Rectilinear Texture Warping for Fast Adaptive Shadow Mapping

Patrick Jansen & Dion Gerritzen
Content of this presentation

- Recap: Shadow Maps
  - Issues
  - Adaptive Shadow Mapping
- Rectilinear Texture Warping
  - Importance map
    - Importance functions
  - Warping maps
  - Shadow map
- Extra features
- Conclusion
Recap
Shadow Maps

Shadow Maps

Render scene depth from light source
- Store in shadow map
- Compare distance from light source with shadow map depth

Shadowed:
\[ \text{Shadow map depth} < \text{distance}(\text{point, light}) \]

Unshadowed:
\[ \text{Shadow map depth} = \text{distance}(\text{point, light}) \]
Recap
Shadow Maps: Issues

- Aliasing:

- High detail shadow maps require huge amounts of memory
Recap

Adaptive Shadow Mapping

- ASM (Fernando et al. 2001) removes aliasing by storing the shadow map as a hierarchical grid structure
- Requires multiple passes on scene geometry in order to generate all of the nodes of the hierarchy
Rectilinear Texture Warping

- New approach to Adaptive Shadow Mapping
  - Adaptive to view, scene and lighting
  - Uses conventional textures rather than hierarchy
- Same resolution, higher quality
  - Approaches quality of raytraced shadows
Rectilinear Texture Warping
Rectilinear Texture Warping

Algorithm

**Algorithm:** RTW Shadow Mapping

(a) Build the importance map
(b) Convert 2-D importance map into 1-D warping maps
   (i) Collapse rows/columns to 1-D importance maps
   (ii) Blur importance maps
   (iii) Build warping maps from importance maps
(c) Render the RTW shadow map
(d) Render the output image from the desired view
Rectilinear Texture Warping

Algorithm - (a) Importance Map

- Assign *importance values* to texels
- Three approaches:
  - Forward Analysis (FA)
  - Backward Analysis (BA)
  - Hybrid Analysis (HA)
Rectilinear Texture Warping
Algorithm - (a) Importance Map

- Assign *importance values* to texels
- Three approaches:
  - Forward Analysis (FA)
    - Create depth image from light's perspective
    - Collect importance values from depth image
  - Backward Analysis (BA)
  - Hybrid Analysis (HA)
Rectilinear Texture Warping

Algorithm - (a) Importance Map

- Assign *importance values* to texels
- Three approaches:
  - Forward Analysis (FA)
  - Backward Analysis (BA)
    - Render output image (without shadows)
    - Project output samples onto light's image space
    - Analyse importance from projected samples
  - Hybrid Analysis (HA)
Rectilinear Texture Warping

Algorithm - (a) Importance Map

- Assign *importance values* to texels
- Three approaches:
  - Forward Analysis (FA)
  - Backward Analysis (BA)
  - Hybrid Analysis (HA)
  - Combination of forward and backward analysis
  - Gives most flexibility
  - Costs more to compute
Rectilinear Texture Warping
Algorithm - (a) Importance Functions

- Apply any set of analytic or heuristic functions
- Paper describes four:
  - Desired view function
  - Distance to eye function
  - Shadow edge function
  - Surface normal function
Rectilinear Texture Warping
Algorithm - (a) Importance Functions

● Apply any set of analytic or heuristic functions
● Paper describes four:
  ○ Desired view function
    ■ Only calculate shadows within the desired view
    ■ Return 1 for any point within the view
    ■ Return 0 otherwise
    ■ Potentially bad results with FA
  ○ Distance to eye function
  ○ Shadow edge function
  ○ Surface normal function
Rectilinear Texture Warping
Algorithm - (a) Importance Functions

- Apply any set of analytic or heuristic functions
- Paper describes four:
  - Desired view function
  - Distance to eye function
    - Regions farther from the desired view require less detail
    - FA: transform texel from light space to view space
    - BA: use depth value directly
    - Output in the range [0,2]
  - Shadow edge function
  - Surface normal function
Rectilinear Texture Warping
Algorithm - (a) Importance Functions

- Apply any set of analytic or heuristic functions
- Paper describes four:
  - Desired view function
  - Distance to eye function
  - Shadow edge function
    - Increase sample rate for regions which contain shadow edges
    - Depth test on each texel and his eight neighbours
    - Edges receive full importance (1), other regions receive minimal importance (0.001)
    - Only works with FA
  - Surface normal function
Rectilinear Texture Warping
Algorithm - (a) Importance Functions

- Apply any set of analytic or heuristic functions
- Paper describes four:
  - Desired view function
  - Distance to eye function
  - Shadow edge function
  - Surface normal function
    - Visibility of surfaces
    - Dot product between surface normal and view direction
    - Works for all analysis methods
Rectilinear Texture Warping

Algorithm - (b.i) Warping Map

- Generate 1D Warping Maps by taking the max importance value for each row and column of the Importance Map.
Rectilinear Texture Warping
Algorithm - (b.i) Warping Map
Rectilinear Texture Warping
Algorithm - (b.ii) Warping Map

Gaussian blur

Produces a smooth transition in sampling rate

Helps reduce non-linear rasterisation errors by reducing amount of tessellation required
Rectilinear Texture Warping
Algorithm - (b.iii) Warping Map

Gaussian blur

1D warp map
Rectilinear Texture Warping
Algorithm - (b.iii) Warping Map

- Build the (1D) warp map
  - Segment map into *super-cells*
    - at equal to or lower than original resolution
  - Each super-cell has an importance value from the 1D importance map
Rectilinear Texture Warping

Algorithm - (b.iii) Warping Map

- Build the (1D) warp map
  - Displace super-cells according to the equation

\[
\left( \sum_{j=1}^{k-1} \frac{I_j}{\sum_{j=1}^{n} I_j} \right) - \left( \frac{k}{n} \right)
\]

- \( k \) = current super-cell
- \( I \) = 1D importance map
- \( n \) = total number of super-cells
Rectilinear Texture Warping

Algorithm - (b.iii) Warping Map

- Build the (1D) warp map
  - Displace super-cells
    - This increases sampling rate in important areas
Rectilinear Texture Warping

Algorithm - (c) Render Shadow Map

- Render the Shadow Map using the Warp Map
  - Project all vertices to their respective position on the conventional shadow map
  - Use the projected location to locate the vertical and horizontal super-cells
  - Relocate the vertices to the output plane by applying their respective warp
Rectilinear Texture Warping

Algorithm - (d) Render Output Image

- **Forward Analysis:**
  - Scene geometry must be drawn from camera view
  - This time with shadows from RTW shadow map

- **Backward (and Hybrid) Analysis:**
  - Output frame has already been computed
  - Calculate shadows and compose on top of image

- **Shadow calculation:**
  - Calculate conventional shadow map coordinates
  - Warp coordinates using Warp Maps
Extra Features
Tessellation

- Warping can cause curved triangle edges
- Solved by tessellation
  - Subdivide edges larger than $\epsilon$
  - $\epsilon = 2.5\%$ of shadow map width gives a good balance between performance and image quality
- Increasing the Gaussian blur of importance map reduces need for tessellation
Extra Features
Percentage Closer Filtering

- Can be applied for two purposes
  - Enhance shadow quality by antialiasing shadow edges
  - Simulate soft shadowing effects
- Pass warped coordinates to PCF
## Conclusion

### Pros & Cons

<table>
<thead>
<tr>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td>● Higher quality shadow with same resolution shadow map</td>
<td>● FA: Details lost with too low resolution of depth image</td>
</tr>
<tr>
<td>● No shimmering</td>
<td>● BA: Different points may project to same importance map location</td>
</tr>
<tr>
<td>● Variation in sampling rate appears seamless</td>
<td>● Slower than conventional method</td>
</tr>
<tr>
<td>● Suitable for interactive purposes</td>
<td></td>
</tr>
</tbody>
</table>
Conclusion

Video :D
Conclusion

Discussion

● Question for the audience:
  ○ Thoughts on suitability for use in games?
Conclusion

Discussion

● Question for the audience:
  ○ Thoughts on suitability for use in games?

Any questions?