Ray Tracing

Perttu Kärnä

3312873

In nature, the objects that are visible to us are reflecting (or emitting) light. The source of light emits tiny particles called photons, that travel through the space until they hit a surface. In perfect vacuum these photons will travel along a straight line. Any surface can do some of the following to the light: reflect, refract or absorb it. Eventually some of these photons end up in the human eye, which enables us to “see” things. If a surface absorbs every photon that hits it’s surface, it can’t be seen (or it can be seen as a black shape), and if it reflects almost all photons that hit it’s surface, it’s being what we call a mirror. *Rene Descartes* introduced the concept of ray tracing, the idea of tracing rays of light and their interaction with surfaces in 1637[[1]](#footnote-1). He applied the laws of refraction and reflection to a water drop to demonstrate the production of rainbows.

Ray casting

Before ray tracing was introduced in the field of computer graphics, Arthur Appel published a paper titled “Some techniques for shading machine rendering of solids” in 1968[[2]](#footnote-2). In the paper he introduced an algorithm that would later on be called ray casting algorithm. The idea of ray casting is to shoot rays from the eye for each pixel, and determine which object the ray will hit first. Based on the lighting of the scene and the material of the property the algorithm can determine the shading of the object the ray is hitting. I.e. when a ray from the eye hits a surface, checking if the point is in shadow can be done by sending a ray from that point to the source of light - if the ray hits some surface on the way, the point is in the shadow.

Ray tracing

In 1979 Turner Whitted introduced an improved version of the ray casting algorithm in his paper “An improved illumination model for shaded display”[[3]](#footnote-3). The general idea was that when a ray hits a surface, it can generate one of the following rays: reflection, refraction or shadow. If the material of the surface is shiny, the ray “continues” from the surface as a reflection. The first object that intersects with the reflection ray is seen from the reflection. If the surface is transparent, a refraction ray is being sent in the similar way (note that the refraction ray can be either entering or exiting a material). Shadow ray is working in similar way as in ray casting. As in nature, a ray of light can be reflected and refracted multiple times.

Advantages

* Realistic illumination of light
* Quite simple to implement (recursive algorithm)

Disadvantages

* High cost of performance
  + There are other algorithms that can share the computation between pixels by using data coherence
  + On the other hand, there are other algorithms that can give even better results than ray tracing (most likely with higher cost of performance, not sure though)
* “Classical ray tracing can provide only limited set of effects: sharp reflections and refractions, and hard shadows”, Moller et al. (Real-Time Rendering, Fourth Edition, 2018)
  + Improvements for ray tracing: path tracing

Additional notes

* The idea of ray tracing is in fact quite simple
* The real-world implementations of ray tracing are probably improved versions of it and much more complex

Additional sources:

Some slides: <https://inst.eecs.berkeley.edu/~cs283/fa10/lectures/283-lecture2.pdf>

Real Time Rendering, Fourth Edition, Chapters 11.2(.2), 11.7: <https://encore.newcastle.edu.au/iii/encore/record/C__Rb4261047__SReal%20time%20rendering__Orightresult__U__X7?lang=eng&suite=cobalt>

1. https://nccastaff.bournemouth.ac.uk/jmacey/CGF/slides/RayTracing4up.pdf [↑](#footnote-ref-1)
2. http://graphics.stanford.edu/courses/Appel.pdf [↑](#footnote-ref-2)
3. http://citeseerx.ist.psu.edu/viewdoc/download;jsessionid=E9EEFA6FABBEA9DC7A163F2873D3B2A2?doi=10.1.1.156.1534&rep=rep1&type=pdf [↑](#footnote-ref-3)