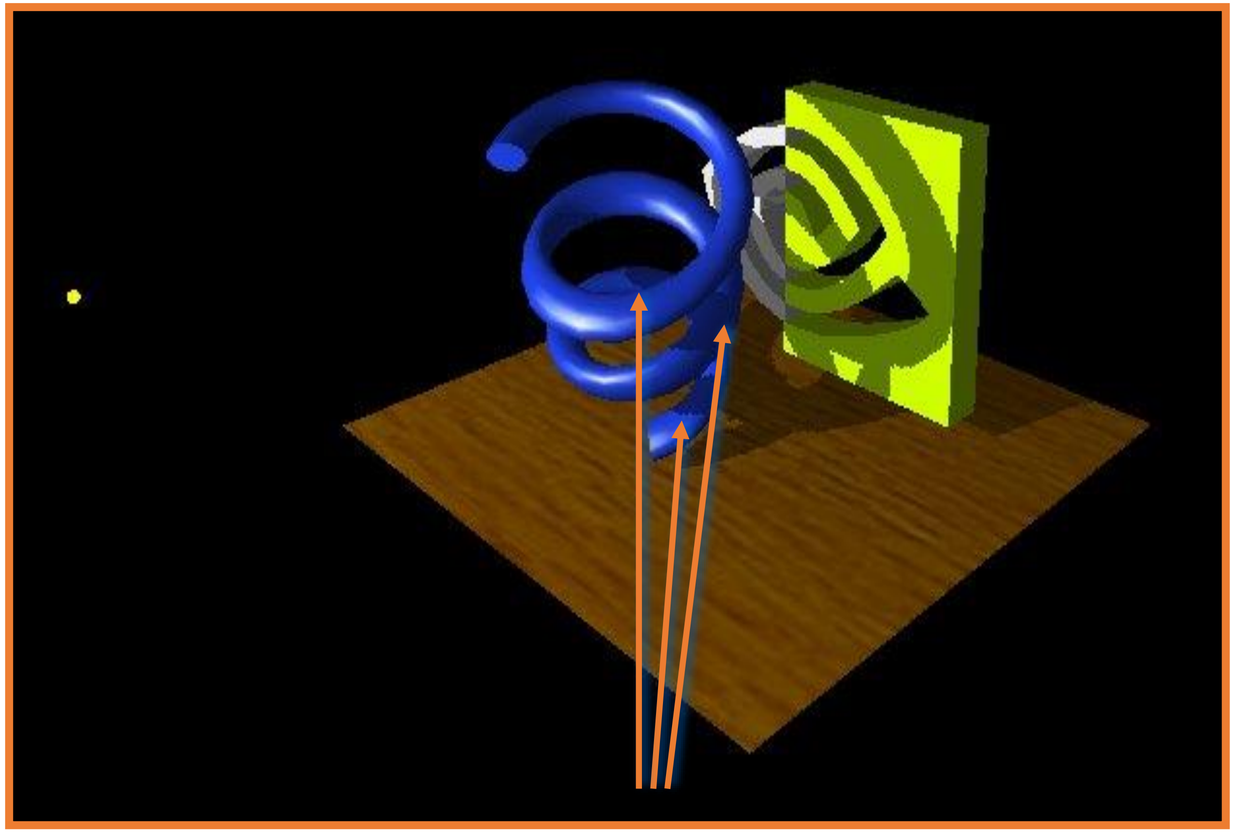
Closer to the miner's lamp example

Note how the color-coded images share similar pattern and the coloration is uniform. Implies single depth map resolution would work well for most of the scene.

Ghosting is where projection would be in shadow.

Motivation. Issues with shadow maps

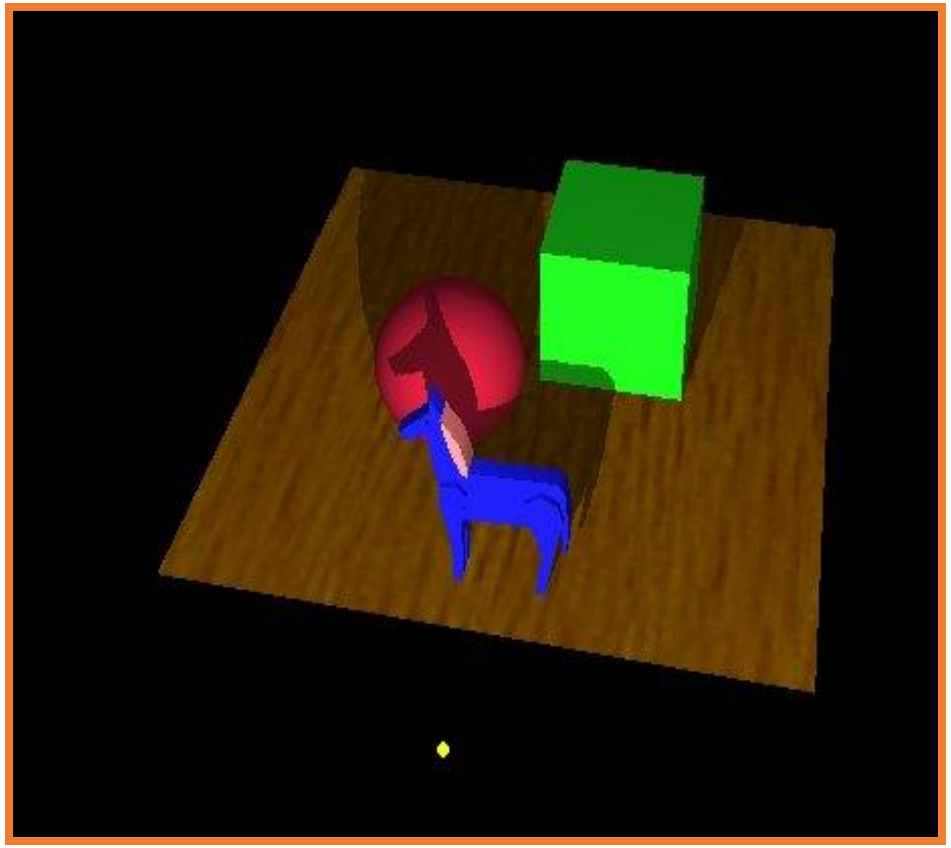
- Smooth surfaces with object self-shadowing



Note object self-shadowing

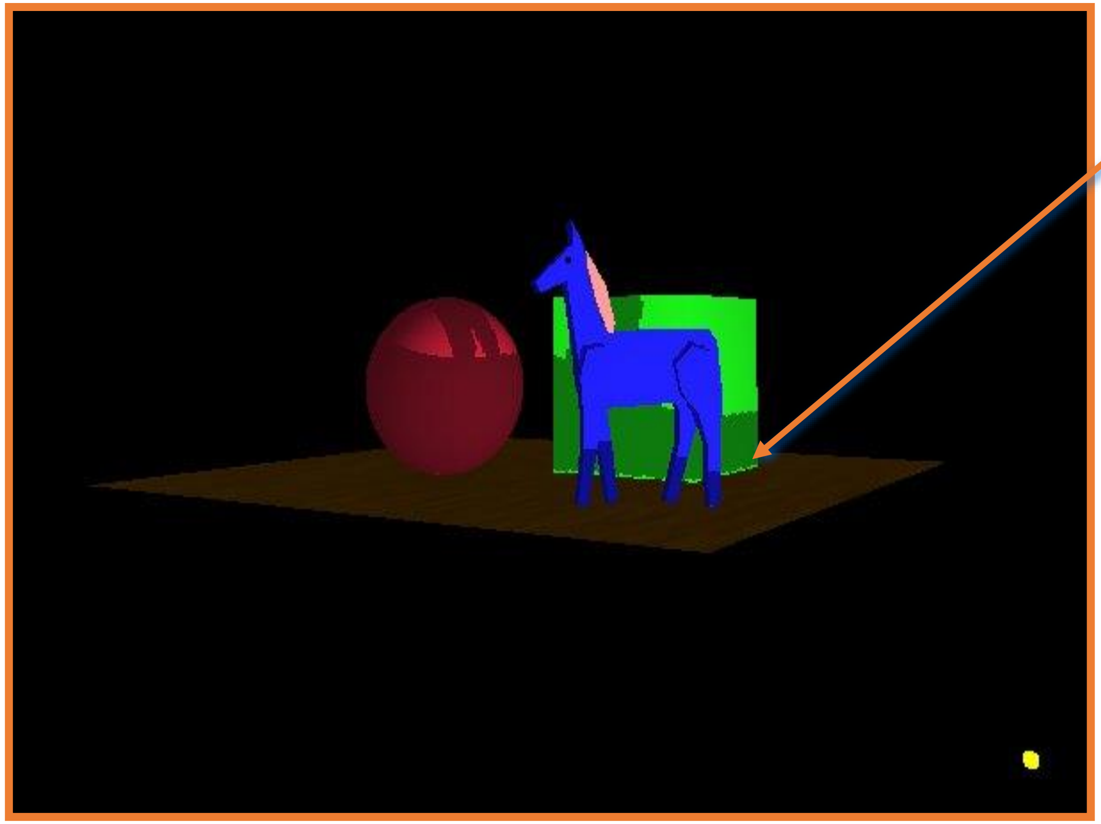
Motivation. Issues with shadow maps

- Complex objects all shadow



Motivation. Issues with shadow maps

- Even the floor casts shadow



Note shadow leakage due to infinitely thin floor

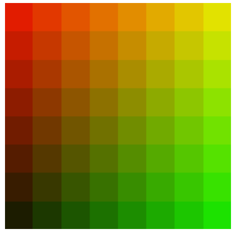
Could be fixed by giving floor thickness

Outline

- *Motivation*
- Practical Shadow Mapping
- Perspective Shadow Maps
- Parallel-split Shadow Maps

Practical Shadow Mapping

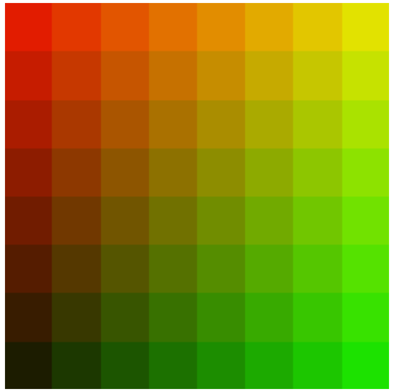
- Render scene with projected texture
  - Control texture encodes cells in shadow map



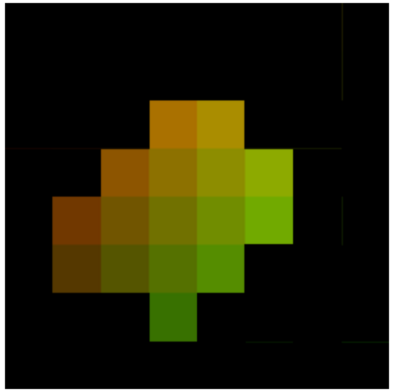
- Read back image
  - Find those cells which are actually used

Practical Shadow Mapping

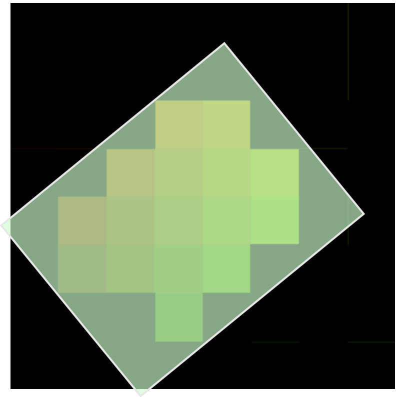
- Fit light frustum to visible cells



map regions



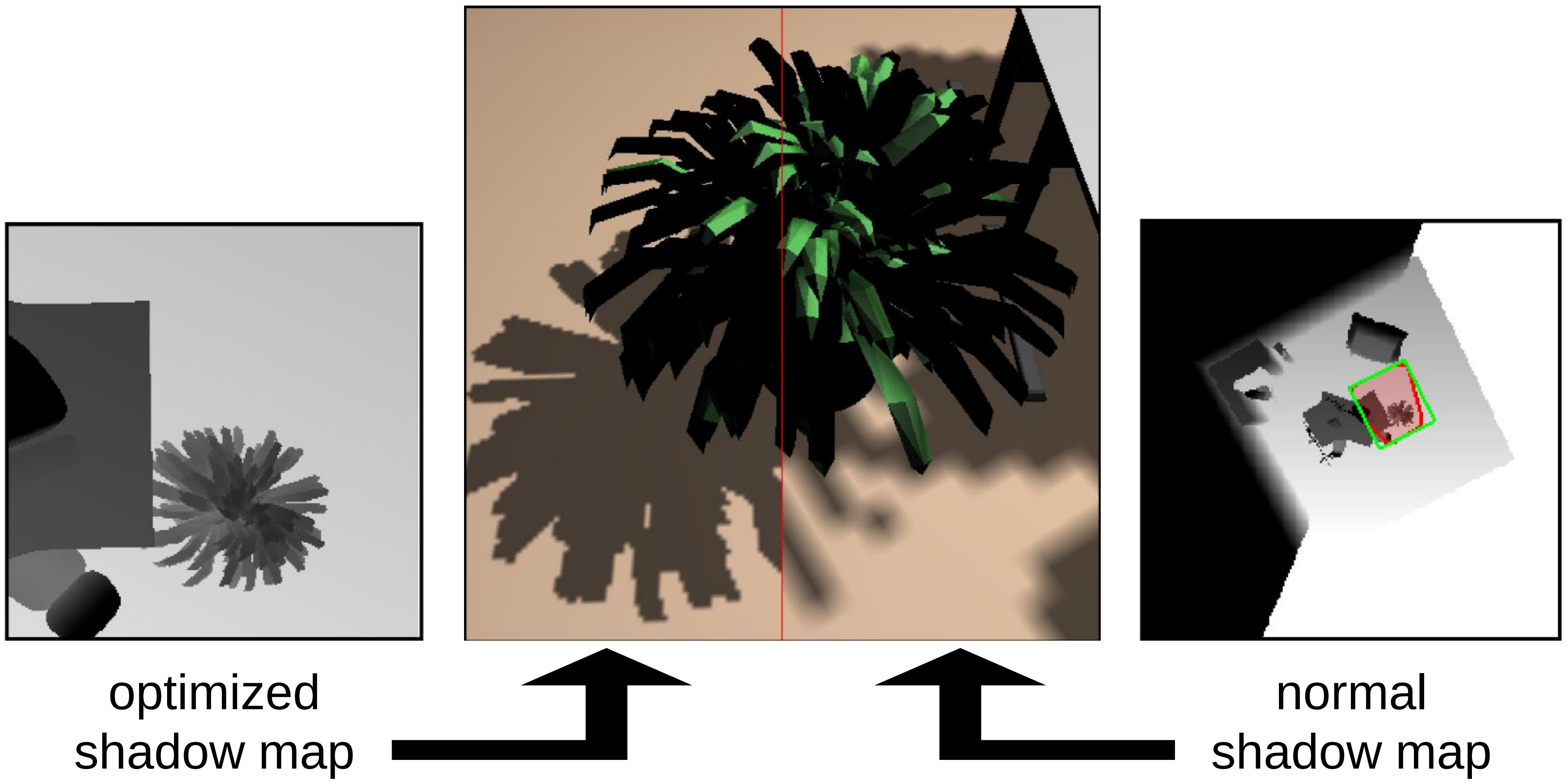
used regions



new map frustum

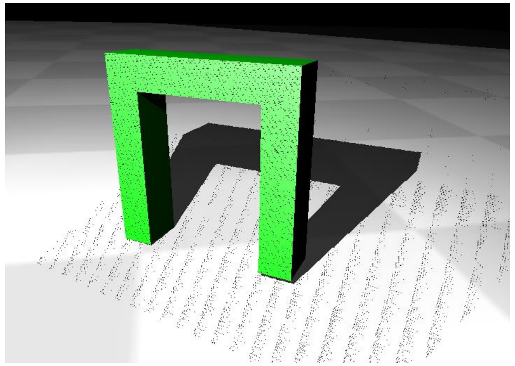
  - Axis-aligned bounding box
  - Minimum-area bounding rectangle
- Generate shadow map using the optimized frustum

Practical Shadow Mapping

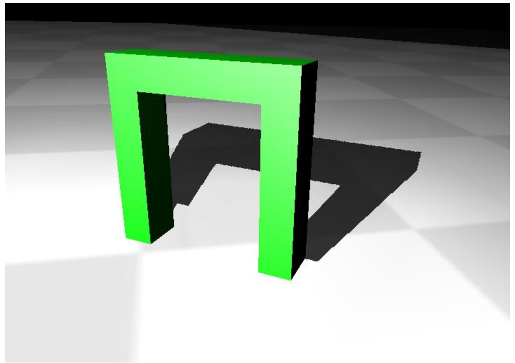


Practical Shadow Mapping

- Control texture can also be used to improve depth sampling
- Reduces artifacts due to limited numerical precision



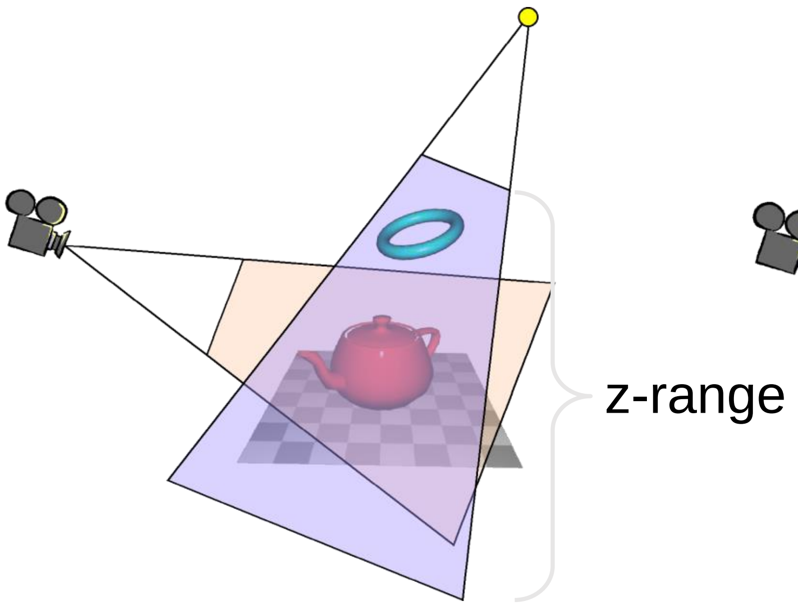
Self shadowing



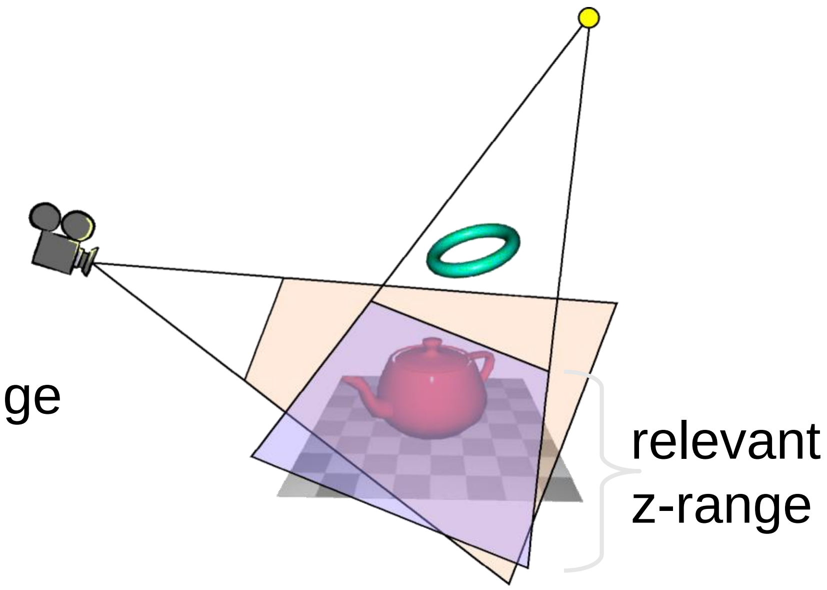
Missing shadows

Practical Shadow Mapping

- Adjust z-range (near/far) for visible objects



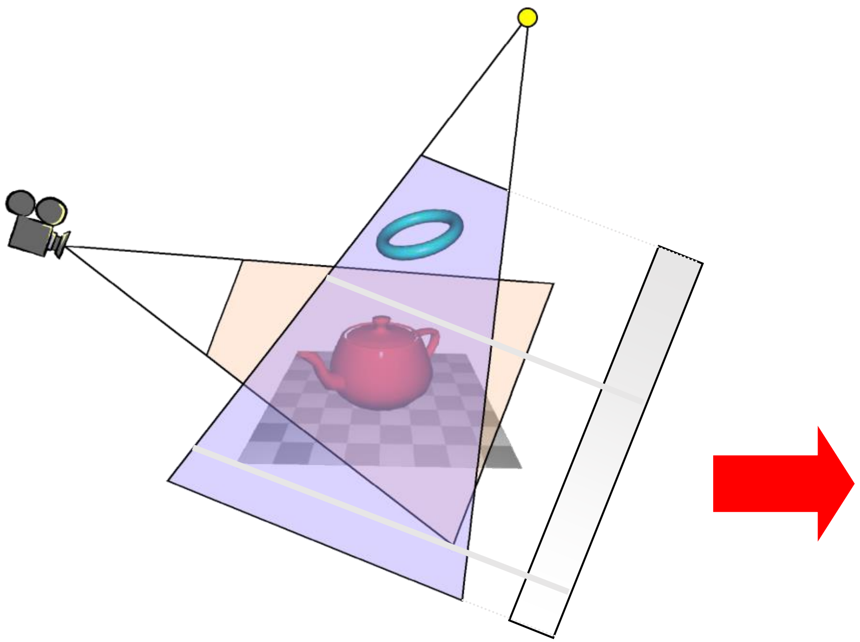
Normal approach:
Near & far plane encloses
all objects !

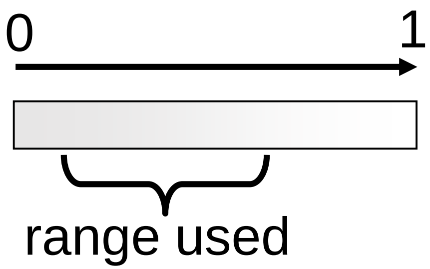


Better:
Near & far plane encloses
all from camera visible objects !

Practical Shadow Mapping

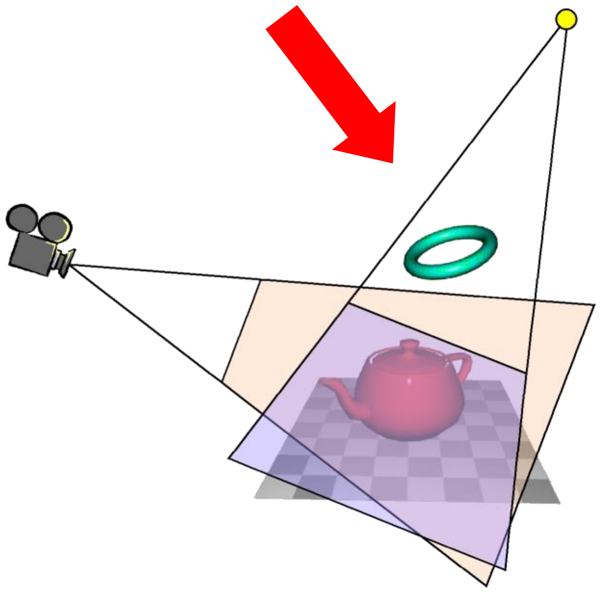
- Use 1D control texture that maps light distance to color (z-ramp)
  - Detect relevant z-range e.g. [0.42;0.72]





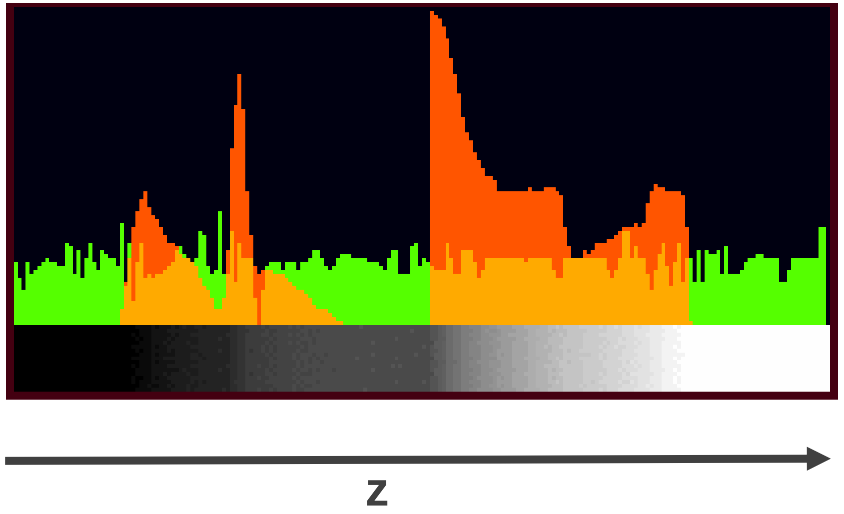
Practical Shadow Mapping

- Problem:
  - Objects in front of near plane are shadow casters !
- Solution:
  - Clamp depth values
    - One bit for objects in front enough (blocked: yes/no)
  - Possible on many cards using depth replace operation
  - Direct support on GPUs



Practical Shadow Mapping

- Can even go further and optimize depth distribution *between* near and far plane
  - Histogram equalization
  - Improves *depth contrast*



Red:
original distribution
Green:
normalized distribution

Practical Shadow Mapping

- Summary
  - Various ways to optimize light source's viewing frustum
  - Better shadow quality
  - Most steps are hardware-accelerated
    - Read-back of control image is main bottleneck
- Extensions
  - Full hardware implementation
  - Combine with other methods
    - Perspective Shadow Maps

Outline

- *Motivation*
- *Practical Shadow Mapping*
- Perspective Shadow Maps
- Parallel-split Shadow Maps

Perspective Shadow Maps

- Marc Stamminger , George Drettakis, Perspective shadow maps, Proceedings of the 29th annual conference on Computer graphics and interactive techniques, July 23-26, 2002, San Antonio, Texas

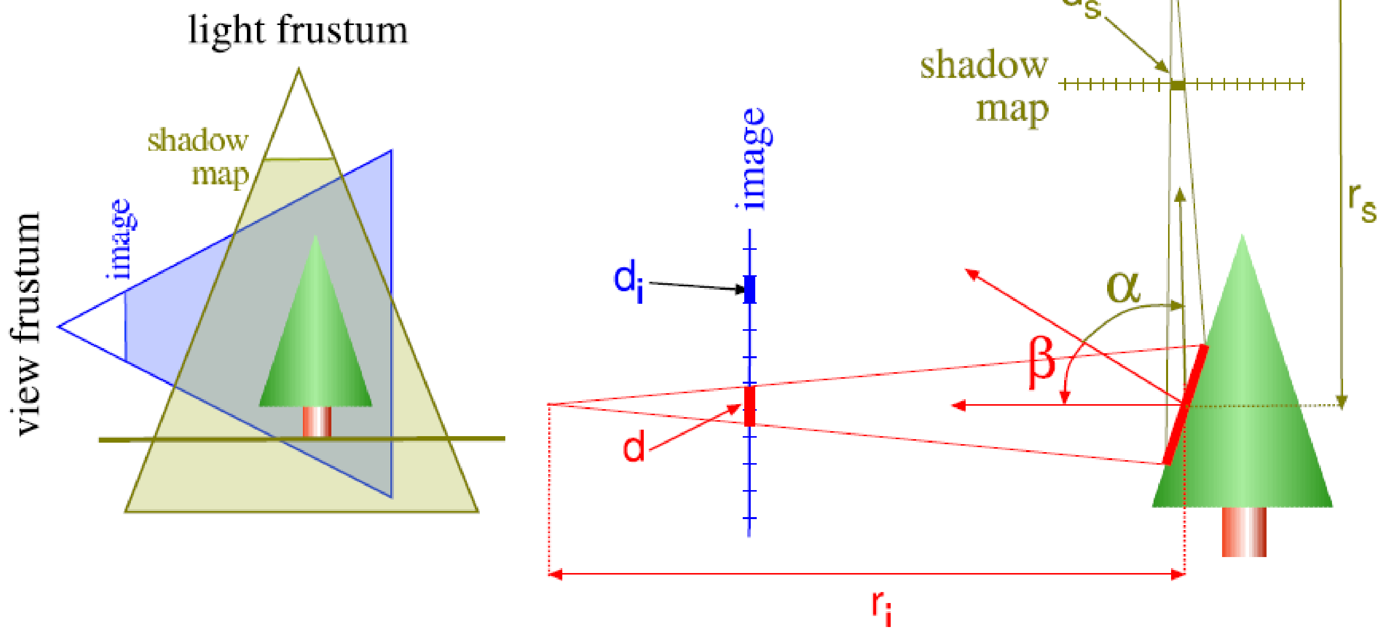
Perspective Shadow Maps

- Main Contribution
  - In this paper they introduced perspective shadow maps, which are generated in normalized device coordinate space. This results in important reduction of shadow map aliasing with almost no overhead.

Perspective Shadow Maps

- Shadow Map Aliasing

$$d = d_s \frac{r_s}{r_i} \frac{\cos\beta}{\cos\alpha}$$



Perspective Shadow Maps

- Shadow Map Aliasing

$$d = \boxed{d_s \frac{r_s}{r_i}} \frac{\cos\beta}{\cos\alpha}$$

  - Perspective Aliasing
    - Undersampling appears when d is larger than the image pixel size d_i.
    - Can be avoided by keeping the fraction r_s/r_i close to a constant.
    - Due to limited memory, the shadow map resolution can only be increased up to a certain limit in practice.

Perspective Shadow Maps

- Shadow Map Aliasing

$$d = d_s \frac{r_s}{r_i} \boxed{\frac{\cos\beta}{\cos\alpha}}$$

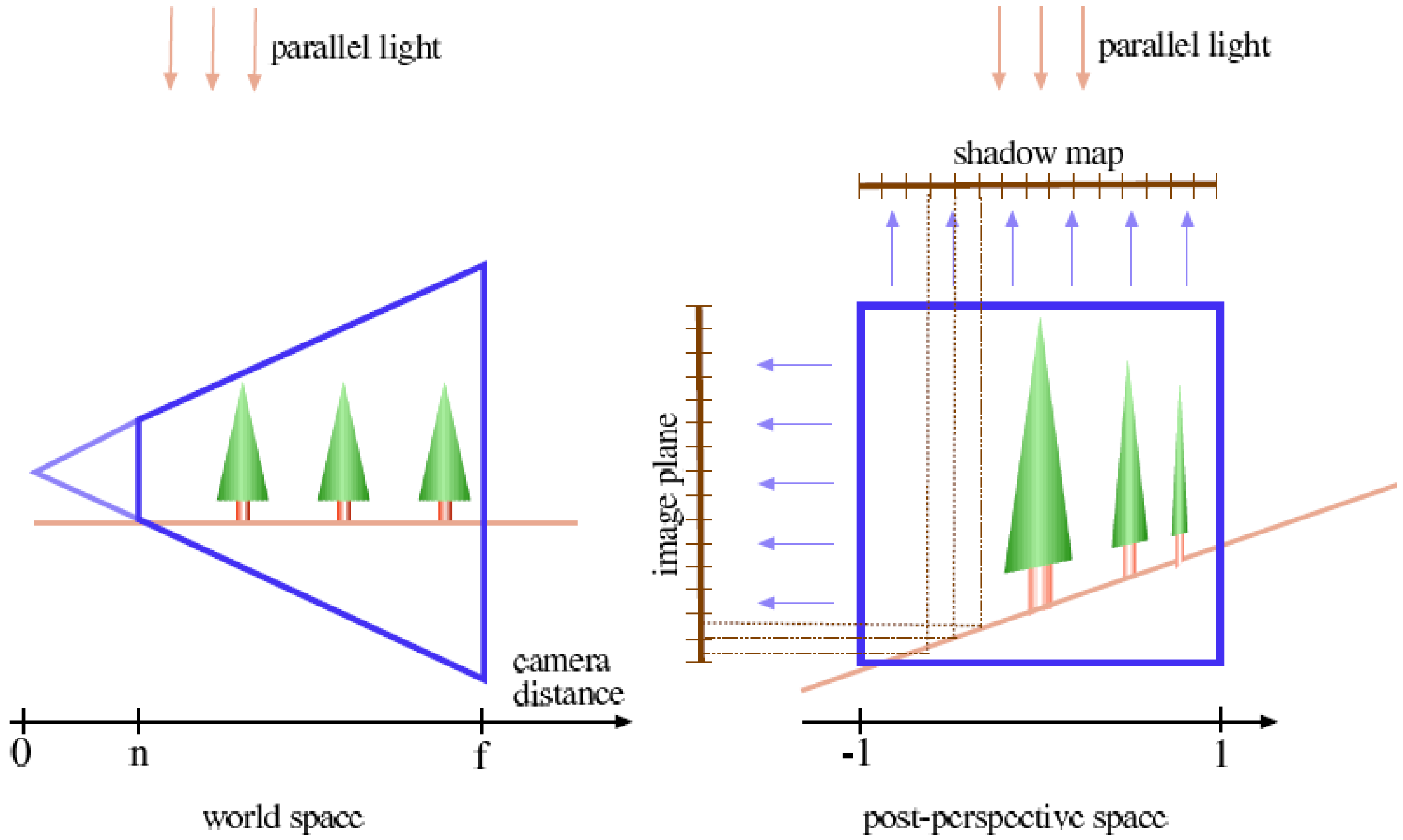
  - Projection Aliasing
    - Appears when cos β/cos α is large
    - Typically happens when the light rays are almost parallel to a surface, so that the shadow stretches along the surface.
    - Require a local increase in shadow map resolution. Not dealing with it…

Perspective Shadow Maps

- Main Idea
  - Try to keeping the fraction r_s/r_i close to a constant.
  - How to do that?
    - First map the scene to post-perspective space
    - Generate a standard shadow map in this space by rendering a view from the transformed light source.
    - We can work in post-perspective space almost like in world space, with the exception of objects behind the viewer

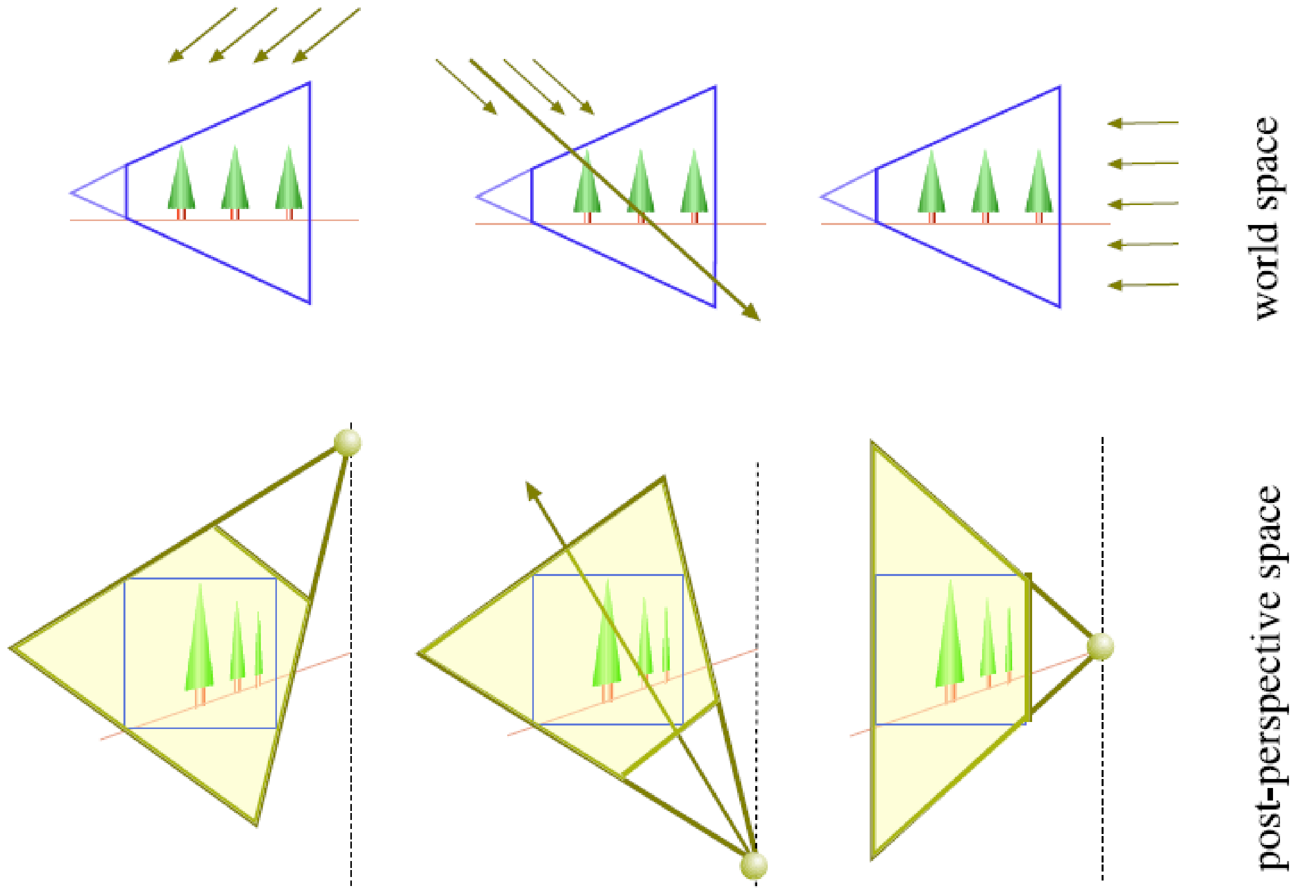
Perspective Shadow Maps

- Overview



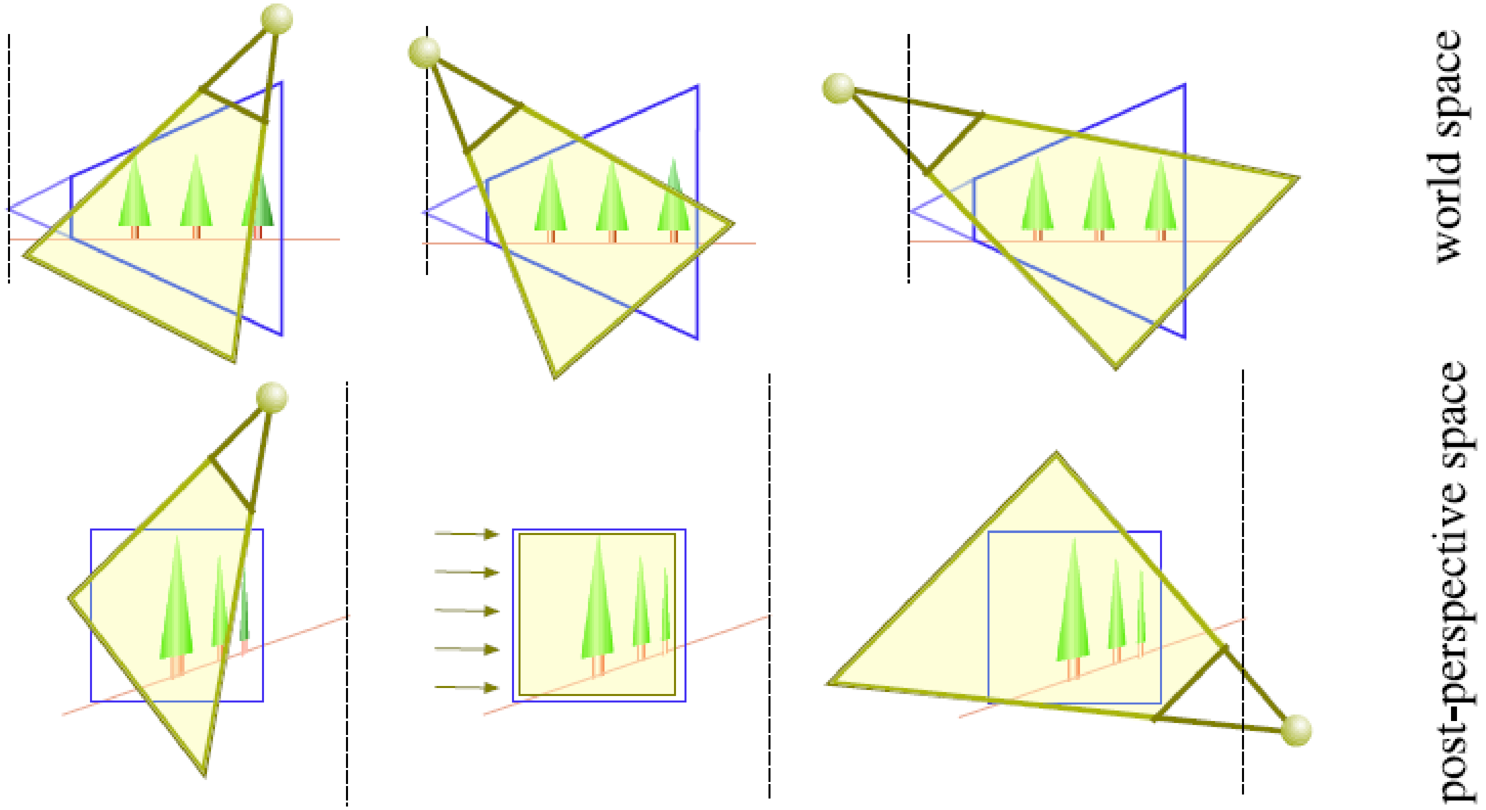
Perspective Shadow Maps

- Directional Light Sources



Perspective Shadow Maps

- Point Light Sources

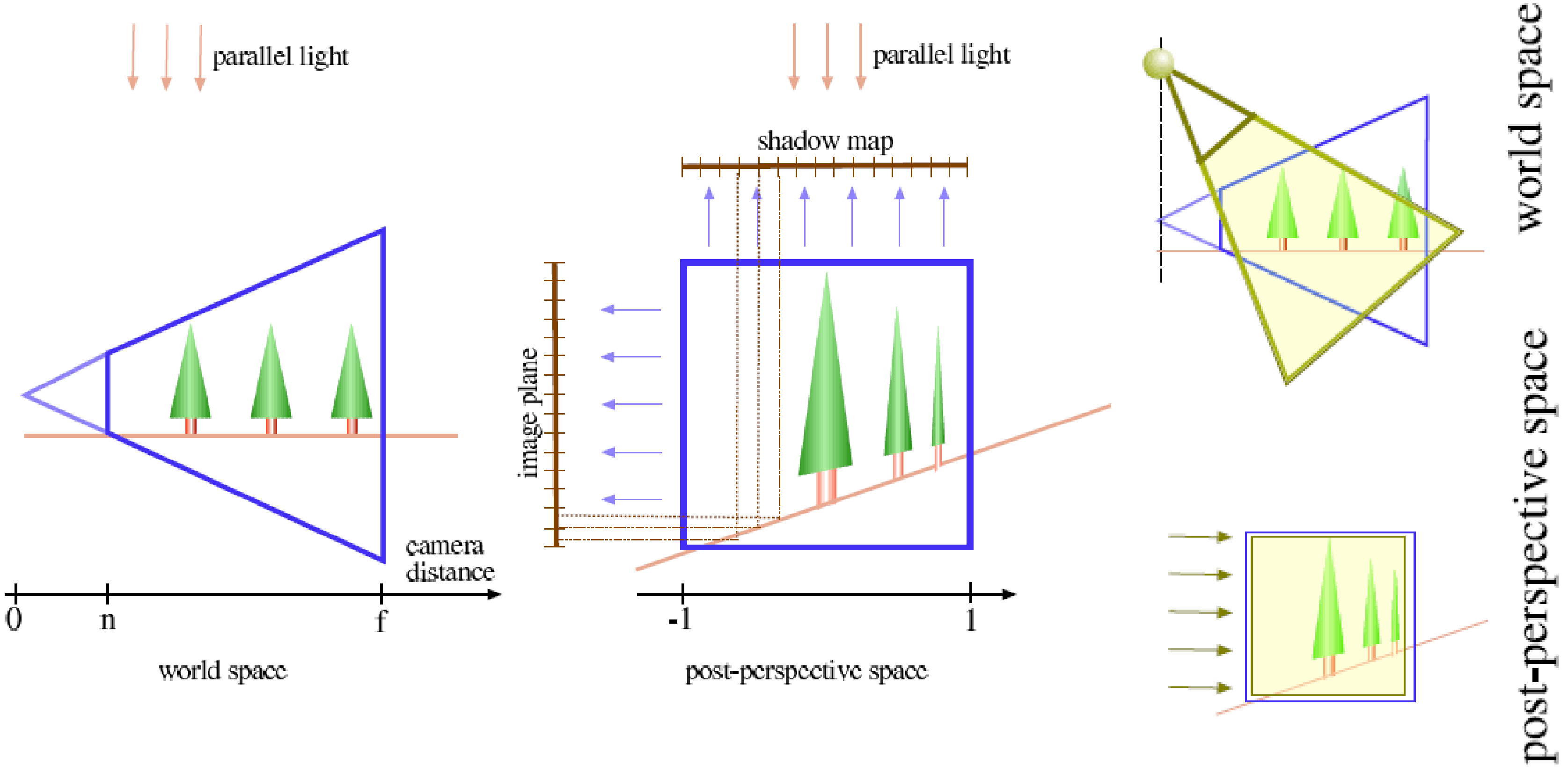


Perspective Shadow Maps

- Discussions
  - In post-perspective space, the final image is an orthogonal view onto the unit cube.
  - Perspective aliasing due to distance to the eye, r_i, is avoided
  - However, if the light source is mapped to a point light in post-perspective space, aliasing due to the distance to the shadow map image plane, r_s, can appear.

Perspective Shadow Maps

- Discussions
  - Ideal cases

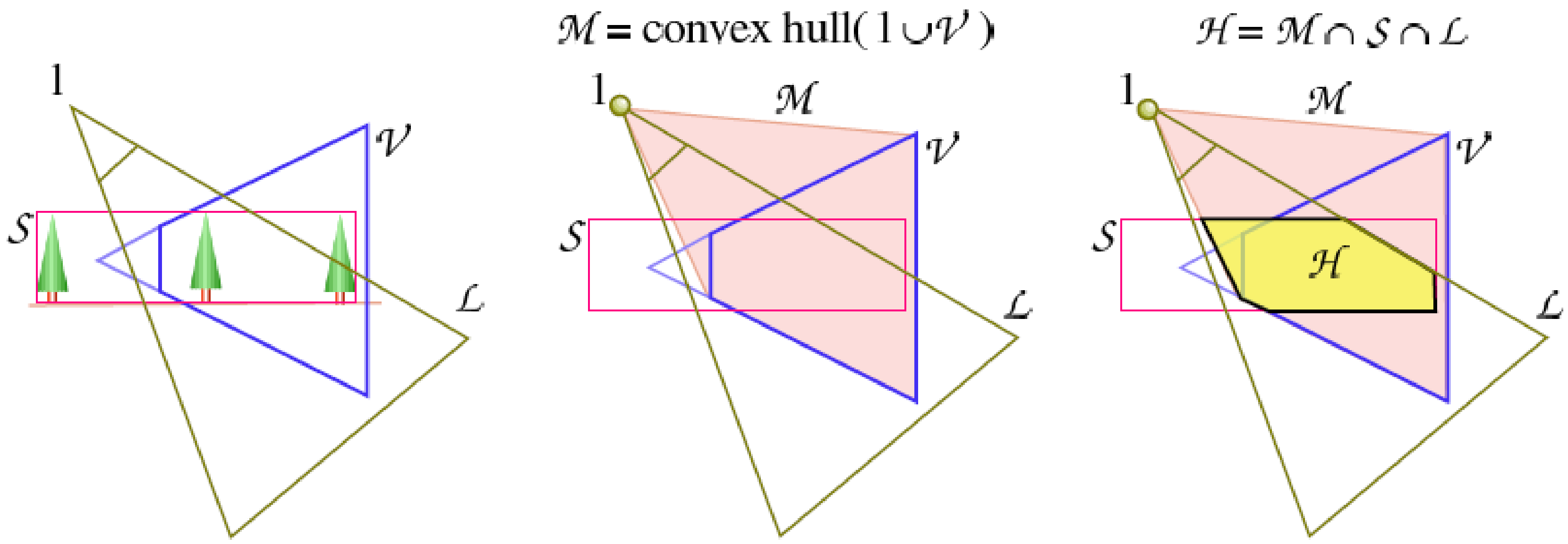


Perspective Shadow Maps

- Discussions
  - For directional light sources
    - The benefit is maximal for a light direction perpendicular to the view direction.
    - Consider the smaller of the two angles formed between the light direction and the viewing direction, the benefit of this approach decreases as this angle becomes smaller.
  - For point light sources
    - Analysis is harder
    - Achieve largest advantages when the point light is far away from the viewing frustum…

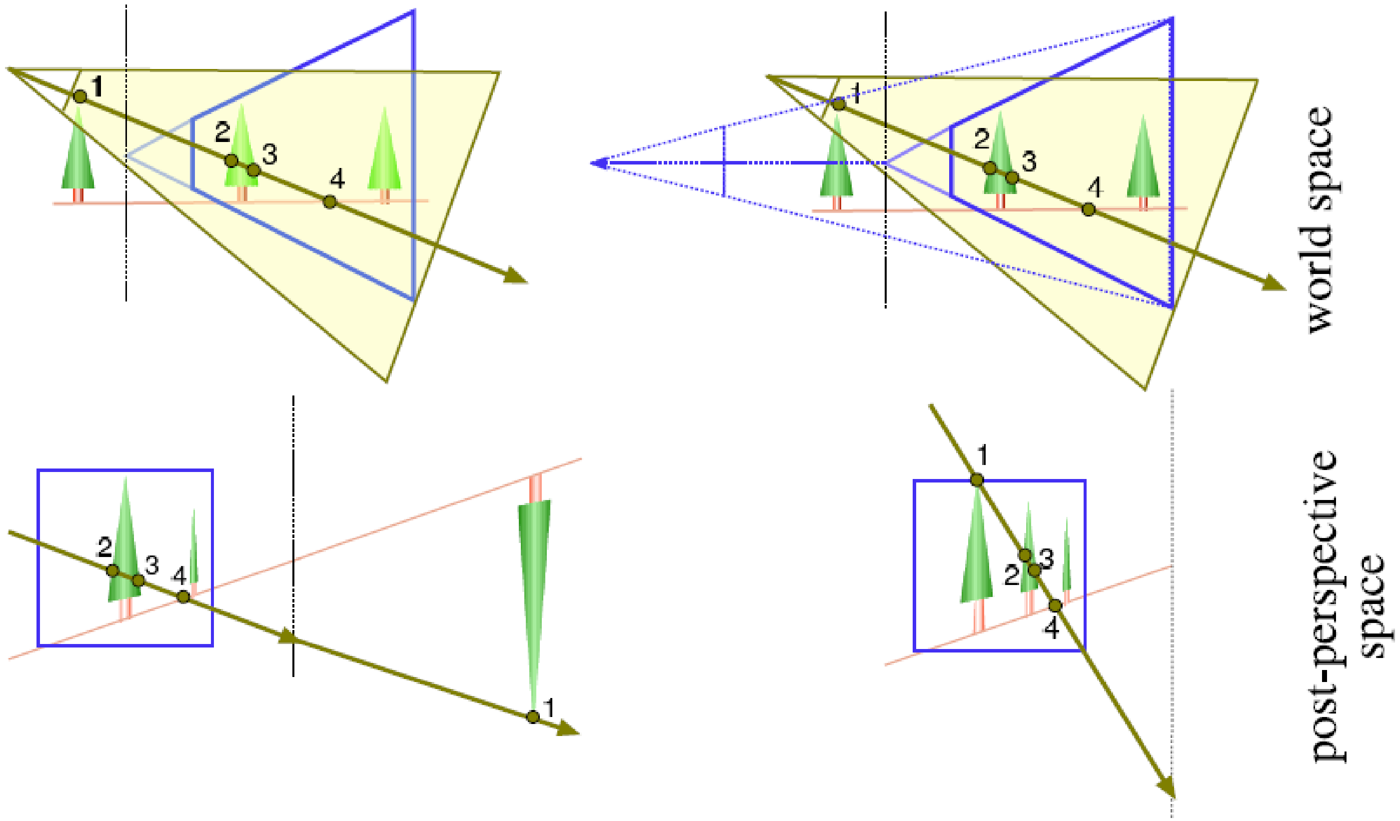
Perspective Shadow Maps

- Including all Objects Casting Shadows



Perspective Shadow Maps

- Including all Objects Casting Shadows



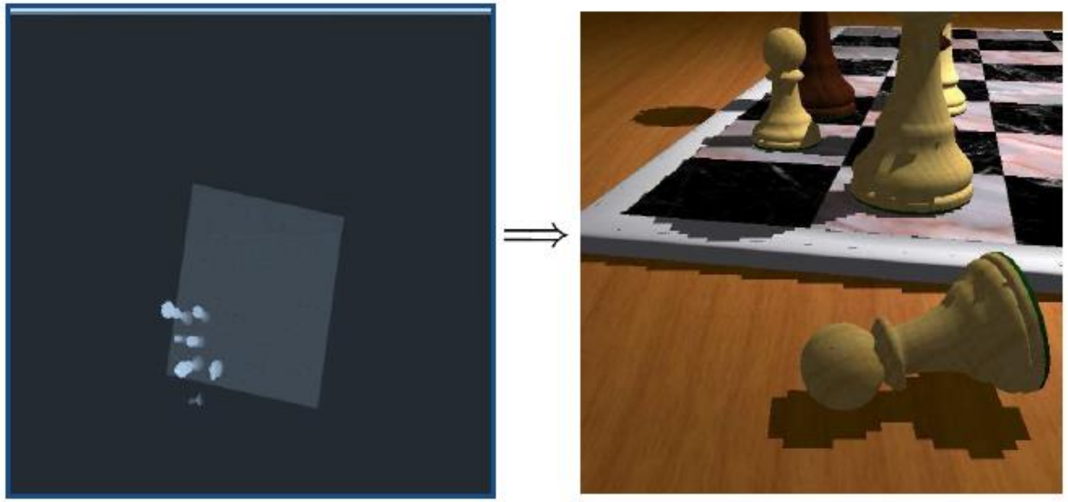
Perspective Shadow Maps

- Including all Objects Casting Shadows
  - We virtually move the camera view point backwards, such that H lies entirely inside the transformed camera frustum;
  - By this, we modify the post-perspective space, resulting in decreased perspective foreshortening.
  - If we move the camera to infinity, we obtain an orthogonal view with a post-perspective space that is equivalent to the original world space; the resulting perspective shadow map then corresponds to a standard uniform shadow map.

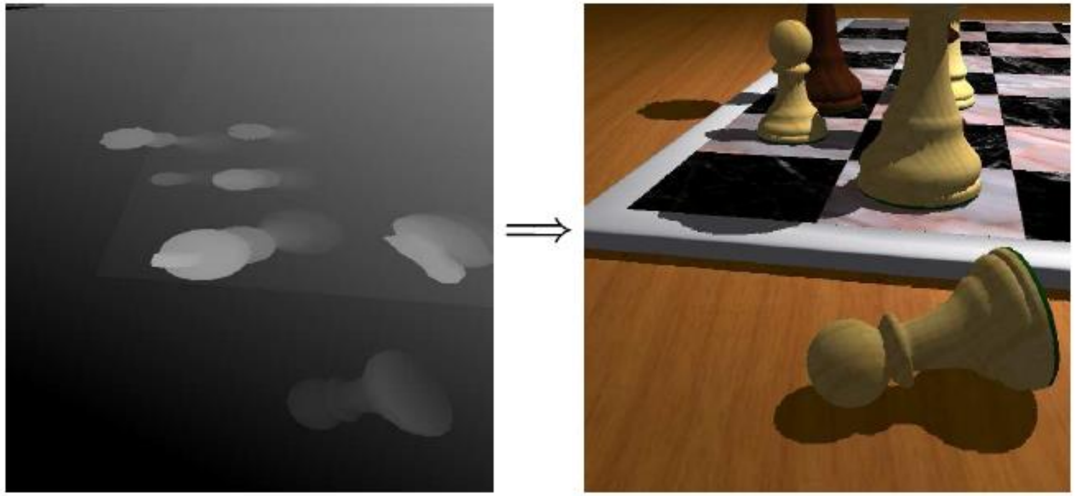
Perspective Shadow Maps

- Results

Uniform Shadow map



Perspective Shadow map



Perspective Shadow Maps

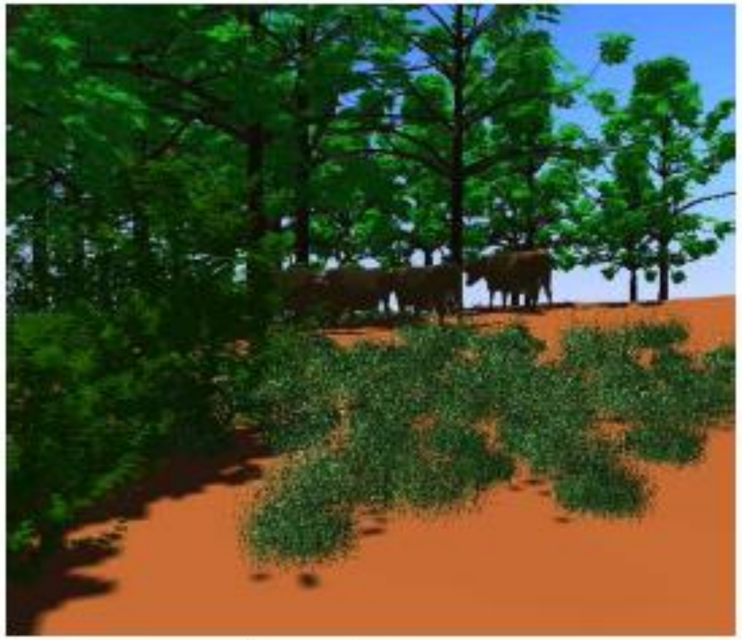
- Results

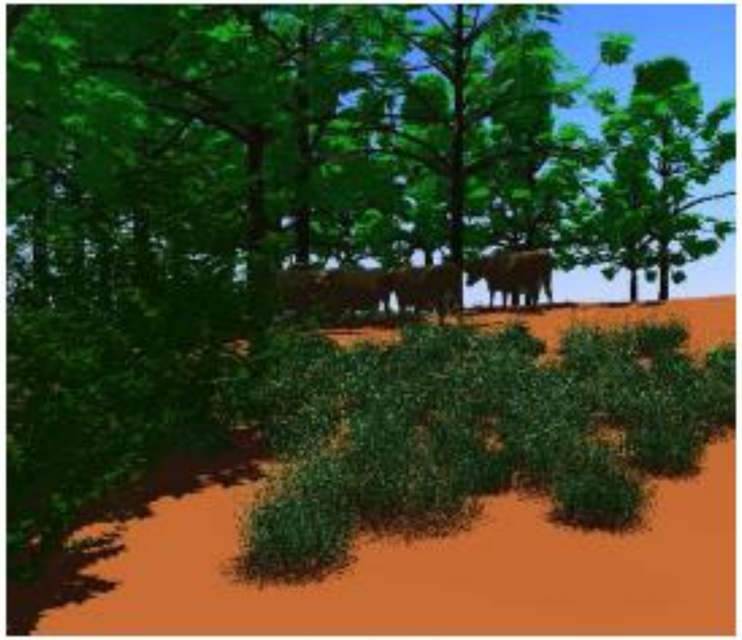
Left is Uniform shadow maps, right is Perspective Shadow Maps

Perspective Shadow Maps

- Results


no shadows


uniform shadow map


perspective shadow map

Perspective Shadow Maps

- Results

Perspective Shadow Maps

- Conclusions
  - Introduced perspective shadow maps, a novel parameterization for shadow maps.
  - The method permits the generation of shadow maps with greatly improved quality, compared to standard uniform shadow maps.
  - Perspective shadow maps can be used in interactive applications and fully exploit shadow map capabilities of recent graphics hardware, but they are also applicable to high-quality software renderers.

Parallel-split Shadow Maps

- Fan Zhang , Hanqiu Sun , Leilei Xu , Lee Kit Lun, Parallel-split shadow maps for large-scale virtual environment Proceedings of the 2006 ACM international conference, June 14-April 17, 2006, Hong Kong, China

Parallel-split Shadow Maps

- Main Contribution
  - In this paper, they present the Parallel-Split Shadow Maps (PSSMs) scheme, which splits the view frustum into different parts by using the planes parallel to the view plane and then generates multiple smaller shadow maps for the split parts.

Parallel-split Shadow Maps

- Shortcomings of shadow mapping for large-scale virtual environments
  - Texture Resolution
  - Global Reparameterizations
  - Geometry Approximation
  - Dueling Frusta Case

Parallel-split Shadow Maps

- Motivation to avoid these shortcomings due to the facts
  - Each of the split parts has an independent shadow map, different parameterizations can be applied in different depth ranges according to the application's requirement.
  - Since the depth range is split into smaller layers, split scheme changes the resolution requirement from sufficient for every point to sufficient for every depth layer.

Parallel-split Shadow Maps

- Motivation to avoid these shortcomings due to the facts
  - The geometry approximation is applied in each of the depth ranges separately. The tighter bounding shape significantly enhances the utilization of the shadow map resolution.
  - Because each shadow map in PSSMs is focused in smaller sub frusta, the shadow qualities in the dueling frusta case is also greatly improved.

Parallel-split Shadow Maps

- Overview

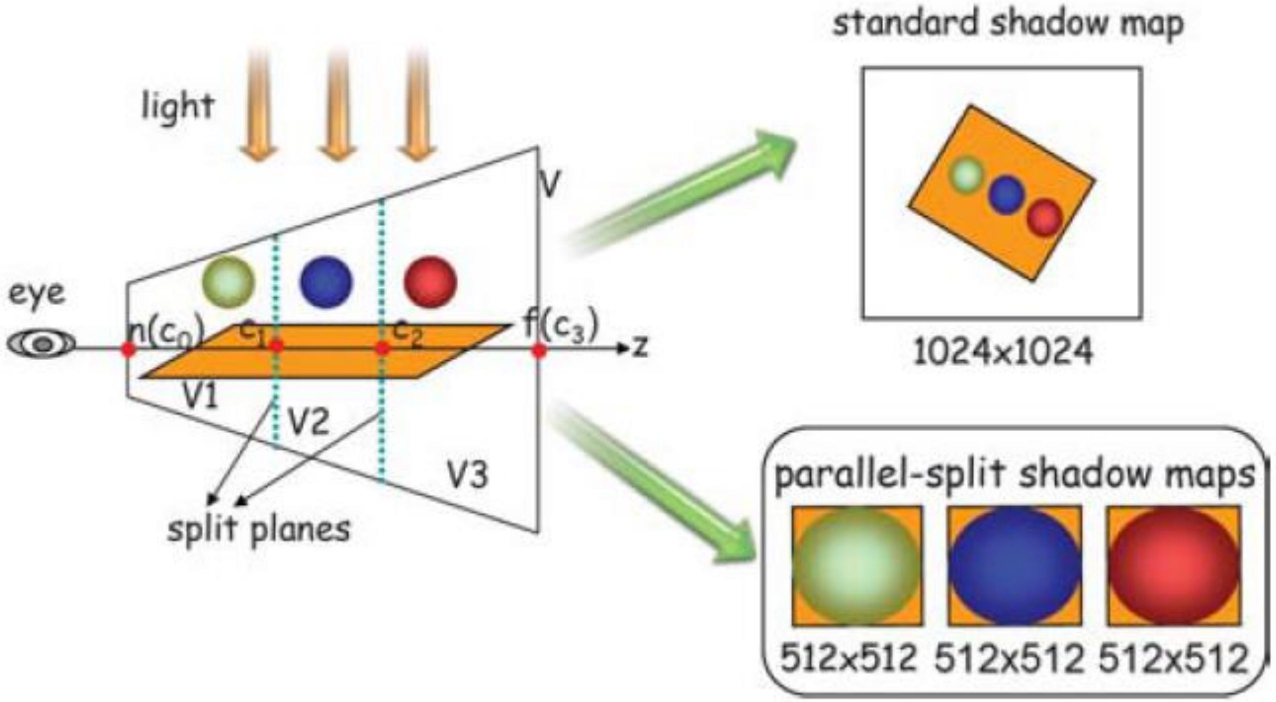
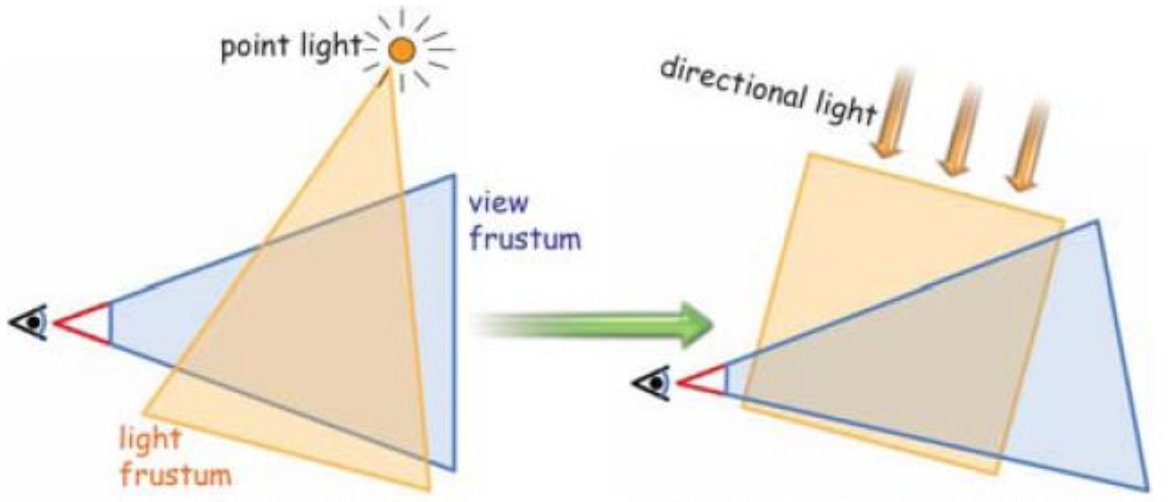


Figure 1: Split the view frustum into three parts, and shadow maps with the same resolution are generated for the split parts.

Parallel-split Shadow Maps

- Light Sources
  - Actually, all kinds of lighting sources including point lighting sources can be unified as directional lighting sources in the light's post-perspective space.



(a) World Space (b) Light's Post-perspective Space

Parallel-split Shadow Maps

- PSSM Algorithm Overview
  - STEP 1 Split the view frustum into multiple depth parts.
  - STEP 2 Split the light's frustum into multiple smaller ones, each of which covers one split part also the objects potentially casting shadows into the part.
  - STEP 3 Render a shadow map for each split part.
  - STEP 4 Render scene shadows for the whole scene.

Parallel-split Shadow Maps

- Some notations

notation	description
V	view frustum.
V_i	the ith split part of $V\,(1 \le i \le m)$.
n and f	near and far planes of V.
m	number of split parts.
C_i	depth of the ith split plane in the view space $(1 \le i \le m-1)$, for convenience we supplement to define $C_0 = n$ and $C_m = f$.
T_i	the shadow map texture for V_i.
$\mathrm{PSSM}(m; [res])$	the split scheme to split V into m parts, and the resolution of each shadow map is res.

Parallel-split Shadow Maps

- View Frustum Split

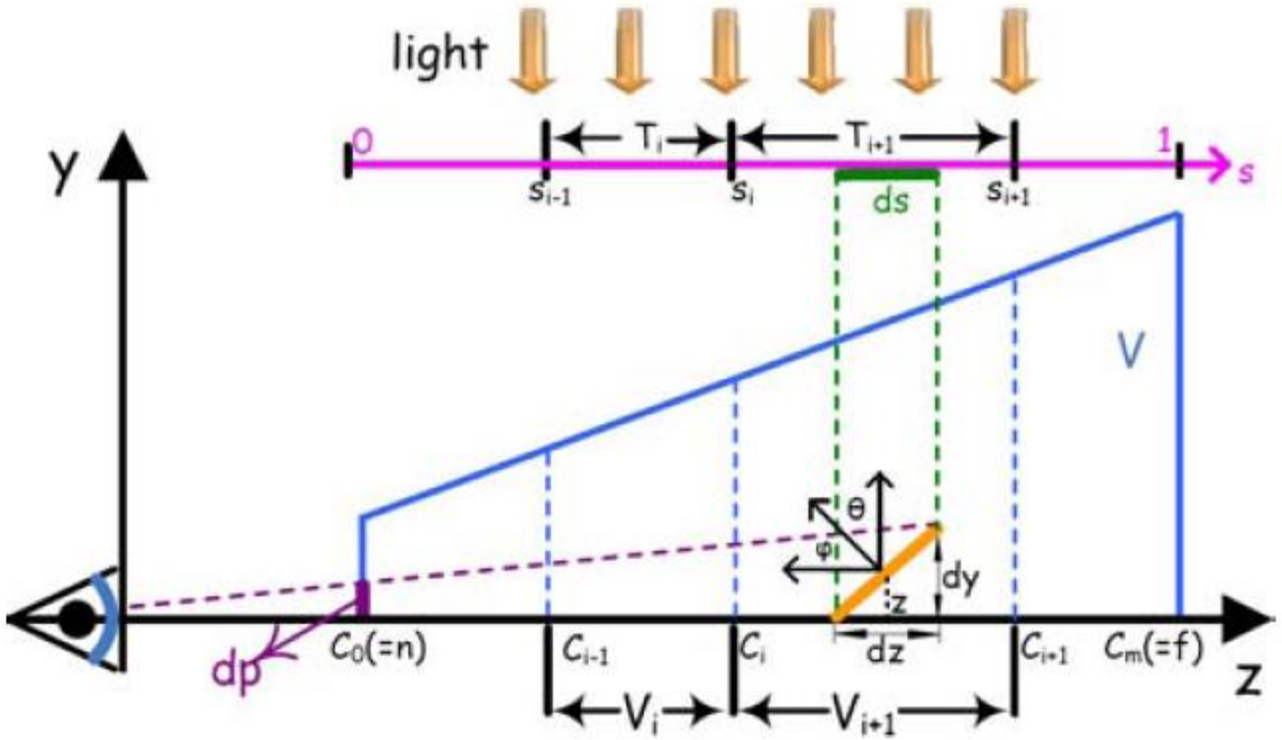


Figure 3: Along the z axis the view frustum is split into parts by using the split planes at $\{C_i \mid 0 \leq i \leq m\}$.

Parallel-split Shadow Maps

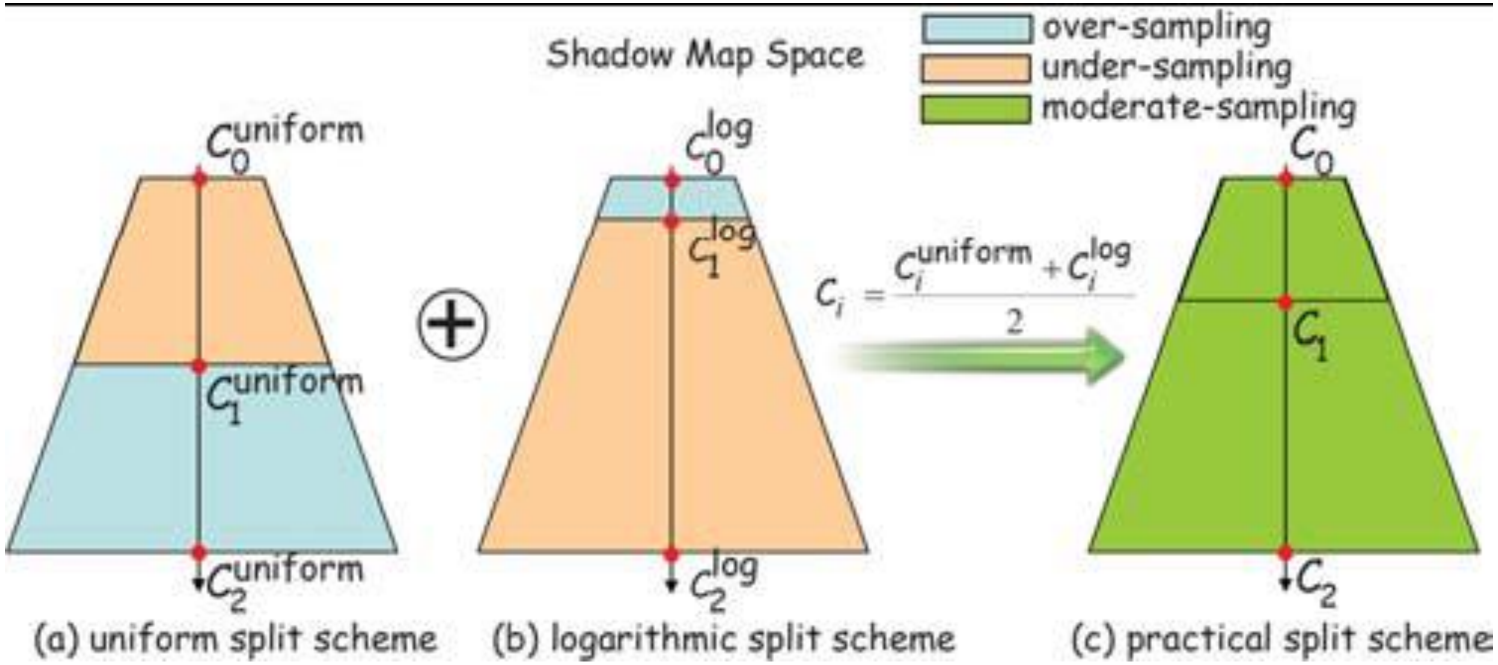
- View Frustum Split
  - Aliasing Problem

$$\frac{\mathrm{d}p}{\mathrm{d}s} = n\frac{\mathrm{d}z}{z\mathrm{d}s}\frac{\cos\varphi}{\cos\theta}$$

    - Perspective Aliasing: comes from the perspective foreshortening effect, can be reduced by applying a global transformation to warp the shadow map texels.
    - Projection Aliasing: related to the scene's geometry details, the local increase of sampling densities on this surface is required to reduce this category of aliasing.
    - Split scheme comes from these analysis

Parallel-split Shadow Maps

- View Frustum Split
  - Three kinds of split



Parallel-split Shadow Maps

- Logarithmic Split Scheme
  - Suppose $\mathrm{d}z/z\mathrm{d}s = \rho$ to be constant
  - Then we can deduce

$$s = \int_0^s \mathrm{d}s = \frac{1}{\rho}\int_n^z \frac{\mathrm{d}z}{z} = \frac{1}{\rho}\ln(z/n)$$

  - Based on the assumption of $s \in [0,1]$ easily we have $\rho = \ln(f/n)$.then we can get

$$s = \frac{\ln(z/n)}{\ln(f/n)}$$

Parallel-split Shadow Maps

- Logarithmic Split Scheme
  - Equation

$$s_i = s(C_i^{log}) = \frac{\ln(C_i^{log}/n)}{\ln(f/n)} \qquad s = \frac{\ln(z/n)}{\ln(f/n)}$$

  can be discretized as

$$C_i^{log} = n(f/n)^{s_i}$$

  - Or
    - Because this split scheme is designed to produce the theoretically even distribution of perspective aliasing errors, the resolution allocated for $[C_i, C_{i-1}]$ should be $s_i=i/m$ of the overall texture resolution.
    - Finally, we get

$$C_i^{log} = n(f/n)^{i/m}$$

Parallel-split Shadow Maps

- Logarithmic Split Scheme
  - The main drawback of this split scheme is that the lengths of split parts near the viewer are too small, so few objects can be included in these split parts.
  - This is due to the theoretically optimal parameterization assumes that the shadow map accurately covers the view frustum and no any resolution is wasted on invisible parts of the scene.

Parallel-split Shadow Maps

- Uniform Split Scheme
  - The simplest split scheme is to place the split planes uniformly along the z axis:

$$C_i^{uniform} = n + (f - n)i/m.$$

  - Because $s = (z - n)/(f - n)$
  - The perspective aliasing is

$$\frac{dp}{ds} \doteq n\frac{dz}{zds} = \frac{n(f-n)}{z}$$

Parallel-split Shadow Maps

- Uniform Split Scheme

$$\frac{\mathrm{d}p}{\mathrm{d}s} \doteq n\frac{\mathrm{d}z}{z\mathrm{d}s} = \frac{n(f-n)}{z}$$

  - Perspective aliasing in uniform reparameterizations increases hyperbolically as the object moves near to the view plane.
  - Therefore, uniform split scheme results in under-sampling at the points near the viewer, over-sampling at the points further from the viewer.

Parallel-split Shadow Maps

- Practical Split Scheme
  - Logarithmic split scheme provides the theoretically optimal aliasing distribution, while uniform split scheme results in the theoretically worst aliasing distribution.
  - Practical split scheme is designed to moderate sampling densities in the above two extreme split schemes.

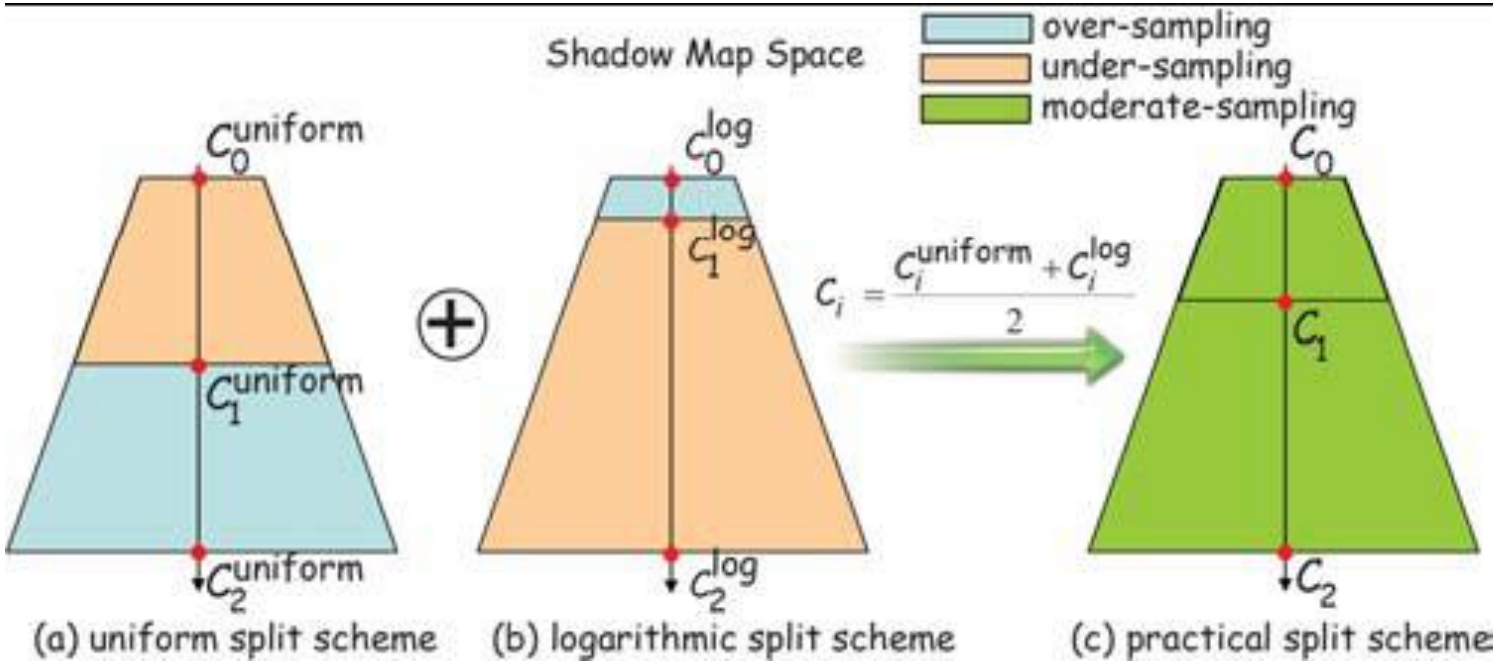
$$C_i = (C_i^{log} + C_i^{uniform})/2$$

  - Or

$$C_i = \frac{n(f/n)^{i/m} + n + (f-n)i/m}{2} + \delta_{bias}, \quad \forall 0 \leq i \leq m.$$

Parallel-split Shadow Maps

- View Frustum Split
  - Three kinds of split



Parallel-split Shadow Maps

- Light's Frustum Split

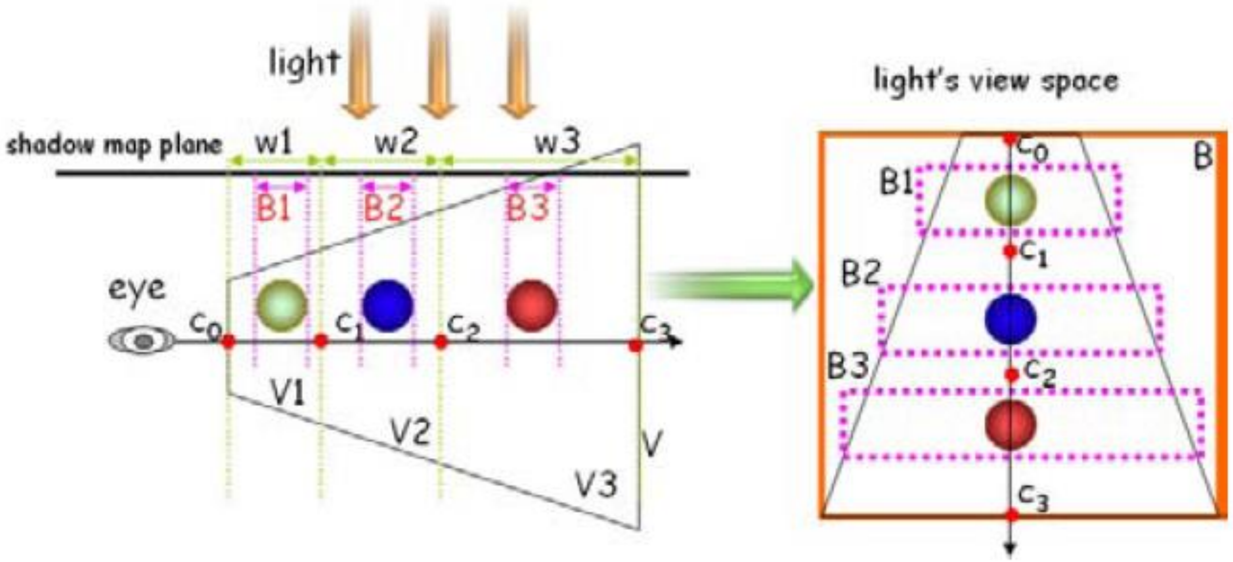


Figure 5: The light's frustum W is split into $\{W_i\}$, each of them covers the split part V_i. Then, applying geometry approximation to W_i to produce a focused bounding shape B_i. The sum of B_i is smaller than the bounding shape B of V.

Parallel-split Shadow Maps

- Scene-shadows Rendering
  - Like standard shadow mapping, each pixel should be transformed into the light space when determining if the pixel is shadowed or not.
  - The differences here are:
    - Select the correct shadow map T_i
    - The pixel should be transformed into W_i rather than W.

Parallel-split Shadow Maps

- Scene-shadows Rendering
  - Multiple texture maps are required.
  - In order to avoid multiple passes for the final scene-shadows rendering, they utilized pixel shader available on programmable GPUs.
  - For each rasterized fragment, they sample the appropriate shadow map based on the depth value of this fragment.

Parallel-split Shadow Maps

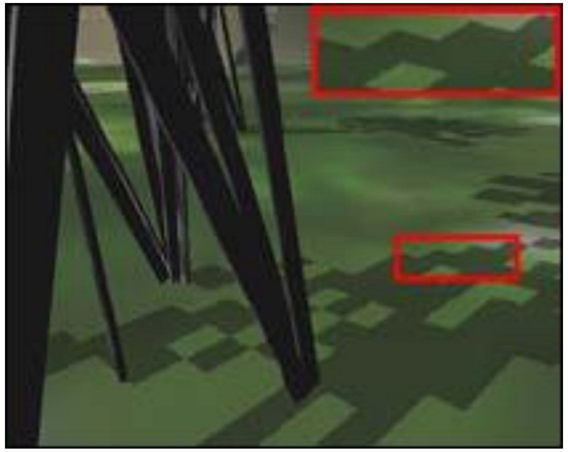
- Scene-shadows Rendering
  - In the view space, obviously, T_i should be used for points located in the range $[C_i, C_{i-1}]$.
  - However, in the fragment buffer, the coordinates are measured in the clip space. So we need to transform
  - If $C_i^{clip} = \frac{f}{f-n}(n - \frac{1}{C_i}) \in [0,1]$ the T_{index} is selected

$$\frac{f}{f-n}(n - \frac{1}{C_{index-1}}) \leq z^{clip} \leq \frac{f}{f-n}(n - \frac{1}{C_{index}})$$

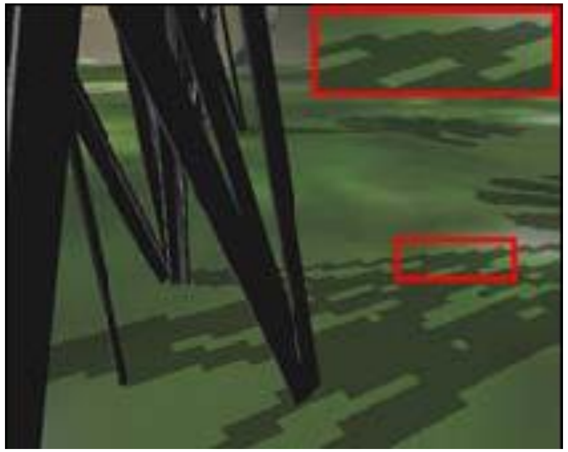
Parallel-split Shadow Maps

- Results

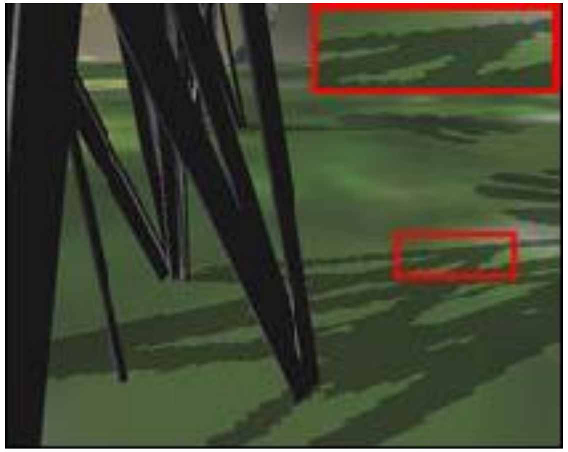
SSM



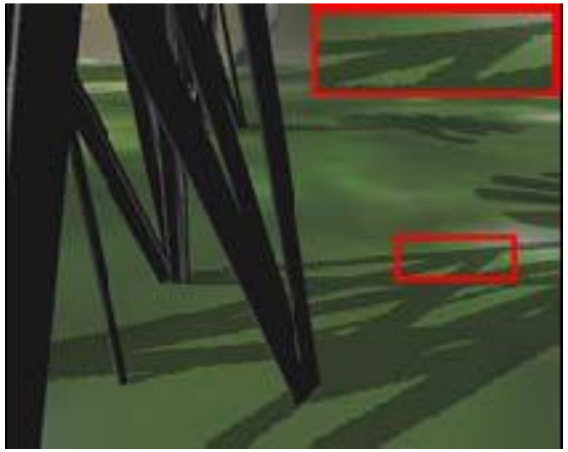
PSM



TSM



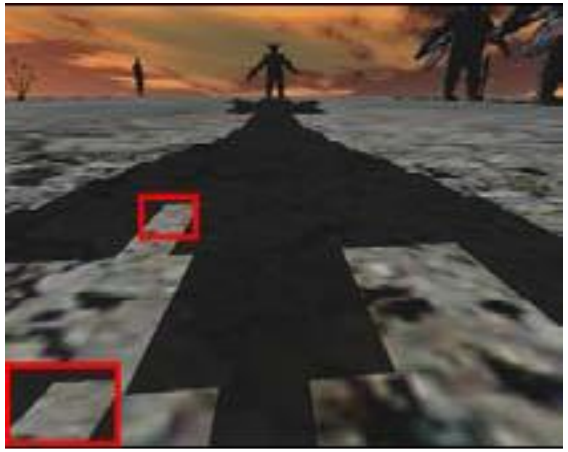
PSSM(3)



Parallel-split Shadow Maps

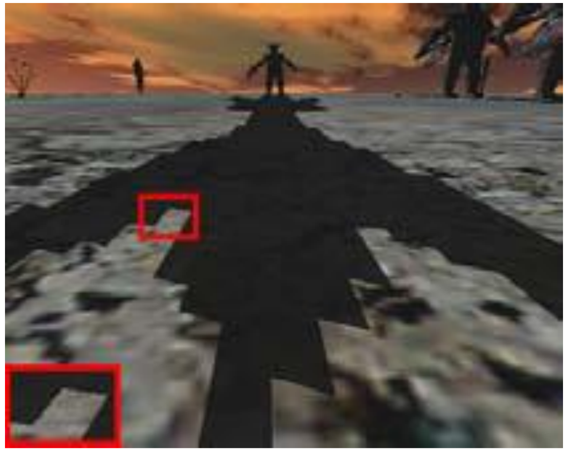
- Results

SSM



PSM

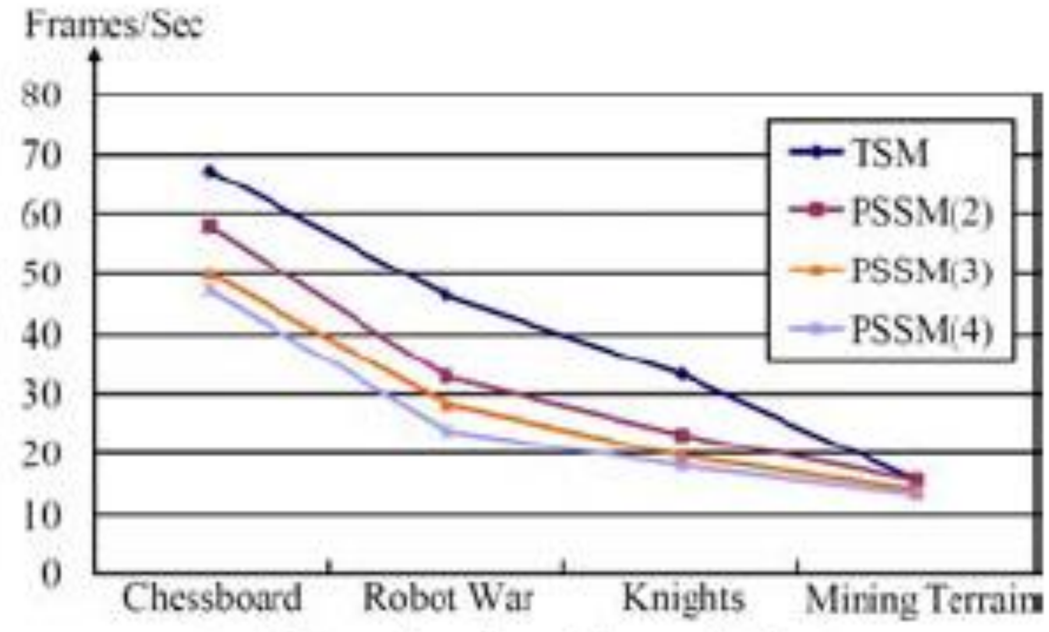
TSM



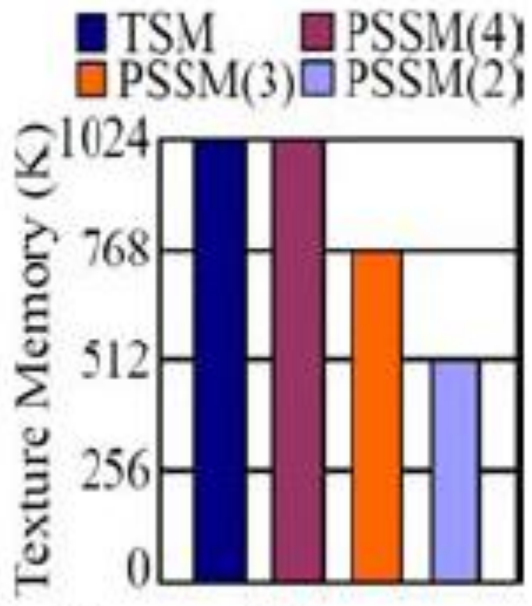
PSSM(3)

Parallel-split Shadow Maps

- Results



(a) Rendering Speed (fps)



(b) Texture Memory (K)

Parallel-split Shadow Maps

- Conclusions
  - This paper developed the Parallel-Split Shadow Maps (PSSMs) scheme, which splits the view frustum into different depth ranges by using split planes parallel to the view plane, and then renders multiple shadow maps for the split parts.
  - They proposed a fast and robust split scheme without expensive scene analysis per frame, which produces moderate sampling densities over the whole depth range.
  - Future work: hardware-accelerated PSSMs to reduce rendering passes for the generation of shadow maps (e.g. using MRT on current shader model).

Summarizations

- Shadowing effects dramatically enhance the realism of virtual environments by providing useful visual cues.
- Shadowing algorithm can be divided into two main categories: Shadow Volume and Shadow Mapping
- Most of games are using shadow mapping as it is much faster.
- Several approach has been conducted to improve the shadow mapping quality. They can be classified as two group: Warped shadow maps and split shadow maps.
- Split shadow maps are most popular in nowadays game engine.

Thanks

- Used slides from:
  - Stefan Brabec, Marc Stamminger, Zhang et al., Ulf Assarsson, Lukai Lan, and others

Improved Hard shadows

Pere-Pau Vázquez
ViRVIG – UPC