Extended Raytracer

Drew Romanyk, Leslie Rice, Basil Hariri

Proposal

- Distribution (Monte-Carlo) Ray Tracing
  - Instead of deterministic secondary rays, use a distribution based on the BRDF
    - Similar in basic concept to anti-aliasing
  - Gives semi-diffuse reflections and refractions
  - Would require KD-trees to accelerate ray intersection
- Texture and Bump Mapping
  - Map texture, specularity, orientation (i.e. bumps), and transparency to triangles/spheres/anything that supports UV coordinates

KD-Trees

- Implemented a KD-tree ray-intersection accelerator
  - Allowed intersections to occur properly
  - Unfortunately, did not provide significant speedup
  - Time spent debugging eventually prevented us from successfully implementing distribution (Monte Carlo) ray tracing.

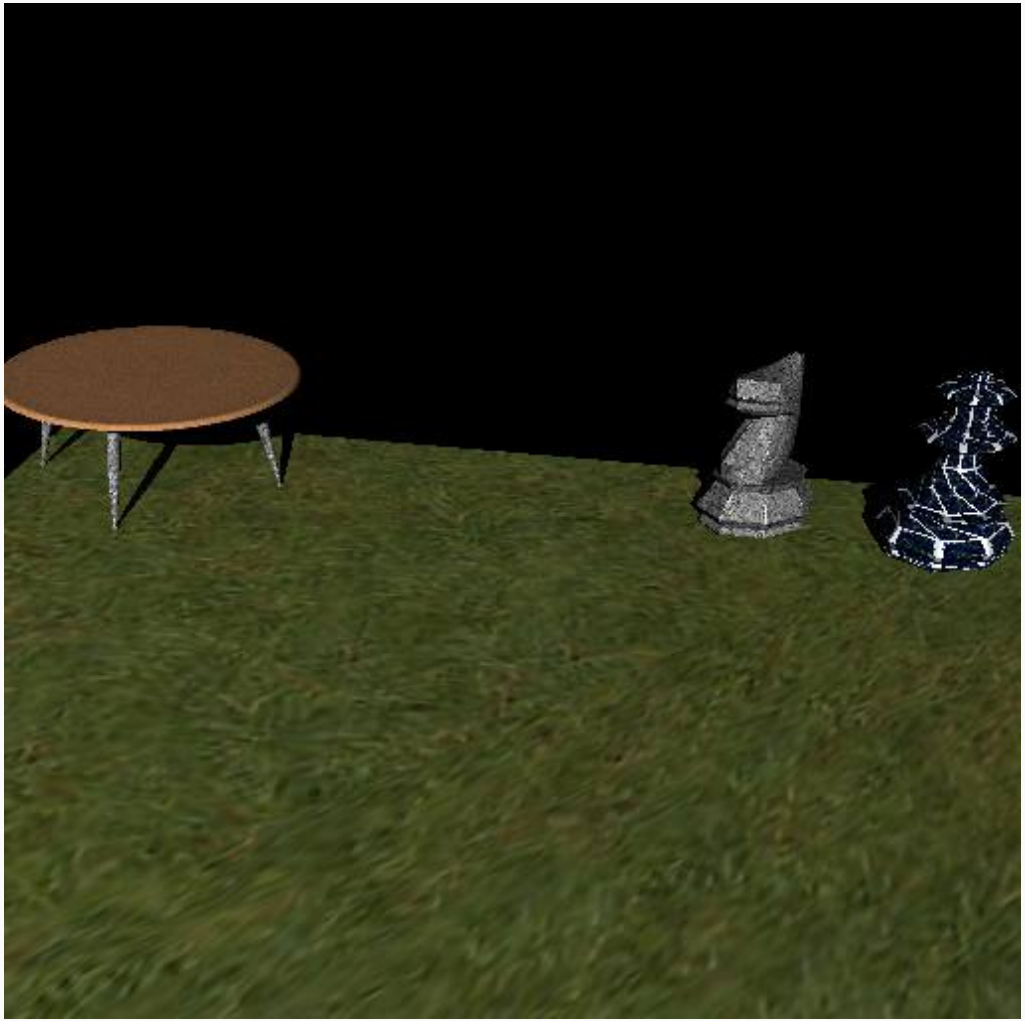
Multithreaded Tracing

- Decided to work on multithreaded raytracing to speed up the rendering
  - Hopeful that speed up would allow us to proceed with distribution ray tracing
  - 16 threads runs ~5x faster than single thread
    - With 1 thread, trimesh1 takes 12.50 seconds to render, with 16 it takes 2.97 seconds
  - Speed up was not enough to finish distribution ray tracing

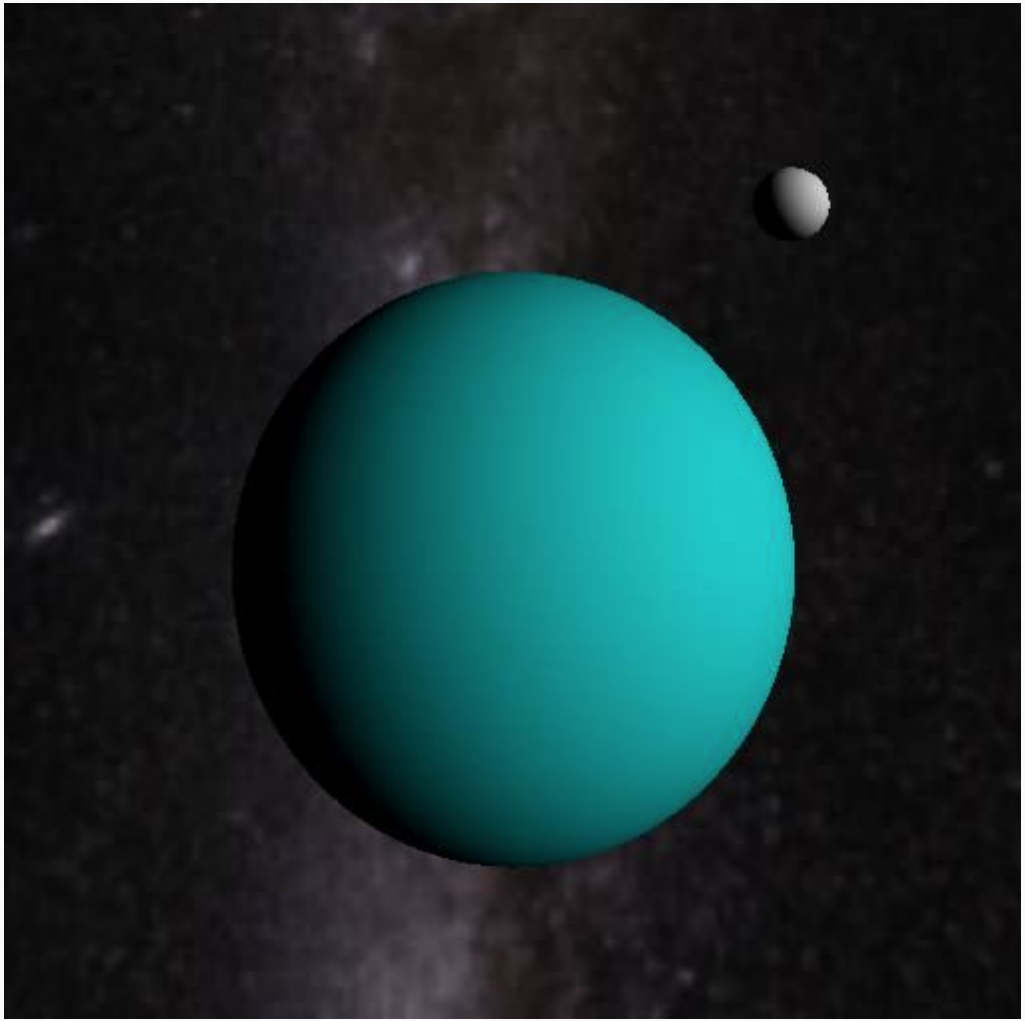
Original Ray Tracer

Texture Mapping

Implemented texture mapping on geometries such as spheres, trimeshes, and anything that supports UV mapping

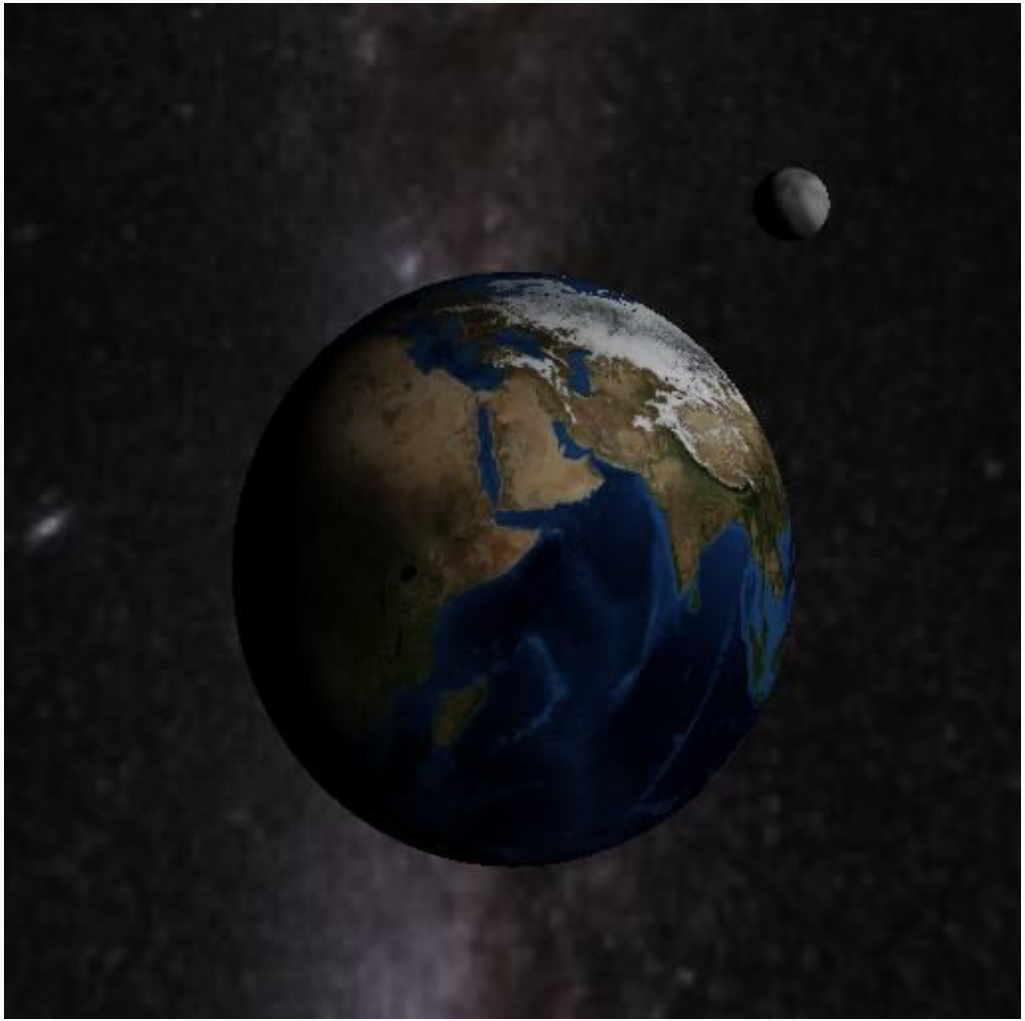


Original Raytracer



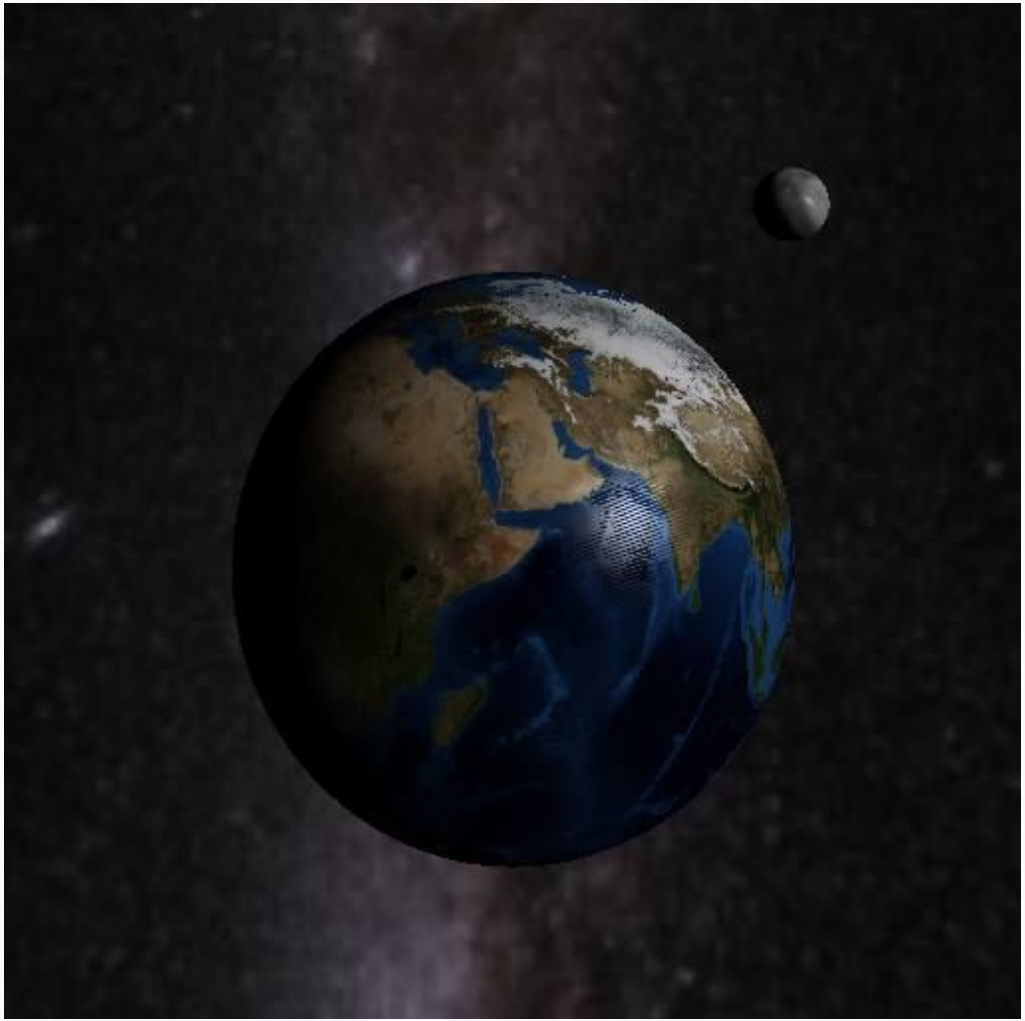
Texture Mapping

Implemented texture mapping on geometries such as spheres, trimeshes, and anything that supports UV mapping



Speculation Mapping

Implemented speculation mapping for all geometries that support UV mapping



Bump (Normal) Mapping

Implemented bump (normal) mapping for all geometries that support UV mapping

Detail (Transparency) Mapping

Implemented Detail (transparency) mapping for all geometries that support UV mapping.

Detail (Transparency) Mapping

Implemented Detail (transparency) mapping for all geometries that support UV mapping.

Detail (Transparency) Mapping

Implemented Detail (transparency) mapping for all geometries that support UV mapping.

The End

Thanks!

Questions?

References

https://s3.amazonaws.com/blenderguru.com/uploads/2011/06/Bump.jpg

https://s3.amazonaws.com/blenderguru.com/uploads/2011/06/Color%20Map.jpg

https://s-media-cache-ak0.pinimg.com/originals/a0/0e/0c/a00e0cacfebe9a34a579fdb0f1d026e2.jpg

https://www.blenderguru.com/tutorials/create-a-realistic-earth/

http://www.opengl-tutorial.org/intermediate-tutorials/tutorial-13-normal-mapping/

https://en.wikipedia.org/wiki/Painter%27s_algorithm

https://docs.unity3d.com/Manual/StandardShaderMaterialParameterNormalMap.html

https://docs.unity3d.com/Manual/StandardShaderMaterialParameterDetail.html

https://en.wikipedia.org/wiki/UV_mapping

Wald, Ingo, and Vlastimil Havran. "On Building Fast Kd-Trees for Ray Tracing, and on Doing That in O(N Log N)." 2006 IEEE Symposium on Interactive Ray Tracing (2006): n. pag. Web.