State of the Art in Interactive Ray Tracing

Peter Shirley (Utah)
Solomon Boulos (Utah)
Gordon Stoll (Intel)
Dinesh Manocha (UNC)
Christian Lauterback (UNC)
Ingo Wald (Utah)
Abe Stephens (Utah)
Sungeui Yoon
Bill Mark (Texas)
Philipp Slusallek (Saarland)
Holgar Schmidt (EADS)
Course overview

What is going on this year in ray tracing (most of course) and what we think the big open problems are.
Why now?

- 1979, Turner Whitted, 74 minutes at 640x480
- 18 Moore's Law CPU doublings in 27 years
- 12 doublings 74min → 1 sec
- We should now have 1024x1024 at 16 f.p.s.
Why now? (2)

- RT is naturally sublinear with scene size
- RT naturally supports good shading
- RT maps well to multi-core architectures
- If Moore's law continues
  - 9 years $\rightarrow$ $2048^2$ with 16 samples per pixel
Moore’s Law and Screens

- Moore’s law: about 100x every decade
- Moore’s law per pixel: 50x every decade
First “real” IRT

Mike Muuss 1995
Some terminology

- Spatial/object subdivision
- Animated model
- SIMD
- Ray packet/bundle
- Multicore
- Whitted-style ray tracing
- Cook-style ray tracing
Animated models

- General (hard)
- Interactive modification (easier)
- Deformable (easier still)
- Articulated (easiest)
SIMD

- Several identical operations done at once
- SSE is the common example

```c
const __m128 uvmask4 = _mm_cmple_ps(_mm_mul_ps(uplusv4, det4),
                                        _mm_mul_ps(det4, det4));
cost __m128 voverd4 = _mm_mul_ps(rcp4, v4);
cost __m128 vmask4 = _mm_cmpge_ps(voverd4, zeroes4);
mask4 = _mm_and_ps(mask4, uvmask4);
mask4 = _mm_and_ps(mask4, vmask4);
```
Ray Packets (Bundles)

- A list of rays
- Some may be inactive
- Some spatial/directional coherence implied
- Useful both for SIMD and non-SIMD
Multicore

- Several CPUs on one chip (or board)
- Most systems are now multicore
Spatial subdivision
Object subdivision
Whitted-Style RT

- About one ray per pixel
- Hard shadows
- Sharp reflections
- Pinhole camera
Whitted advantages

- Caching works
- Many CPUs work
- Few samples per pixels needed
Cook-Style RT

- Tens of rays per pixel (no branching)
- Soft shadows
- Fuzzy reflections
- Depth-of-field
Hybrid

- Use Cook just for some effects with branching
- Common for shadows
Each ray has

- pixel position
- lens position
- light position
- time
Branching

- Typically, there is little or none
- Each ray creates a path (not a tree)
Stratification
Some open questions

- Which of BVH vs k-d vs grid?
- What are the target scenes/applications?
- What hardware will we use?
- Adaptive or constant sampling?
- Interface to batch renderers?
- Procedural geometry?
- API?
Academics’ dilemma

The number of papers “mine-able” is proportional to the awkwardness of approach.

Interactive Ray Tracing 2006, next month (Poster abstracts due Aug 6)
<table>
<thead>
<tr>
<th>Time</th>
<th>Session</th>
<th>Presenter</th>
</tr>
</thead>
<tbody>
<tr>
<td>8:30-9:15</td>
<td>Intro</td>
<td>Shirley, Boulos</td>
</tr>
<tr>
<td>9:15-10:15</td>
<td>Optimizing RT</td>
<td>Stoll</td>
</tr>
<tr>
<td>10:30-11:00</td>
<td>Dynamic Environments</td>
<td>Manocha, Lauterback</td>
</tr>
<tr>
<td>11:00-noon</td>
<td>Dyn. Models</td>
<td>Wald</td>
</tr>
<tr>
<td>noon-12:15</td>
<td>Cook RT</td>
<td>Boulos</td>
</tr>
<tr>
<td>Time</td>
<td>Session</td>
<td>Speaker(s)</td>
</tr>
<tr>
<td>--------------</td>
<td>----------------------------------</td>
<td>----------------------</td>
</tr>
<tr>
<td>1:30-1:50</td>
<td>Big iron</td>
<td>Stephens</td>
</tr>
<tr>
<td>1:50-2:30</td>
<td>LODs &amp; Data layout</td>
<td>Manocha, Yoon</td>
</tr>
<tr>
<td>2:30-3:30</td>
<td>Razor</td>
<td>Stoll, Mark</td>
</tr>
<tr>
<td>Break</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3:45-4:45</td>
<td>Architectures</td>
<td>Slusallek Schmidt</td>
</tr>
<tr>
<td>4:45-5:30</td>
<td>Open prob's</td>
<td>All</td>
</tr>
</tbody>
</table>
Where are we now?
(demo: Solomon Boulos)
Questions?