Real-Time Realism will require...

Tomas Akenine-Möller
Lund University and Intel Corporation

Panel at High-Performance Graphics 2010
2010-06-27

Contents

- Talk about differences/similarities between ray tracing and rasterization
- Then argue that there is lots of work to be done in visibility research...
- ...to reach real-time realism

- Disclaimer: note that a rather sloppy O-notation is used in these slides.

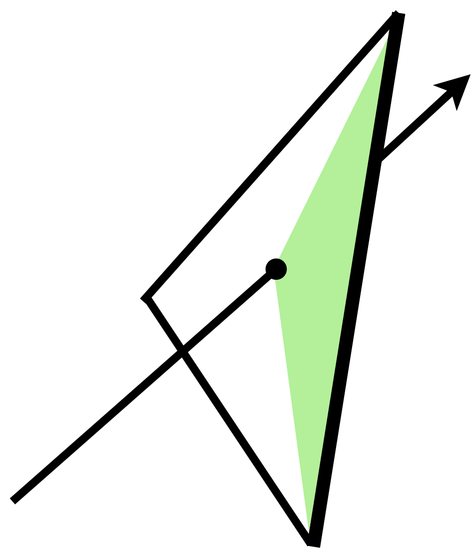
  A short paper will be written based on these slides during 2010.

Ray Tracing vs Rasterization?

- Many similarities in how visibility is computed
  - Not really a big conceptual difference
- Explore that a bit here...

Ray/sample in triangle testing

- Ray/triangle intersection
  - Most tests are basically *signed volume* computations [Kensler & Shirley, IRT 2006]
  - Hanrahan did similar things in *homogeneous* coords

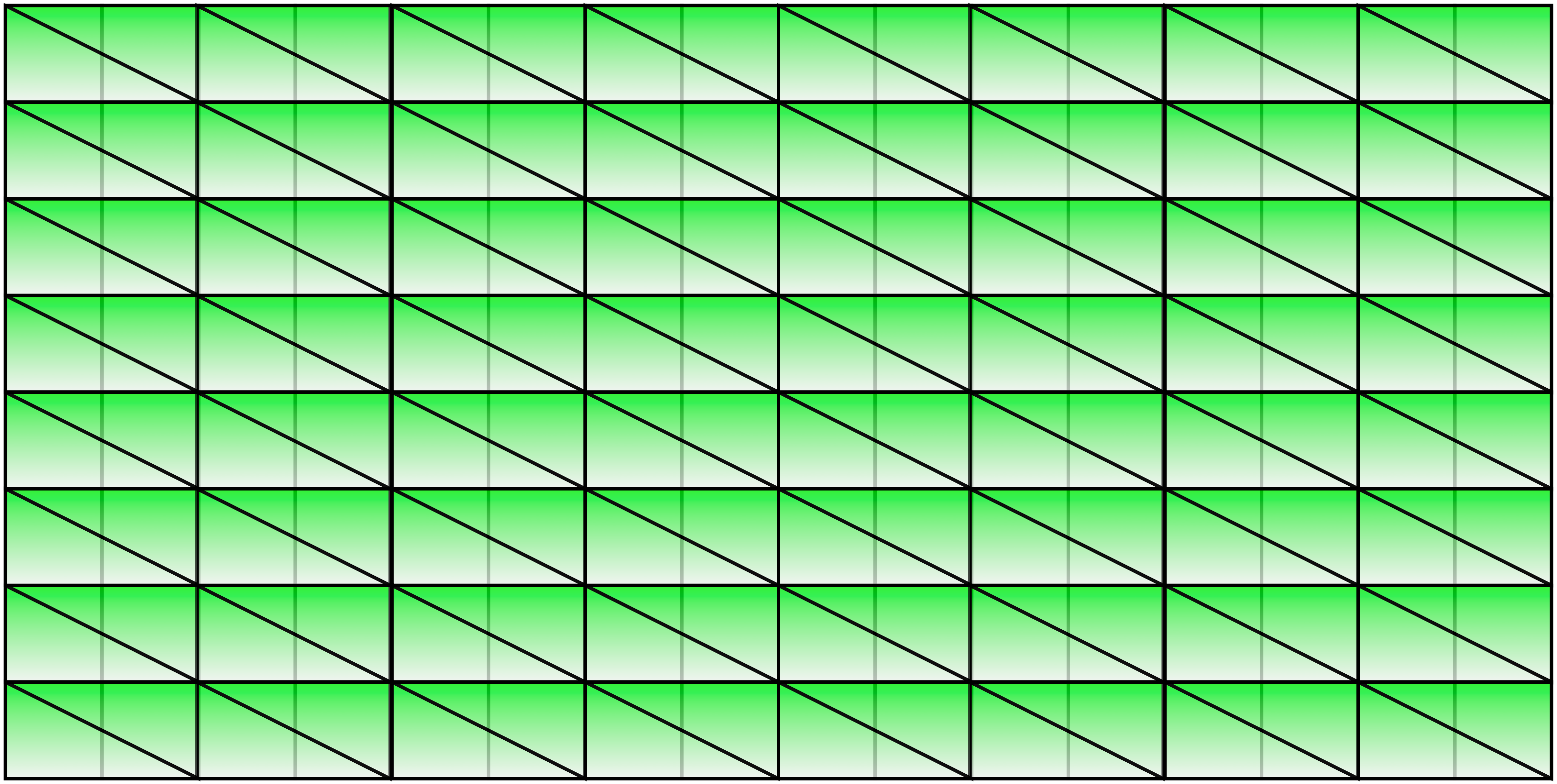


- Sample/triangle rasterization
  - Could use edge equations [Pineda88] or *homogeneous* edge equations [Olano & Greer 97]
  - Equivalent to testing if the sample is on the "right" side of the plane from the viewer through a triangle edge
  - That is, a *signed volume*

Complexity analysis

- The usual arguments for ray tracing:
  - Ray tracing is $O(\log n)$, while...
  - ...rasterization is $O(n)$
- Why is rasterization so successful for coherent rays?
  - The GPU? Not only...
- For *coherent* rays, the analysis above is not quite correct (I think).

Coherent rasterization complexity



- Approximately 1-2 triangles per pixel
- $O(\log n)$ for ray tracing per pixel
- $O(1)$ for rasterization per pixel

Rasterization complexity

- Hence, for coherent rays, $O(d)$ per pixel, where d is the depth complexity
- Wonderful paper by Cox & Hanrahan 1993, showed that overdraw is:

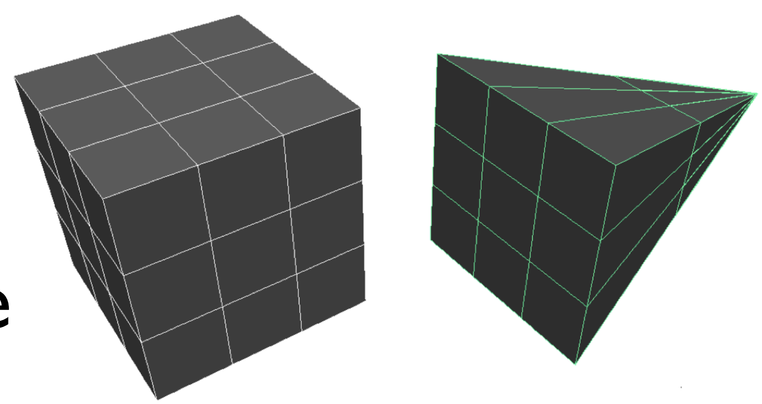
$$1 + \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \cdots + \frac{1}{d} \approx \ln(d) + \gamma$$

$$\gamma = 0.57721\ldots$$

- So, $O(\log d)$ for shading
  - But, game devs do rough front2back sort, actually better
- With shading after visibility (deferred): $\approx O(1)$ shading cost

More complexity

- In rasterizers, we use coarser BVH to cull outside frustum
  - Ray tracing is $\approx$ view frustum culling for 1 pixel
  - So rasterization complexity is (for arbitrary rays): $O(\log n + kd)$
    - Leaf node size >1 triangle, hence the k, where $k > 1$
  - Now, if you only want coarse (1st bounce GI) visibility, rasterization becomes interesting again
  - Plus, rasterization basically builds a projected uniform grid in camera space [Hunt & Mark 2008] using Zmax per tile



- Ray tracing can use small frusta around packets of rays as well
  - Ray tracing starts to resemble rasterization for coherent rays. See, for example, [Reshetov2008].

Similarities & Differences

	Ray tracing	*Rasterization*
Point/ray inside triangle	Signed volumes, i.e., ≈plane through edge dot ray	Homogeneous edge equations = "planes" through tri edges
Acceleration data structure	Yes, BVH/Kd-tree down to the *individual* triangles	Yes, BVH down to *groups* of triangles + builds (on-the-fly) uniform grid in projected space
Primary rays	$O(\log n)$ ...or a bit faster: packets	$O(d)$ ...or a bit faster: Zmax + occlusion queries
Secondary rays	$O(\log n)$ ...not counting ray-tri tests	$O(\log n+kd)$
(Shading)	$O(1)$	Could be $O(1)$ with "deferred," otherwise $O(\log d)$, or a little better

Possible conclusion

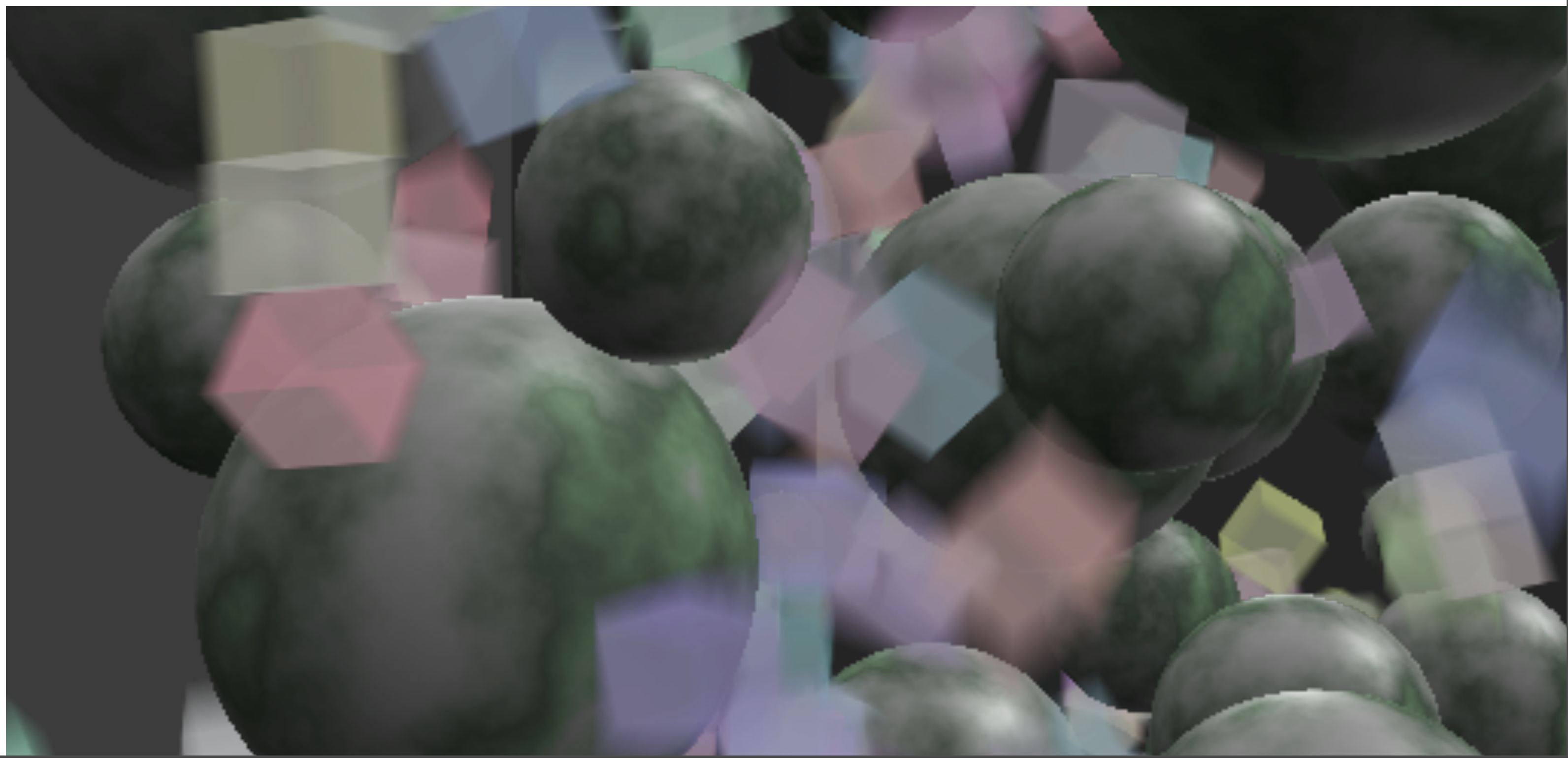
Ray tracing and rasterization are (“converging” to) the same visibility algorithm

(in a broad sense)

- Examples:
  - Micropolygon Ray Tracing with Defocus and Motion Blur by Hou et al., SIGGRAPH 2010
    - Ray tracing using BVH, then basically rasterization when you reach the leaves
  - Low-res hierarchical “rasterization” of indirect light
    - [Bunnell 2005], [Christensen 2008],[Ritschel et al., 2009]
    - [Kautz et al. 2004]
  - Coarser BVH, vertex-culling/ray tracing [Reshetov 2008]
- Likely, we will see new combinations soon

Lots of progress in visibility lately

- Stochastic rasterization is hot... again
  - [Akenine-Möller et al., GH 2007], [Toth & Linder, MSc thesis 2008], [Hou et al., SIGASIA 2009], [McGuire et al., HPG 2010]
- Decoupled shader caching
  - [Hasselgren & Akenine-Möller, EGSR 2006],[Ragan-Kelley et al. TOG 2010], [Burns & Fatahalian, HPG 2010]
- Analytical
  - Bandwidth/compute gap continues to grow, so might make more sense in the future
  - [Gribel et al. HPG2010]
- Combinations:
  - [Bunnell 2005], [Christensen 2008], [Ritschel et al., 2009]
  - [Hou et al. 2010]



What we want for real-time...

- Motion blur, depth of field, stereo, low-res GI, micropolygons, and more...
- So, I *think* that (near-term and good-enough) real-time realism will require a lot of:

Innovation in visibility algorithms

- ...but this is only *one* ingredient.
- Future of visibility may be a sweet-and-sour mix of rasterization, ray tracing, point sampling, and analytical visibility

Thanks for listening!

...and thanks to the Advanced Rendering Technology group at Intel for feedback!