Parallel-Split Shadow Maps for Large-scale Virtual Environments

Hanqiu Sun

VR, Visualization and Imaging Research Centre
Dept. of Computer Science & Engineering
The Chinese University of Hong Kong

Motivation of Shadow Rendering

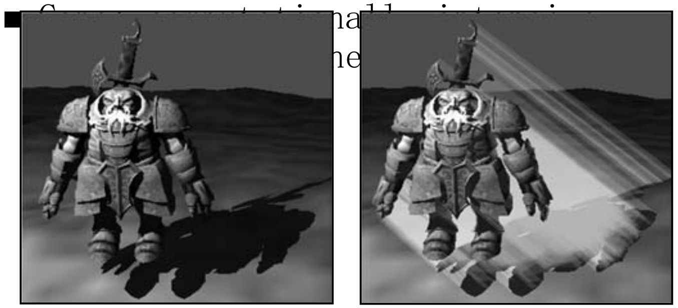
- Important for increasing scene realism
- Current graphics programming APIs didn' t implement shadow effects
  - Use local illumination models which don' t consider the spatial relation between objects.
- Global illumination models
  - Highly realistic rendering effect (ray tracing, radiosity, photon mapping)
  - Hard to apply for real-time applications

Shadow Rendering Techniques

- Shadow Volume
  - Object-space algorithm
  - Accurate
  - Computational expensive, good tessellation
- Shadow Mapping
  - Image-space algorithm
  - Fast rendering speed, only one extra rendering pass needed
  - Aliasing errors from insufficient texture resolution

Introduction

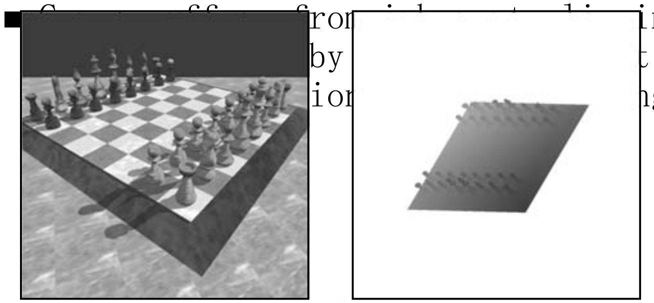
- Object-based Algorithm: Shadow Volumes
  - Pros: with the screen-space resolution.



shadows visualization of shadow volumes

Introduction

- Image-based Algorithm: Shadow Mapping
  - Pros: widely used in real-time applications by its efficiency and generality.
  - ■ ... from ... ing ... by ... ion ... ng).

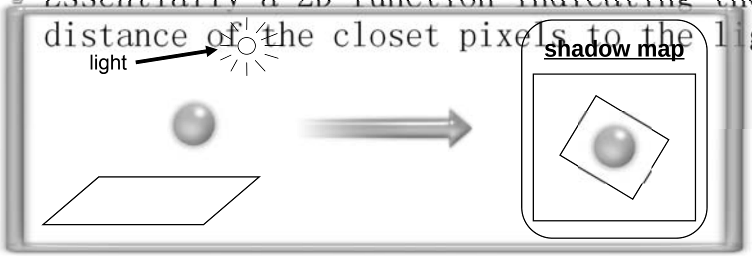


Shadow Mapping

- What does image-based mean?
  - No geometry information needed
  - Suffers from aliasing artifacts
- Multi-pass rendering
  - Render the scene two passes rather single one
  - Render the scene from the light's point-of-view
    - Extract depth buffer as a texture, i.e. shadow map
  - Render the scene normally with the depth map

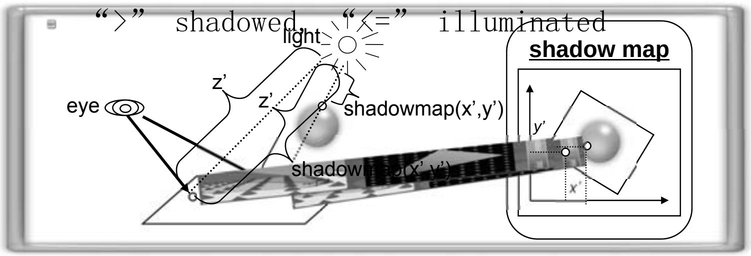
Shadow Mapping

- Pass 1: depth testing in light's point-of-view
  - Current depth buffer is a "depth map" or "shadow map"
  - Essentially a 2D function indicating the distance of the closet pixels to the light



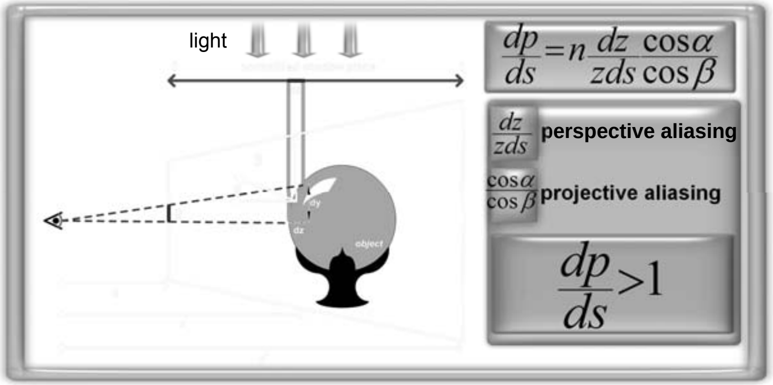
Shadow Mapping

- Pass 2: render shadows with shadow map
  - Transform each pixel to light's space
    - (x, y, z) in viewer's space → (x', y', z') in light's space
  - Compare z' to the depth value at (x', y') stored in shadow map
    - ">" shadowed, "<=" illuminated



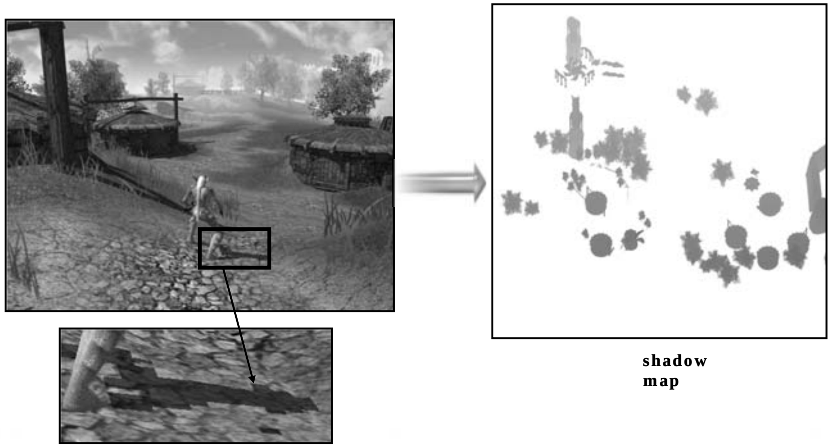
Aliasing Errors Analysis

- Where do aliasing errors come from?



Aliasing in Shadow Mapping

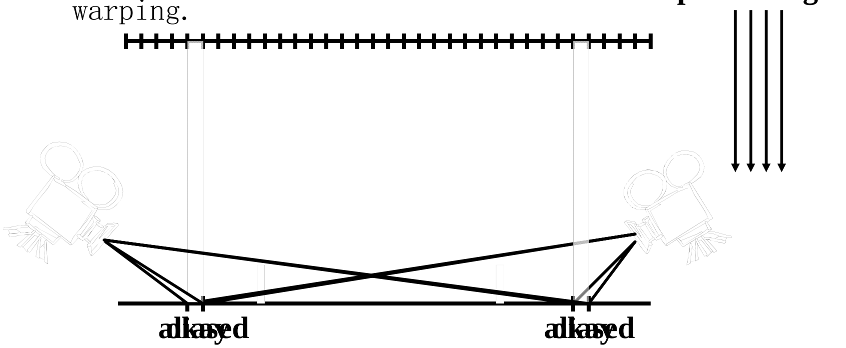
- Anti-aliasing is required in shadow mapping due to the image-based nature.



shadow map

Perspective Aliasing

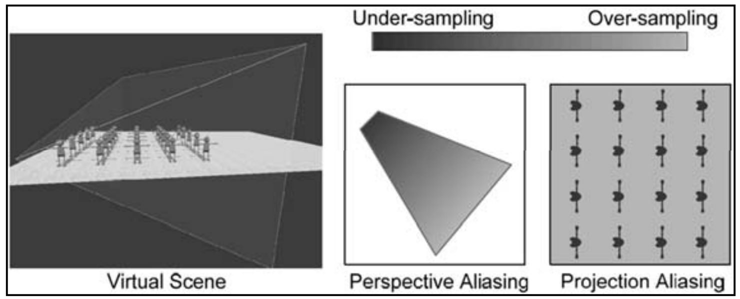
- View dependent, under-sampling for near areas, super-sampling for distant areas.
- The only kind of aliasing can be alleviated by a global transformation like a perspective warping.



Courtesy of SIGGRAPH'04 course notes

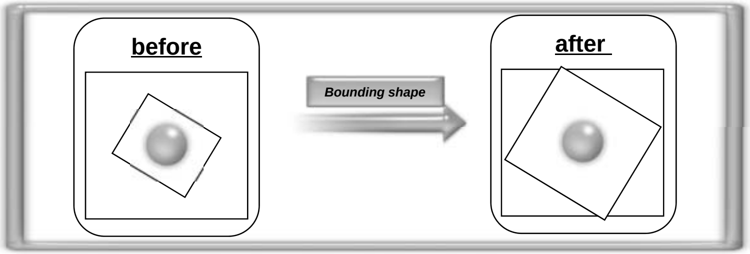
Projection Aliasing

- Reducing projection aliasing force us to abandon hardware-acceleration. (pure software solution)
- In this presentation, we only address on the techniques reducing perspective aliasing errors.



Main Research Directions

- Geometry approximation latest: EUROGRAPHIC'04
  - Focus shadow map on visible objects
  - Approximate the bounding shape of the intersection between light frustum and view frustum



Main Research Directions

- Perspective shadow maps latest: SIGGRAPH'02, EUROGRAPHIC'04
  - Render shadow map in post-perspective space rather traditional view space
- Hieratical data structure latest: SIGGRAPH'01, CGI'04
  - Adaptive resolution to viewer' s position
  - Inconveniently used on current hardware
- Hybrid shadow maps with shadow volume latest: SIGGRAPH'03
  - Take both merits and drawbacks

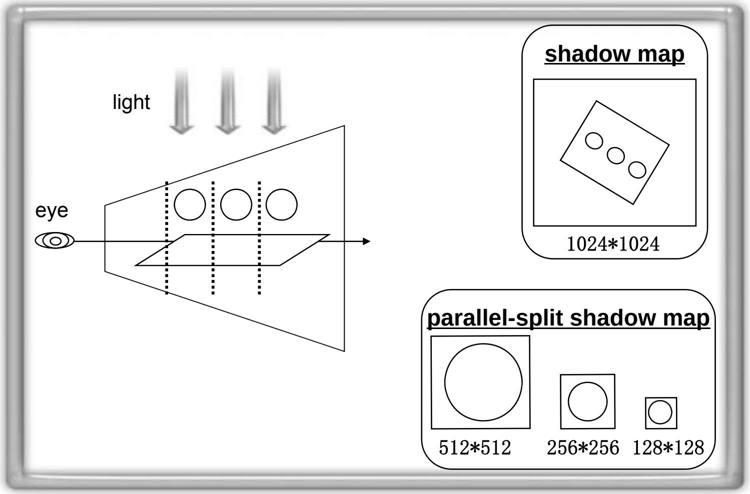
Problems for Current Methods

- Re-parameterize shadow map for the whole scene
  - Treat all objects in the same manner
- Single shadow map with huge resolution
  - Impractical to memory-sensitive applications
- Geometry approximation approaches still need improvement
  - More accurate approximation, more resolution increasing

Parallel-Split Shadow Maps

- Use multiple smaller shadow maps rather single huge one
  - Total memory requirement is less, but with better shadow qualities
- Use multiple planes parallel to view plane to split the scene into different depth layers
  - Re-parameterization is proceeded in each layer rather the whole scene

Parallel-Split Shadow Maps

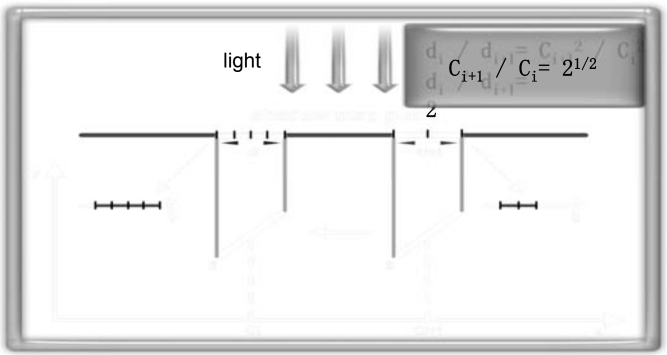


Advantages

- Suitable for large-scale environments
  - Make the distribution of aliasing errors along depth range more even
- Less texture memory requirement
- Higher resolution utilization ratio
  - More compact bounding shape
- Extendable framework easily integrated with other shadow mapping algorithms
  - Any algorithm can be applied to any layer

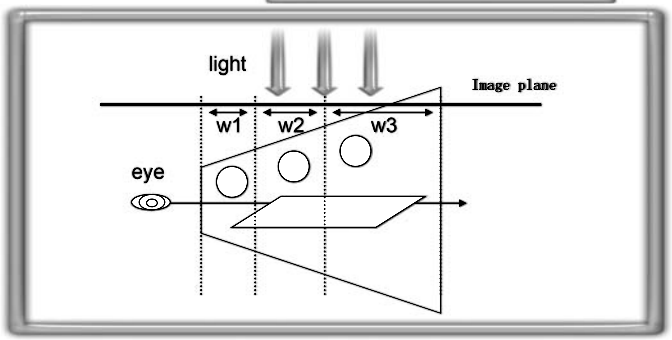
Implementation Steps

- Step 1: view frustum split
  - How to determine the split planes' positions?



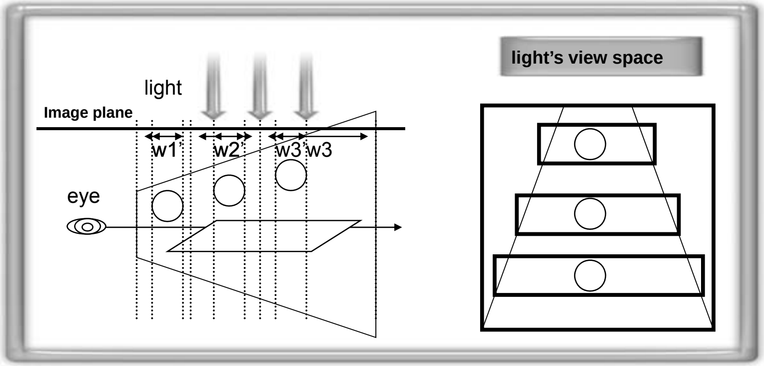
Implementation Steps

- Step 2: light's view plane split
  - Calculate the shadow map viewport for each split part, i.e. shadow map window



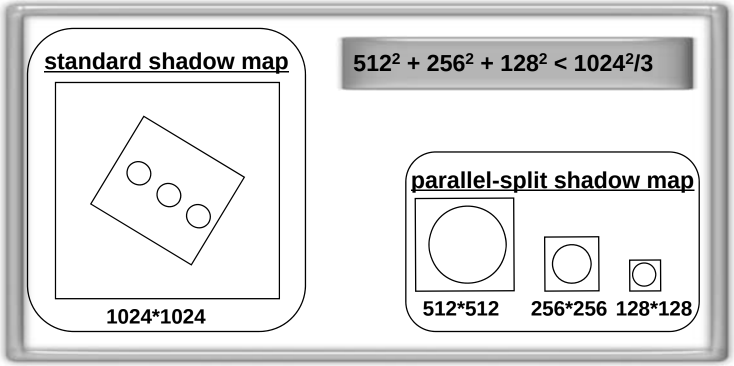
Implementation Steps

- Step 3: focus shadow map windows
  - Increase resolution for each shadow map



Implementation Steps

- Step 4: PSSMs rendering
  - Resolutions configuration



Implementation Steps

- Step 5: shadowed scene rendering
  - Need multiple rendering passes for shadow maps generation
    - Multiple Rendering Targets (OpenGL 2.0/DirectX 9)
  - Need multiple rendering passes for final scene-shadows rendering
    - Pixel Shader
  - Practical number of split parts
    - split-1, split-2

PSSMs on DX9 Hardware

DirectX-9 Level GPU acceleration (e.g. GeForce 6800)

For each pixel *p(x, y, z)* in pixel shader, if

the associated shadow map T_i and texture coordinates tex_i should be selected to do shadow determination.

Partially GPU-accelerated: by using pixel shader on DX9 GPU, scene-shadows rendering only needs single one rendering pass.

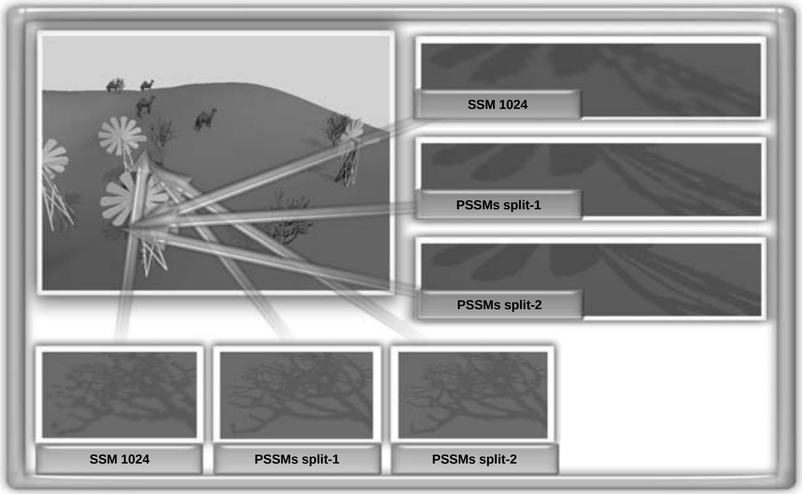
PSSMs on DX10 Hardware

DirectX-10 Level GPU acceleration (e.g. GeForce 8800)

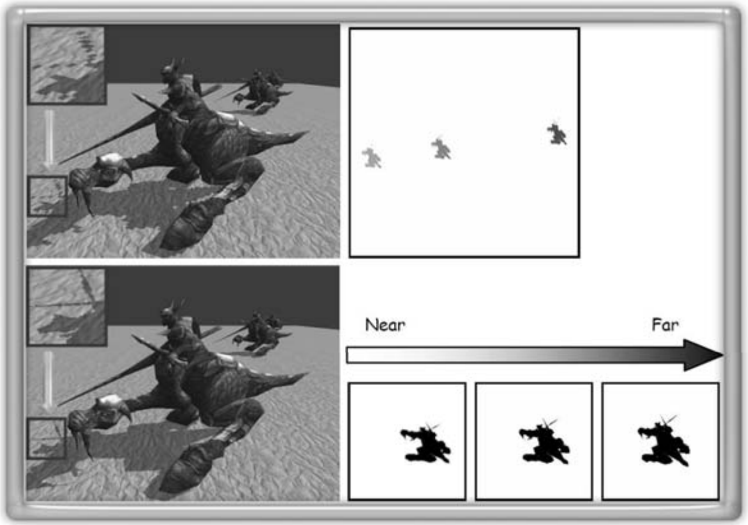
- render target array allows rendering the object into multiple render targets and depth stencil textures simultaneously.
- geometry shader controls which render target every primitive will be sent to and applies the appropriate transformation matrix to the vertex coordinates.

Fully GPU-accelerated: with the above two brand new features, both generating shadow maps and scene-shadows rendering only require single one rendering pass!

Experimental Results



Parallel-Split Shadow Maps



View Frustum Split

- Uniform Split Scheme
  - the aliasing distribution is same as that of standard shadow maps.
- Logarithmic Split Scheme
  - this scheme produces the theoretically even distribution of perspective aliasing errors.
- Practical Split Scheme
  - this scheme produces the "moderate" sampling densities over the whole depth range.

Uniform Split Scheme

- $\{C_i^{uniform}\}\ 0 \le i \le m$

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

  - the sampling densities are same as standard shadow maps.
  - under-sampling for the objects near the viewer, over-sampling for the distant objects.
  - worst case.

Logarithmic Split Scheme

- $\{C_i^{\log}\}\ \ 0 \le i \le m$

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

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

  - since this scheme produces the theoretically even aliasing distribution.

$$s_i = 1/n \Rightarrow C_i^{\log} = n(\frac{n}{f})^{i/m}$$

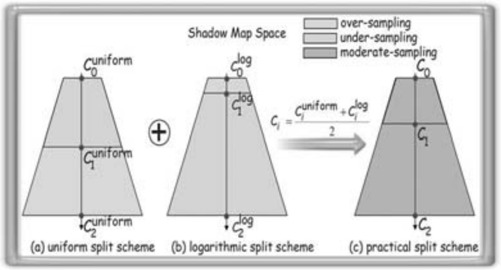
  - the theoretically best scheme assumes that no any resolution wasted for the invisible parts.
  - in practice, too much resolution assigned to the objects near the viewer.

Practical Split Scheme

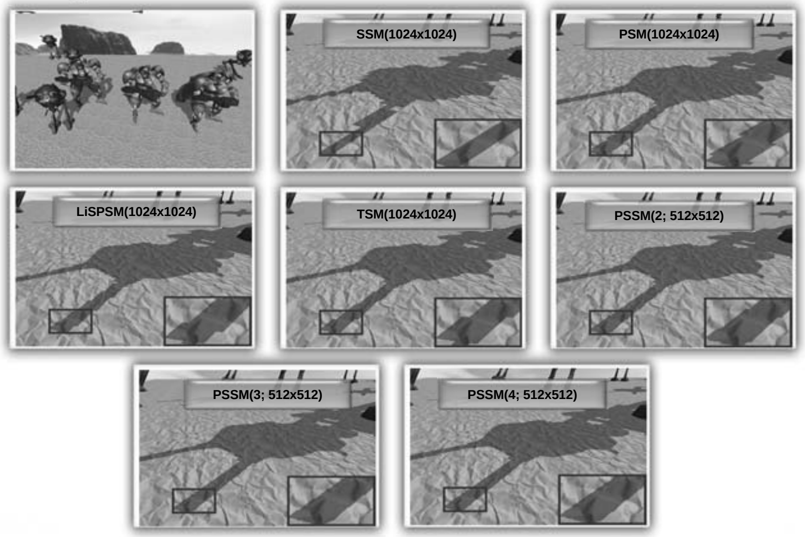
- $\{C_i\}\ \ 0 \le i \le m$

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

  - moderate the optimal and worst sampling densities.
  - no need for the complicated analysis.

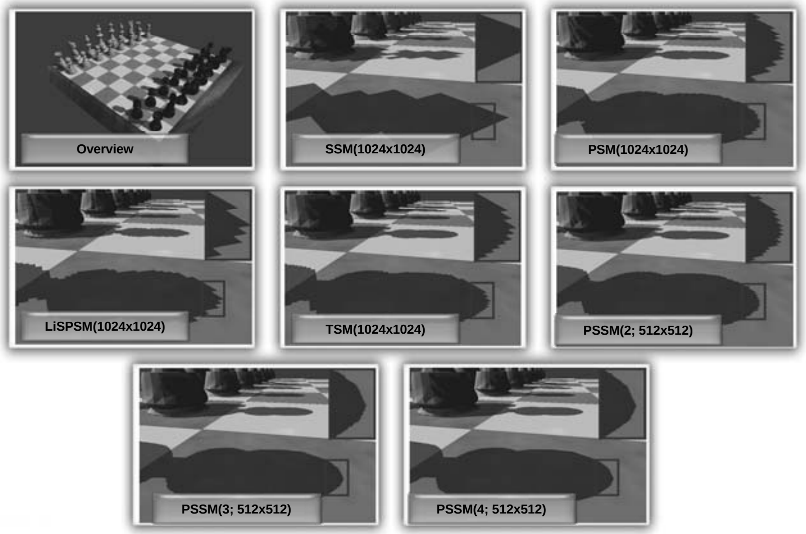


Experimental



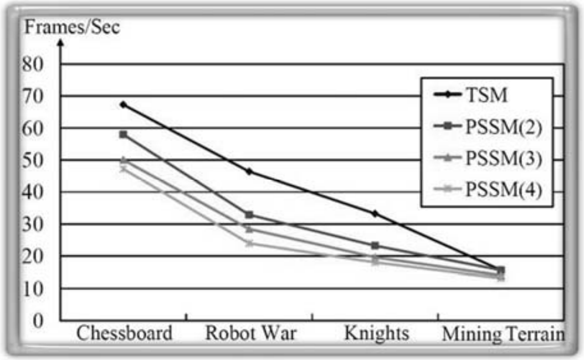
Experimental Results (2)

Experimental Results (3)



Performance

- Real-time for PSSM(2), PSSM(3), PSSM(4).



Pixel Shader Program

```
float4 PixelShader_Program(PS_INPUT IN) : COLOR
{
    ... // other statements
    float bias = 0.004;
    bool stop = false;
    float shadow_value = 0;
    for (int i=0; i<3 && !stop; i++){
        if (pos.z/pos.w <= splitPositions[i]) {
            float depth = IN.tex[i].z / IN.tex[i].w;
            float depth_SM = tex2Dproj (PSSM_Sampler[i], IN.tex[i]);
            shadow_value = (depth < depth_SM + bias);
            stop = true;
        }
    }
    return shadow_value * color;
}
```

PSSMs Applications

Adopted in commercial games! PSSM[4; 1Kx1K]

Screenshots from the game **Dawnspire: Prelude** (*www.dawnspire.com*) courtesy of Silent Grove Studios

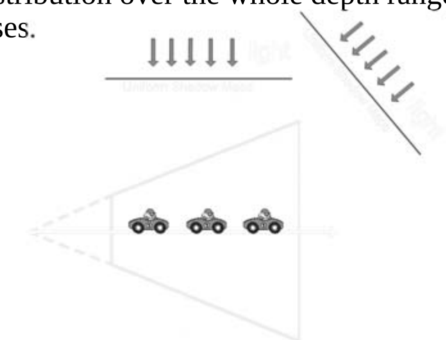
Visual Results

PSSMs Applications

- The famous 3D graphics OpenGL-based SDK **OpenSceneGraph** http://www.openscenegraph.org/ start to support PSSMs.
- The game engine **Hammer Engine** http://www.hmrengine.com/en/ integrates our PSSMs scheme
- Several implementations (including OpenGL- and DirectX-based) of PSSMs have been developed by volunteers.

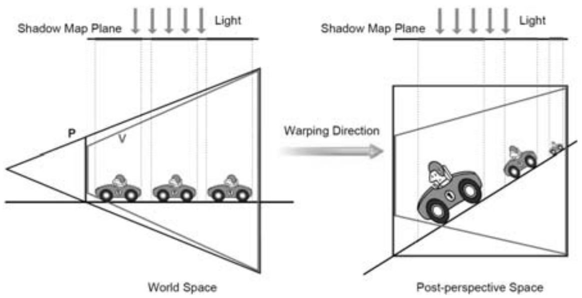
Direction-dependent Parameterization

- Problems of existing parameterizations:
  - The "optimal" distribution of aliasing errors is achieved only in the ideal case, i.e. the light direction and view direction are orthogonal.
  - No generalized theoretical framework to analyze the aliasing distribution over the whole depth range in general cases.

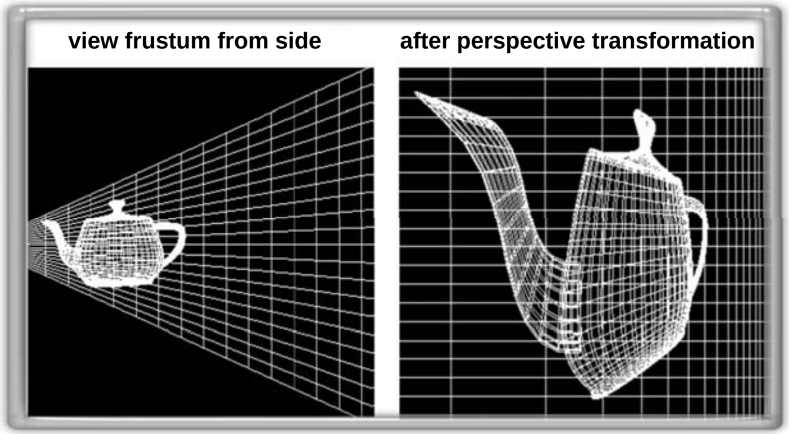


Perspective Warping

- Generate shadow maps in the post-perspective space.
- Increase sampling densities for near objects.

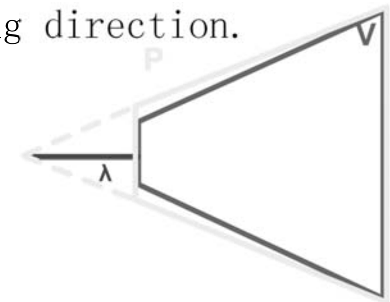


Perspective Warping



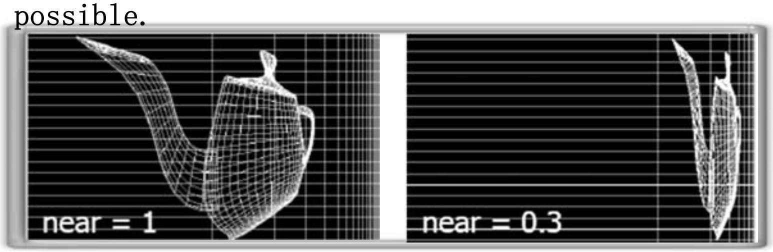
Freedoms of Warping Frustum

- The essential difference among variant perspective parameterizations is the selection of near plane.
- Warping direction.



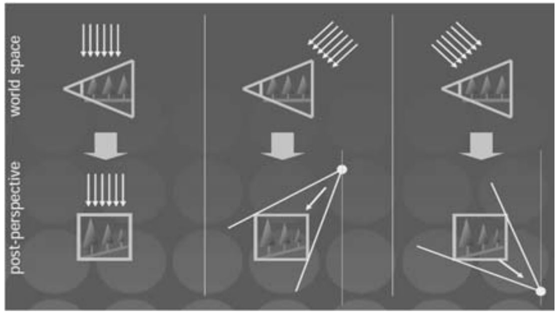
Near Plane Selection

- Near plane of the warping frustum determines how strong the warping effect is.
- Notice that stand shadow map also can be thought of as a special perspective parameterization! Where the near plane is set at infinity.
- Too small near plane will cause an over-strong warping effect. In practice, enlarge near as possible.



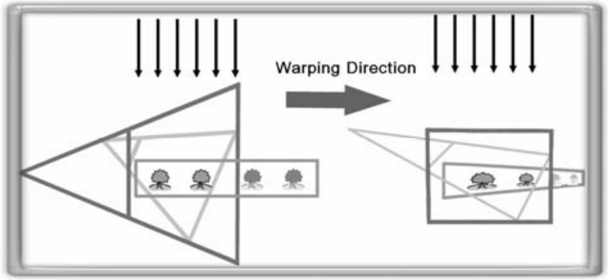
Warping Direction

- It' s important! The type of lighting source will be frequently changed after perspective warping!
- The frequently changed light types also result in the mapping singularities.



Warping Direction

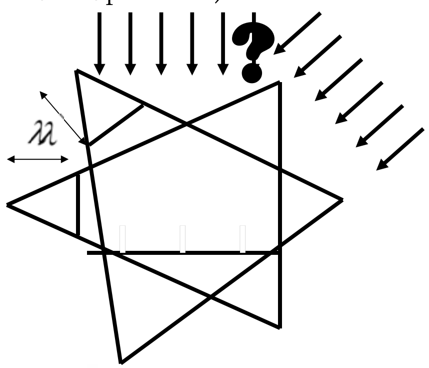
- A smart selection of the warping direction was proposed in [Wimmer, M. et al. 2004]. The warping direction is set to parallel to the shadow map plane.
- All types of lighting sources are converted to directional ones. No mapping singularities produced!



Problems (1)

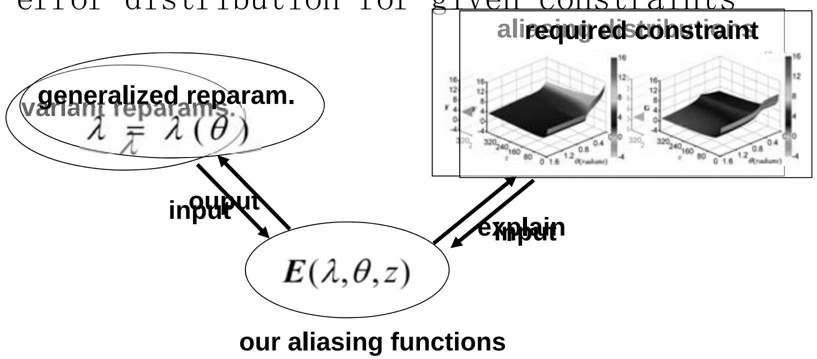
- Optimal distribution of aliasing errors is achieved in ideal case only
  - previous work $\lambda = \text{constant}$ (or not direction-optimal)

Stabilizing optimal aliasing distribution requires direction-adaptive $\lambda(\theta)$



Our Goals

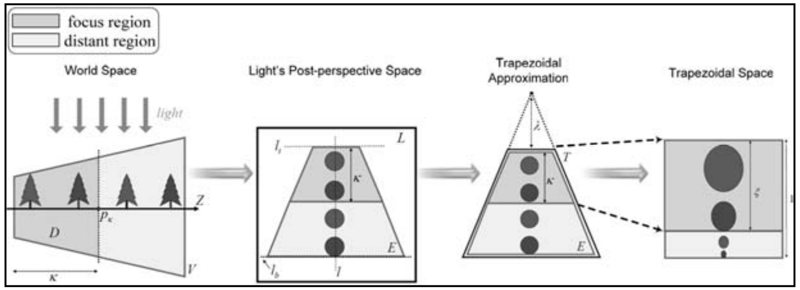
- Re-explain existing persp. reparams.
- Develop algorithms that stabilize optimal error distribution for given constraints



FTSM

Trapezoidal Shadow Maps (TSMs)

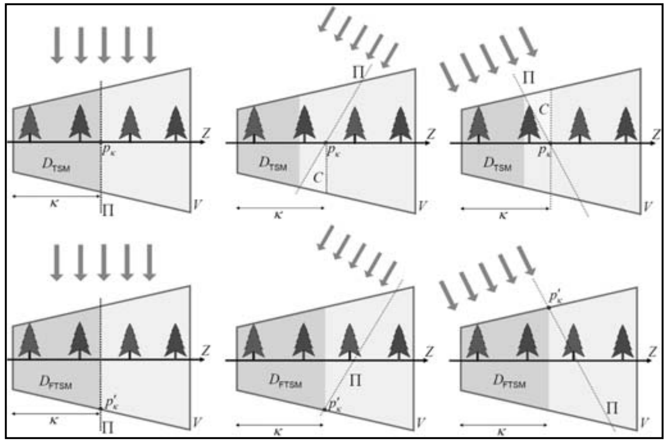
- **use a trapezoid to approximate the view frustum as seen from light.**
- **map the first 50% depth range to 80% fraction of shadow map.**



FTSM

Problem 1 of TSM

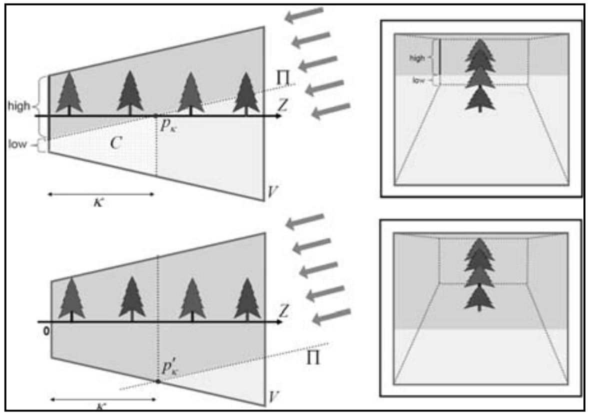
- **The focus region is not preserved in general case.**



FTSM

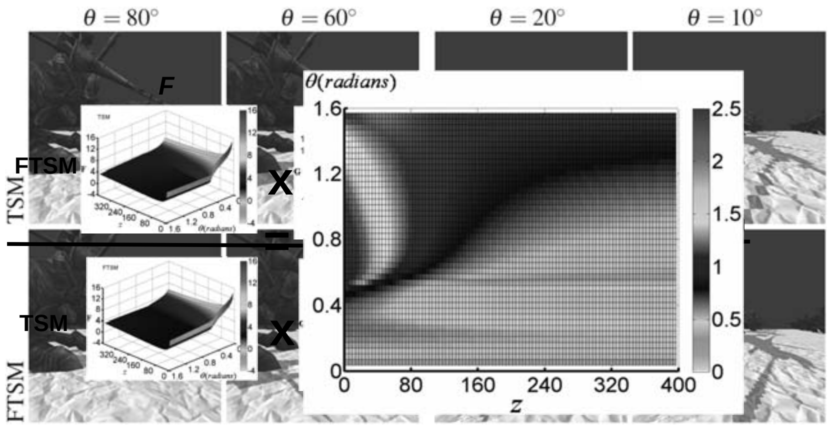
Problem 2 of TSM

- **unexpected stretching on the foreground**

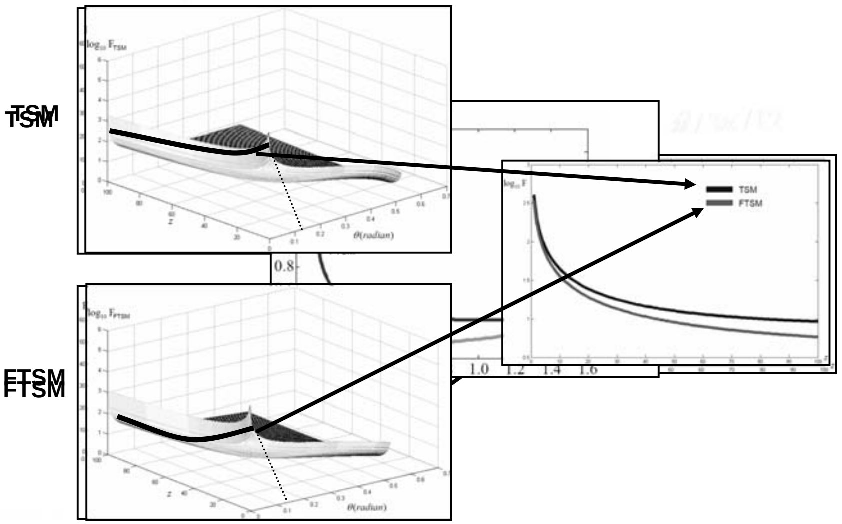


TSM vs. FTSM

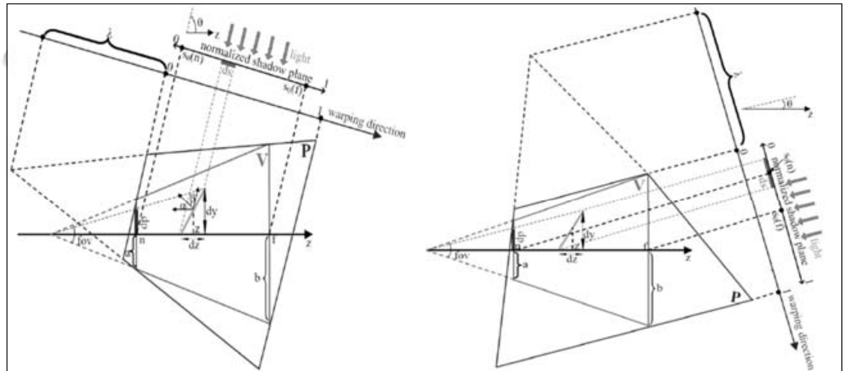
- Aliasing distributions
  - Both x-direction and z-direction are considered



TSM vs. FTSM



Direction-dependent Parameterization



(a) $|\theta| \in [fov/2, \pi/2]$ (b) $|\theta| \in [0, fov/2]$

Generalized Linear Parameterization

- Linear aliasing distribution.
  - $F = k$ (k is constant)
- Linear distribution requires $\varphi(\lambda, \theta) = 0$

$$\lambda = \begin{cases} \lambda_1(\theta) = \frac{n - \frac{a}{\tan\theta}}{f - n + \frac{a+b}{\tan\theta}} & |\theta| \in [\frac{fov}{2}, \frac{\pi}{2}] \\ \lambda_2(\theta) = \frac{f\tan\theta - b}{2b} & |\theta| \in [0, \frac{fov}{2}] \end{cases}$$

- Our selection depends on θ rather a constant used in prior work.

Generalized Linear Parameterization

- $\lambda_1 \geq 0$ and $\lambda_2 \leq 0$
  - The near plane should be positive!
  - The linear condition can NOT be satisfied in all cases!
  - The field-of-view is usually not very wide, so we can say that the linear parameterization can be achieved in MOST general cases.

Generalized Linear Parameterization

- How to modify the equation for coding?
  - We need to determine a constant γ and a new function $\lambda_2^*(\theta) \geq 0$ when $0 \leq \theta \leq \gamma$.
- Why dose γ appear?
  - Notice that $\lambda_1(f\omega=0)=0$ which will cause all pixels mapped into single texel! The warping effect is too strong, no any sense in practice.

$$\lambda_{\mathbf{GLPR}} = \begin{cases} \lambda_1(\theta) = \frac{n - \frac{a}{\tan\theta}}{f - n + \frac{a+b}{\tan\theta}} & |\theta| \in [\gamma, \frac{\pi}{2}] \\ \lambda_2^*(\theta) & |\theta| \in [0, \gamma] \end{cases}$$

Generalized Linear Parameterization

- How to select the function $\lambda_2^*(\theta)$?
  - Continuous λ transition at γ.

$$\boxed{\lambda_2^*(\gamma) = \lambda_1(\gamma)}$$

    - To keep the consistent transition of shadow qualities, continuous λ transition at θ=γshould be guaranteed.
  - GLPR converges to SSM as θ goes to 0.

$$\boxed{\lim_{\theta \to 0} \lambda_{\text{GLPR}}(\theta) = \lim_{\theta \to 0} \lambda_2^*(\theta) = \infty}$$

    - All perspective parameterizations degrade to standard shadow maps as the light direction goes to parallel to the view direction.

Generalized Linear Parameterization

- In our system, the following function that satisfies the previous two criteria.

$$\lambda_2^*(\theta) = \lambda_1(\gamma)/\sin(\frac{\theta}{2\gamma}\pi) = \frac{n - \frac{a}{\tan(\gamma}}{f - n + \frac{a+b}{\tan(\gamma)}} \frac{1}{\sin(\frac{\theta}{2\gamma}\pi)}$$

- The selection λ for GLPR

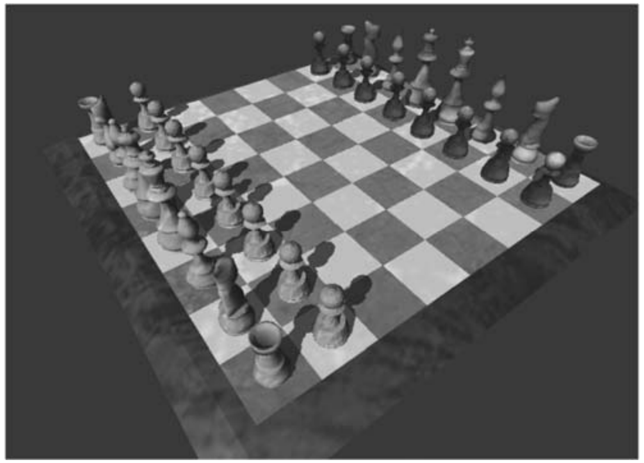
$$\lambda_{\mathbf{GLPR}} = \begin{cases} \lambda_1(\theta) = \frac{n - \frac{a}{\tan\theta}}{f - n + \frac{a+b}{\tan\theta}} & |\theta| \in [\gamma, \frac{\pi}{2}] \\ \lambda_2^*(\theta) = \lambda_1(\gamma)/\sin(\frac{\theta}{2\gamma}\pi) & |\theta| \in [0, \gamma] \end{cases}$$

Generalized Linear Parameterization

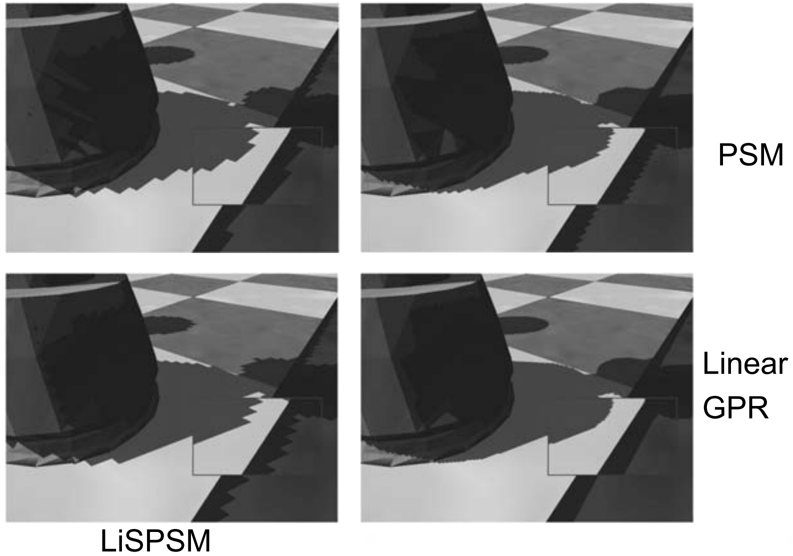
- How to determine γ?
  - Make sure $\lambda(\theta)$ is not very small.
    - why? An over strong warping will be caused by a too small near plane selection. In practice, we'd better let λ be larger as possible.
  - In our current implementation, we select a γ satisfying

$$\lambda(\gamma) \geq 0.7 \frac{n}{f-n}$$

Virtual Chess-board Scene



Visual Quality



Conclusion

- Shadow rendering techniques
  - Shadow volume and Shadow mapping
- Shadow mapping aliasing errors
- Problems for current shadow mapping algorithms
  - Treat all objects in the same manner
  - Huge memory requirement in some cases
  - Need more precise bounding shape

Conclusion

- Parallel-Split Shadow Maps
  - Regard the scene as depth layers
  - Render shadow maps for split parts
- Advantages
  - Suitable for large-scale virtual environments
  - More compact bounding shape
  - Less texture memory requirement
  - A "layered" framework easily integrated with other shadow mapping algorithms

Conclusion

- We have formulized the Generalized Linear Perspective Reparameterization GLPR in shadow mapping

Future work

- Analysis for distribution of aliasing errors along arbitrary warping direction and light sources
- Investigate effective integration with other shadow mapping algorithms

Q & A