Foundations of Raytracing

Introduction to raytracing

- Raytracing is historically early method for image synthesis based on 3D scene description
- Due to hardware advancements, rasterization-based rendering gained a lot of attention
- Raytracing is intuitive and straightforward method of simulating physical phenomena of how image is created. Based upon raytracing, more advanced rendering methods are build. Furthermore, understanding raytracing gives good foundations for understanding rasterization-based rendering systems.

Raytracing: intuition

- To produce an image, first, we need information from where we look at 3D scene and a 2D surface on which 3D scene will be projected as an image.
  - In lesson on camera, we mentioned exactly these elements: eye and film plane. Therefore, for these purposes, we can use pinhole camera.
  - Captured image color and brightness of objects in 3D scene is a result of light and objects (shape and material) interaction
- To figure out interaction of light and objects we need information about objects in 3D scene as well as light sources.
  - Without light we wouldn't see objects, without objects we wouldn't see light.
  - Using this information, we can simulate light-object interaction.
- Simulating physical (real-world) light-object interaction means simulating all light rays paths from light and their interactions with objects. Furthermore, only small amount of that light actually falls on camera. This kind of simulation is called **forward ray-tracing or light tracing.** Simulating this is not tractable! Therefore, we simplify:
  - First, Only light-object interaction that are visible from camera should be simulated
  - Second, we reverse the process: we start with rays that fall into camera and build from there.
- Based on previous discussion, we trace rays from sensor to the objects. This simulation is called **backward ray-tracing or eye-tracing***.

* "An Improved Illumination Model for Shaded Display" - Turner Whitted

Backward ray-tracing*

- Based on previous discussion, now we understand the first steps:
  - Generate ray using camera (eye and film) → primary/camera/visibility ray
  - Trace ray into the scene
- Tracing ray into the scene can end up in following possibilities:
  - Intersects with object
  - No intersection →hit background
- In the case where ray hits the object, two main steps are done:
  - Calculate amount of light (incoming) falling on intersection point → shadow/light ray.
    - This ray can be directly traced from intersection to light source → direct illumination
    - Advanced: sample various directions from this point in 3D scene to obtain reflections of other objects → global illumination
  - Calculate amount of light reflected in primary ray direction using incoming light and material specification

* Method in computer graphics using the concept of shooting and following rays from light or eye is called path-tracing.

Practical raytracing*

Algorithm:

- For each pixel in raster image:
  - Generate primary ray by connecting eye and film using camera information
  - Shoot primary ray into the scene
  - For each object in the scene
    - Check if intersected with primary ray. If intersect multiple objects, take one closest to the eye
  - For intersection, generate shadow ray from intersection point to all lights in the scene
    - If shadow ray is not intersecting anything, lit the pixel

* Arthur Appel in 1969 - "Some Techniques for Shading Machine Renderings of Solids"

Practical raytracing: notes

- Discussed method can be separated in:
  - Determining visibility: determine which point in 3D scene is visible for each pixel of image
  - Shading: calculating the color of visible point
- Both steps require extremely time-consuming intersections between rays and scene objects (geometry)
  - Efficient ray-object intersection method is needed
  - Efficient spatial search is needed
- That is why real-time graphics is predominantly using GPU rasterization approach
- For high-quality production rendering, ray-tracing is almost always used.
- However, real-time ray-tracing is now possible to certain degree with certain hardware and it is hot research topic!

Extending basic raytracing