Space-Time Hierarchical Occlusion Culling for Micropolygon Rendering with Motion Blur

Solomon Boulos, Edward Luong, Kayvon Fatahalian, Henry Moreton, Pat Hanrahan
HPG 2010
High interest in Motion Blur

- [Akenine-Moller et al. EGSR 2007]
- [Fatahalian et al. HPG 2009]
- Three more papers here

- This paper: occlusion with motion blur
Figure 12. Frame from Luxo Jr.

Figure 13. Shadow maps from Luxo Jr.

Figure 14. Red's Dream

[Luxo Jr., Pixar 1986]
Figure 12. Frame from Luxo Jr.

Figure 13. Shadow maps from Luxo Jr.

Figure 14. Red’s Dream

[Red’s Dream, Pixar 1987]
Culling: Cost/Benefit Tradeoff

Input → Split → Dice/Disp → Shade → Rast
Culling: Cost/Benefit Tradeoff
Culling: Cost/Benefit Tradeoff

Input ➔ Split ➔ Dice/Disp ➔ Shade ➔ Rast
Culling: Cost/Benefit Tradeoff

Input → Split → Dice/Disp → Shade → Rast
Culling: Cost/Benefit Tradeoff

Input → Split → Dice/Disp → Shade → Rast
Contributions

• TZ-Pyramid: Data structure for efficient hierarchical occlusion culling with motion blur

• Analysis: where to use the tz-pyramid in a micropolygon pipeline to optimize the cost/benefit tradeoff
Background
(Occlusion culling without motion blur)
Figure 3.4 A scene and its corresponding z-pyramid. The finest level of the pyramid is the ordinary z-buffer. At all other levels, each z sample is the farthest z from the observer in the corresponding $2 \times 2$ window of the next finer level. Every entry in the pyramid therefore represents the farthest z for a square region of the screen.

[Greene et al. 1993]
Z-pyramid Update

Multi-sample

zNear  zFar
Z-pyramid Update

Multi-sample

Level 1

zNear  zFar
Z-pyramid Update

Multi-sample → Level 1 → Level 2

zNear → zFar
Z-pyramid Update

Multi-sample

Level 1

Level 2

zNear

zFar
Z-pyramid Update

Multi-sample ➔ Level 1 ➔ Level 2

zNear ➔ zFar
Z-pyramid Culling

zNear  zFar
Z-pyramid Culling
Z-pyramid Culling
Z-pyramid as Acceleration Structure

zNear  zFar
Optimization: Traversal Initialization

zNear zFar
Motion Blur
Switch to Space-Time

zNear  zFar
Switch to Space-Time

![Diagram showing a 2x2 grid with time steps from $t_0$ to $t_3$ and labels for zNear and zFar.]
Idea 1: Reuse the current z-pyramid

Multi-sample

Level 1

Level 2

zNear

zFar
Idea 1: Reuse the current z-pyramid

Multi-sample

Level 1

Level 2

zNear

zFar
Problem: Too Conservative
Problem: Too Conservative

Solid Occluder
Idea 2: z-pyramid per time

Multi-sample

Level 1

Level 2

zNear

zFar
TZ-Slice: More effective
TZ-Slice: More expensive
TZ-Slice: More expensive
“Moving” depends on scale
“Moving” depends on scale
“Moving” depends on scale
TZ-Pyramid

Time Level 1

<table>
<thead>
<tr>
<th>t₀</th>
<th>t₀</th>
</tr>
</thead>
<tbody>
<tr>
<td>t₀</td>
<td>t₀</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>t₁</th>
<th>t₁</th>
</tr>
</thead>
<tbody>
<tr>
<td>t₁</td>
<td>t₁</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>t₂</th>
<th>t₂</th>
</tr>
</thead>
<tbody>
<tr>
<td>t₂</td>
<td>t₂</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>t₃</th>
<th>t₃</th>
</tr>
</thead>
<tbody>
<tr>
<td>t₃</td>
<td>t₃</td>
</tr>
</tbody>
</table>
TZ-Pyramid

Time Level 1

Time Level 2

zNear

zFar
TZ-Pyramid

Time Level 1

t₀ | t₁ | t₂ | t₃
TZ-Pyramid

Time Level 1

- $t_0$
- $t_1$
- $t_2$
- $t_3$

Time Level 2

- $[t_0, t_1]$
- $[t_2, t_3]$

zFar

zNear
TZ-Pyramid

Time Level 1

\[ t_0 \]
\[ t_1 \]
\[ t_2 \]
\[ t_3 \]

Time Level 2

\[ [t_0, t_1] \]
\[ [t_2, t_3] \]

Time Level 3

\[ [t_0, t_3] \]
Space-Time Recursive Update

Time Level 1

Time Level 2

Time Level 3

zFar

zNear
Space-Time Recursive Update

Time Level 1

Time Level 2

Time Level 3
Space-Time Recursive Update

Time Level 1

\[ t_0 \quad t_0 \quad t_1 \quad t_1 \]

\[ t_0 \quad t_0 \quad t_1 \quad t_1 \]

Time Level 2

\[ [t_0, t_1] \quad [t_0', t_1] \quad [t_0, t_1] \quad [t_0', t_1] \]

\[ [t_0, t_1] \quad [t_0', t_1] \quad [t_0, t_1] \quad [t_0', t_1] \]

Time Level 3

\[ [t_0, t_3] \quad [t_0', t_3] \quad [t_0, t_3] \quad [t_0', t_3] \]

\[ [t_0, t_3] \quad [t_0', t_3] \quad [t_0, t_3] \quad [t_0', t_3] \]
Space-Time Recursive Update

Time Level 1

\[ t_0 \quad t_1 \quad t_2 \quad t_3 \]

Time Level 2

\[ [t_0, t_1] \quad [t_2, t_3] \]

Time Level 3

\[ [t_0, t_3] \]

zFar

zNear
Space-Time Recursive Update

Time Level 1
- $t_0$
- $t_1$
- $t_2$
- $t_3$

Time Level 2
- $[t_0, t_1]$
- $[t_2, t_3]$

Time Level 3
- $[t_0, t_3]$
Space-Time Recursive Update

Time Level 1
- $t_0$
- $t_1$
- $t_2$
- $t_3$

Time Level 2
- $[t_0, t_1]$
- $[t_2, t_3]$

Time Level 3
- $[t_0, t_3]$
Space-Time Recursive Update

Time Level 1
- \( t_0 \)
- \( t_1 \)
- \( t_2 \)
- \( t_3 \)

Time Level 2
- \([t_0, t_1]\)
- \([t_2, t_3]\)

Time Level 3
- \([t_0, t_3]\)
TZ-Pyramid Culling
TZ-Pyramid Culling
TZ-Pyramid Culling

Diagram showing TZ-Pyramid Culling with various time points and intervals.
TZ-Pyramid Culling
TZ-Pyramid Culling
TZ-Pyramid Culling
Evaluation

STICKS

ARMY

ZINKIA

Rendered at 1080p, 16 samples per pixel, 2x2 pixel interleave
Evaluation: Metrics of Interest

- Diced vertices
- Shaded vertices

- Depth comparisons
  - Coarse: Inner nodes
  - Fine: Multi-sample z
When to Cull

Input -> Split -> Dice/Disp -> Shade -> Rast
No Occlusion Culling

Pipeline Costs:
- 7.85 Million
- 36.8 Million
- 22.3 Million
- 128.1 Million

Culling Cost:
- 1.0
- 0.75
- 0.50
- 0.25
- 0

Input → Split → Dice/Disp → Shade → Rast
Just prior to shading
Just prior to dicing

Culling Cost

Pipeline Costs

Input ➔ Split ➔ Dice/Disp ➔ Shade ➔ Rast
Both before dicing and shading

Culling Cost

Pipeline Costs

Input → Split → Dice/Disp → Shade → Rast
Cull at all stages

**Culling Cost**

- 1.0
- 0.75
- 0.50
- 0.25
- 0

**Pipeline Costs**

- Input
- Split
- Dice/Disp
- Shade
- Rast

Diagram showing the culling cost and pipeline costs across different stages.
Resolution Tradeoff: Prefer Temporal

diced points

shaded points

z reads
TZ-Pyramid: More efficient

Graphs showing the comparison between tz-slice and tz-pyramid for Coarse Depth Reads and Total Depth Reads against Pixels of Motion.
Recap

• TZ-Pyramid: extension of z-pyramid for motion blur
  • More effective than reusing z-pyramid
  • More efficient than TZ-slice
  • Manageable footprint
• Culling at all stages works best
  • Pays for itself
  • Culling earlier increases benefit while reducing cost
Next Steps

• Traditional optimizations
  • Compression, Resolution Tradeoffs, Fixed-Function
• More applications
• Future pipeline integration
Thanks

- National Science Foundation Graduate Research Fellowship
- Intel Larrabee Research Grant
- Stanford Pervasive Parallelism Laboratory
  - Oracle, NVIDIA, IBM, NEC, AMD, Intel
- Kurt Akeley, Margarita Bratkova, James Hegarty, Mike Houston, Bill Mark, Jonathan Ragan-Kelley.
- Zinkia Entertainment, S.A.