Rendering Objects Realistically in a 3D Space Using Ray Tracing

Eric George
Computer Science 30-Cohort AP
Ernest Manning High School

Ray Tracing to Improve Realism

As technology develops and the communication world moves online, it becomes imperative for graphical objects to be rendered onto a device. To portray the real world on a screen effectively opens new aspects in communication, entertainment, and even business and industry. The essential purpose of this ray tracing is to bring realism onto a screen.

Design

The design of the Ray Tracer Graphical User Interface follows the standard architecture of MVC (Model View Controller). This architecture is used primarily for fast development and ease of adding to the program. While the architecture might lack scalability, it is useful for the implementing improvements in ray tracing to improve the program. One of these design features was to have the rendering occur on a separate thread so that the GUI (Graphical User Interface) would remain responsive. This was implemented through computing the rendering on a separate thread from the GUI.

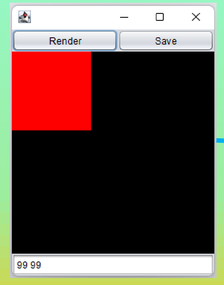

Figure 1. *Red Square*

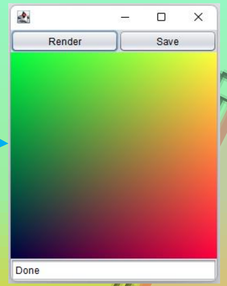

Figure 2. Working with colour

Another important design feature is having shapes and materials be their own classes. This allows instances to act differently based on whether they are a sphere or are metallic, but also allows them to be identifiable by the shape and material of the object. As well, ray tracing of the object differs based on the shape and material, so this type of design follows the open-closed principle – open to extension but closed to modification.

Figure 3. A red sphere

Figure 4. Coloured with normals

Ray Tracing

Ray tracing attempts to imitate reality, however an exact imitation is by no means the most efficient way of doing it. While normally light is emitted at a source and scatters, bouncing off objects countless times before, by chance, it hits the human retina and is perceived. However, computing this would be heavy for the machine if possible. Instead, the reverse process is done, where light is traced from the retina outwards, bouncing off objects to a limit. This greatly increases efficiency and decreases workload.

Figure 5. Multiple Spheres

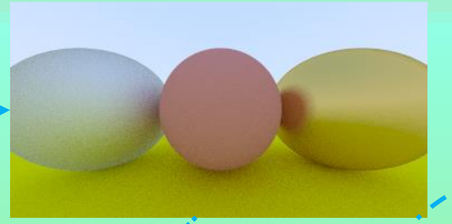

Figure 6. Different Materials

Ray Tracing Cont.

Rays are shot from a camera object, which has its own location, at a 'viewing screen' which represents the various angles at which the light strikes the eye and is received by the retina. When an object is struck by a ray object, how the ray scatters is defined by the material of the object. For example, a metal sphere scatters light differently than a Lambertian metal sphere, as shown in Figure 6. As well, the shape of the object is used to determine how the ray hits the object and the angle it is reflected off by. This results in more precise rendering of the objects.

Figure 7. Glass

Figure 8. Camera and Defocus Blur

Ray Tracing Cont.

Another important feature is implementing a glass material. An important feature of glass is that it sometimes refracts and sometimes reflects. Implementing this realistically would've been computationally intensive if not for the Schlick approximation, which simplifies it with realistic results. As well, having the rays shoot from a moveable camera allows one to adjust the viewing location without adjusting the entire scene. Combining all these efforts together results in the ray tracer being able to produce images that are close to reality.

Figure 9. Random Scene

Conclusions and the Future of Rendering

The ray tracer accurately renders spheres of different sizes at varying locations and of different materials: metal, Lambertian metal, and dielectrics (glass). It can produce these images to a great degree of realism using a manageable amount of computation power. If the object is described using a material and shape implemented by the ray tracer, it can be rendered– a wonderful architectural product. However, there is much that can be improved:

- Increasing efficiency for dynamic ray tracing in a non-static context.
- Improvement upon shapes implemented or dynamic system for any polygon.
- Increasing material span or systematically determining scatter by a variety of material factors.

Regardless, rendering can only get faster and more realistic in the future. Indeed, the future of rendering is something to look forward to.