Computer Graphics

(Introduction to Raytracing)

Lecture 008

What is Ray Tracing ?

- Ray tracing is a method for generating realistic images by computer, in which the paths of individual rays of light are followed from the viewer to their points of origin.
- A Ray tracer is any program that implements this method.
- Ray tracing is best known for modeling three phenomena
  - *Reflection*
  - *Refraction*
  - *Shadows*
- Since ray tracing makes use of the actual physics and mathematics behind light interaction with objects, the images it produces can be very photo-realistic.

Scenes, Objects, Light Sources and a Camera

- A scene is a collection of objects and light sources that are viewed from a camera. Each object is arranged in the world space. Thus for example you can have a number of objects rotating around a light source and a camera from where you are viewing the scene.

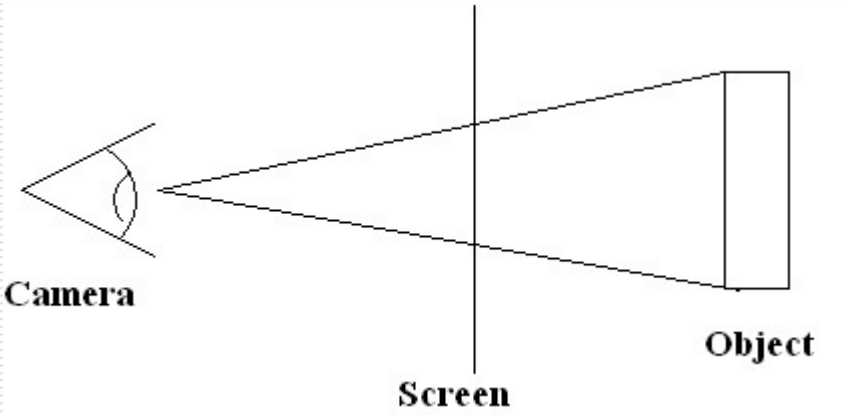
- Objects can be anything really, however in the context of ray tracers, objects need to have some mathematical description. For example spheres, cones, boxes, planes, etc… Clearly these objects can be combined together to form more complex objects. Objects have different properties associated with them. For example shininess, colour, textures.

- Light sources (like objects) can be placed anywhere on the scene. Some light sources have a direction and others simply emit light in all directions. Light sources are obviously a key element for any ray tracer. Light sources have an associated light intensity, which describes their brightness and colour.

- Finally, we need a camera from where to view the scene. The position of the camera will determine where (on the computer screen) the rays hit.

Ray Casting

- Ray casting is a method in which the visible surfaces of objects (those parts of the scene that are immediately visible to the camera) are found by throwing (or casting) rays of light from the viewer into the scene.



- Rays can be thought of straight lines extending from the eye into the scene. These rays are the foundation of any ray tracer.

Ray Casting (Algorithm)

- For each pixel in the screen, a ray is cast from the eye (camera), through that pixel, and into the image's world-space.
  - Every object in the scene is tested to see if the given ray intersects any of them. If there are multiple objects in the scene, it is possible that any given ray intersect more than one object in the scene.
  - For each ray, the intersection that is nearest to the eye (camera) is the one that is visible to the eye. The intensity (that is, colour) at that point is the colour that is given to the pixel through which the ray passed.
- This process is carried out for every pixel on the screen. So suppose we have a resolution of 640x480 for a total of 307,200 pixels with 30 objects in the scene. This basic rendering would process a total of 9,216,000 intersection test !
- This makes the algorithm very computational intensive.

Ray Tracing Introduction

- Whereas ray casting concerns itself only with finding the visible surfaces of objects, ray tracing takes that a few steps further and actually tries to determine what each visible surface looks like.
- This is obviously very computational intensive, however, ray tracing allows you to create several kinds of effects that are very difficult or even impossible to do with other rendering methods.
- These effects include three items common to every ray tracer:
  - Reflection
  - Transparency
  - Shadows
- Ray tracing algorithms are recursive in nature.
- Recursion is a means of computing a result in which a given process repeats itself a number of times. Clearly infinite recursion is not useful in our case, so we need recursion to stop after a given number of loops.

Ray Tracing (Basic Algorithm)

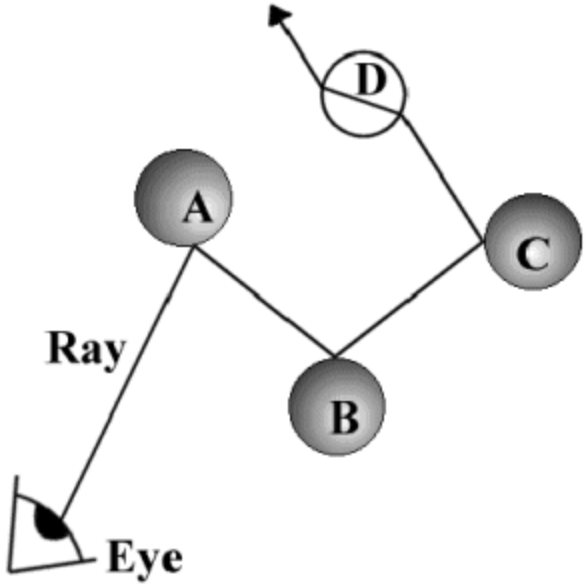
- The algorithm does the same work as the ray casting algorithm in order to determine the nearest object intersection.
- Once these point of intersections are determined, the algorithm recursively shoots more rays from these points of intersections.
- These new rays (secondary rays) are used to determine
  - What objects are reflected at that point
  - What objects may be seen through the object at that point
  - What light sources are visible from that point.
- The new rays that are created depend also on the material of the surface with which the ray intersects. For example if the object is a glass of water that we need to determine what objects are visible through the glass. If the material is specular (a mirror for example) the algorithm needs to calculate the reflected objects on the surface.

Ray Tracing (Reflection)

- If the surface that the ray intersected was reflective, like a mirror, the ray tracer must determine the color at that point by finding, not only the color of the surface, but also the color of the reflection of any objects at that point. For instance, imagine a shiny wooden table. From where you look at it, you will be able to see the reflections of objects on the table. This is because the light from those objects travels to the table, bounces off it, and travels to your eye.

- One can immediately note how the recursion works here. The value at D is bounced back to C, then B, then to the original intersection.

- The algorithm has to calculate the angle at which the ray should bounce off from each sphere.

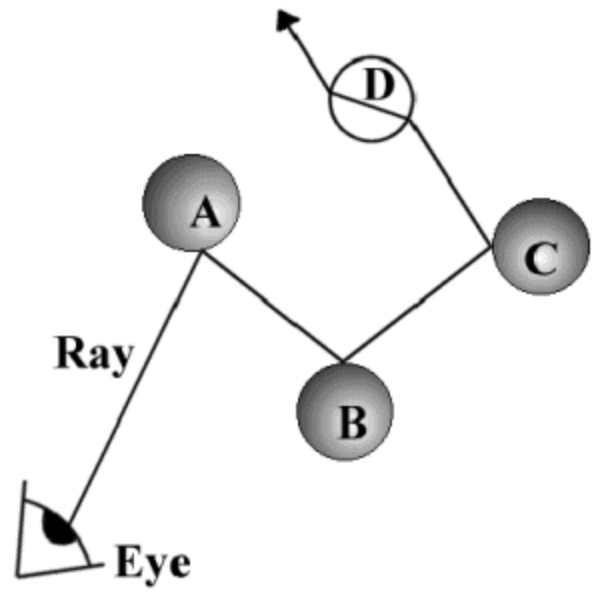


Ray Tracing (Refraction/Transparency)

- Transparency is modeled similarly to reflection, but instead of bouncing the new (or "spawned") ray off of the surface, the ray is bent into and through the surface to model *refraction*. Refraction is an optical phenomenon caused when light bends as it travels through a given substance. For instance, fill a glass with water, and put a pencil in it. Looking from a bit above the glass, the pencil will appear to be bent where it enters the water.

- Some objects will bends light more than others. This parameter would form part of the object material.

- Sphere D is transparent, therefore light is bent both on hitting and on exiting the sphere.



Ray Tracing (Shadows)

- Shadows are relatively easy to calculate. Once the algorithm determines the points of intersections, the shadows may be calculated by firing more rays, one at each of the light sources. If any non-transparent objects intersect the ray, then no light can arrive to the first object via that ray, and the surface is in shadow.

- The ray travels from the eye to where it intersects with sphere A. In order to determine whether there is a shadow at that point, one shadow ray is fired at each light source. The shadow ray from A to the second light source travels uninterrupted, so light reaches the point from that light source. However, sphere B lies between sphere A and light source #1, so the point is in shadow with respect to that light source. The intensity of color at that point on sphere A would represent the fact that only one of the two light sources is shining directly on that point.