Real Time Ray-Tracing in 3D Applications

Tristan Roy Zippert

December 2, 2021

Contents

Raytracing in the video game *CyberPunk 2077* [8].

1 Introduction

The prevalence of game consoles, Graphics Cards (GPUs) and modern mobile phones has put the spotlight on real-time computer graphics and 3D rendering. Real-time rendering has become even more of a commonplace with Computer Generated Imagery (CGI), Video Games, and data based simulations. However, due to the inherit complexity of projecting geometric primitives to a bitmap image (such as a screen), there are many different approaches to real-time 3D rendering. One such complexity to 3D rendering is creating photo-realistic simulations to mimic that of real-world materials.

Before the introduction of rasterization based shading rendering techniques, ray-tracing was used to showcase the concept of simulated light tracing for computers [2]. Due to the limitations of computer hardware at the time, ray-traced reflection simulation was not feasible. The recent introduction of specialized ray-tracing (RTX) computer hardware has enabled the simulation of realistic light reflection in applications without much degradation on performance.

2 History

2.1 Foundations of Reflection Simulation

Since the early days of computer generated pictures, light reflection was used to create a more photo-realistic image. The first breakthrough in this field of generating realistic reflections was with the introduction of Gouraud shading, a method that used the estimation and averaging of a models surface normal to determine color intensity at the models vertices[4]. This differed from the shading model used at the time, Flat Shading, due to the fact that Flat Shading renders brightness highlights across the whole vertex face.

A breakthrough in reflection and light rendering was outlined by researcher Bui Tuong Phong in 1975, with the introduction of the "Phong shading" technique. Phong shading makes use of linear interpolation to compute polygon surface normals, which can then be used to express the reflection characteristics of a simulated material [9]. Phong shading is currently the de-facto shading technique used for curved surface reflections since it is easy to compute, and generates better results compared to Gouraud shading. Both Gouraud shading and Phong shading are only used to describe how light is scattered and usually work alongside rasterization or ray-tracing algorithms to simulate the process of light scattering.

2.2 Rendering Equation

The goal of current industry standard rasterization and ray-tracing rendering processes is to simulate the process of light scattering to mimic that of real-life. A generalized equation, as exemplified in Figure 1, was created by David Immel and James Kajiya to compute light radiance leaving a certain point on a surface[6]. Solving and using this equation when making a renderer is essential for realistic rendering, due to the fact that it's the basis for how light is simulated with the surface reflection and the oncoming light angle(s).

$$\mathrm{L}_0(P,\omega_0) = \int_{S^2} f(P,\omega_0,\omega_i) L_i(P,\omega_i) |cos\theta_i| d\omega_i$$

Figure 1: Rendering Equation

Figure 1 represents one variation of the rendering equation, as multiple variations of this equation exist with the same generalized concept. L_0 is the sum of emitted light from all directions (light radiance) at point P, while L_i is computed recursively from all surfaces visible from point P [5, p. 12]. Most solutions for how to solve this equation for a 3D renderer make use of a method derived from a Monte Carlo method of randomly sampling parts of a scene. Monte Carlo methods of random sampling are used for ray-traced rendering as it is used to sample more than one ray per pixel.

3 Ray-Tracing

Ray-tracing based rendering systems in real-time 3D applications use rays cast from a scene camera that "bounce" towards scene lights to illuminate object surfaces. This method of calculating reflections has been in use before, however, it has been reserved for non-real-time applications. Hardware accelerated ray-tracing allows for the generation of realistic looking light reflections with little to no impact on the speed of a renderer. The previous best method for simulating real-time reflections only simulated light bounces for only the objects in the view of the camera. A comparison between the two types of light simulation techniques is shown in Figure 2.

Figure 2: Real-time ray-tracing in *Control*. Source: Remedy Entertainment [1, p. 743]

As shown in Figure 2, specifically on the right side, the effect of real time ray-tracing can achieve realistic lighting without an impact on performance. The left half of the image showcases the use of the previous method, specifically the "Screen-Space" reflection technique. This method of reflection generation can only reflect the light that's in the view of the simulated camera, and is specifically noticeable with the missing character reflection in the lower left image.

4 Future of Real-Time Rendering

While calculating ray-traces are still computationally expensive – but well within the range of performance for some real-time applications – it still experiences slowdowns due to combining and sampling high amounts of ray bounces. A new, efficient, model of ray-tracing has started to emerge for real-time 3D applications. Spatiotemporal Resample generates the scene geometry and only recalculates (or "resamples") casted light rays for objects of the scene that have changed [3]. This allows for speed increases in applications already using real-time ray-tracing, while at the same time maintaining the hyper-realistic look of the generated images. Ray-tracing can also be combined with current rasterization techniques to not only allow support on older hardware and systems, but also offer faster performance in applications that need the speed of a baked-light map.

4.1 Ray-Marching

Another iteration of ray-based rendering systems is that of ray-marching based rendering systems. Much like ray-tracing, ray-marching sends rays from the scene camera to bounce towards the simulated scene light. However instead of computing reflections from one intersection sample from the cast ray, as done with ray-tracing, ray-marching marches along (forwards and backwards) to determine the intersection sample points. Some implementations of ray-march based renderer can produce a higher fidelity image compared to that of ray-tracing, but at the cost of performance. Ray-marching techniques could be adapted to allow the use of pre-computed light probes for change within a depth buffer for computing reflections[7]. With the constant strive towards hyper-realistic computer rendering, ray-tracing and ray-marching is the next step for 3D rendering systems.

About the Author

Tristan Zippert is an undergraduate computer science student at the University of Maine. He plans to pursue a concentration on hybrid rendering systems – systems that are a combination of both standard rasterization and real-time raytracing. In his free time he works on a bare-bones game engine with 3D and 2D rendering capabilities in C++ using Vulkan and OpenGL.

5 References

[1] Peter Shirley Adam Marrs and Ingo Wald, eds. *Ray Tracing Gems II*. http://raytracinggems.com/rtg2. Apress, 2021.

[2] Arthur Appel. "Some techniques for shading machine renderings of solids". In: *Proceedings of the April 30–May 2, 1968, spring joint computer conference on - AFIPS 68 (Spring)* (1968). DOI: 10.1145/1468075.1468082.

[3] Benedikt Bitterli et al. "Spatiotemporal Reservoir Resampling for Real-Time Ray Tracing with Dynamic Direct Lighting". In: *ACM Trans. Graph.* 39.4 (July 2020). ISSN: 0730-0301. DOI: 10.1145/3386569.3392481. URL: https://doi-org.wv-o-ursus-proxy02.ursus.maine.edu/10.1145/3386569.3392481.

[4] H. Gouraud. "Continuous Shading of Curved Surfaces". In: *IEEE Transactions on Computers* C-20.6 (1971), pp. 623–629. DOI: 10.1109/T-C.1971.223313.

[5] Eric Haines and Tomas Akenine-Möller, eds. *Ray Tracing Gems*. http://raytracinggems.com. Apress, 2019.

[6] James T. Kajiya. "The Rendering Equation". In: *Proceedings of the 13th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '86. New York, NY, USA: Association for Computing Machinery, 1986, pp. 143–150. ISBN: 0897911962. DOI: 10.1145/15922.15902. URL: https://doi.org/10.1145/15922.15902.

[7] Morgan McGuire et al. "Real-Time Global Illumination Using Precomputed Light Field Probes". In: *Proceedings of the 21st ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*. I3D '17. San Francisco, California: Association for Computing Machinery, 2017. ISBN: 9781450348867. DOI: 10.1145/3023368.3023378. URL: https://doi.org/10.1145/3023368.3023378.

[8] Alessio Palumbo. *Cyberpunk 2077 Looks Fantastic in These New Ray Traced PC Screenshots*. June 2020. URL: https://wccftech.com/cyberpunk-2077-looks-fantastic-in-these-new-ray-traced-pc-screenshots/.

[9] Bui Tuong Phong. "Illumination for computer generated pictures". In: *Communications of the ACM* 18.6 (1975), pp. 311–317. DOI: 10.1145/360825.360839.