CS-174A Discussion 1B, Week 10

@ Yunqi Guo

@ DODD 161 / Friday / 12:00pm-1:50pm

@ https://github.com/luckiday/cs174a-1b-2019f (https://github.com/luckiday/cs174a-1b-2019f)
(Short link: https://bit.ly/32Zt3sg (https://bit.ly/32Zt3sg))

Outline

- Final review

Final Exam Review

- Notes are from lecture slides

Ray Tracing

http://madebyevan.com/webgl-path-tracing/ (http://madebyevan.com/webgl-path-tracing/)

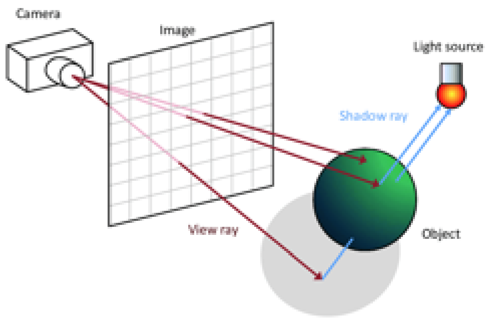
Ray Casting vs Ray Tracing

- *Ray casting* involves creating a ray and calculating what it hits (or if it hits a particular thing).
- *Ray tracing* involves creating a ray and tracing its path as it reflects/refracts through the scene so that you can determine what light might arrive at its source from the other direction.

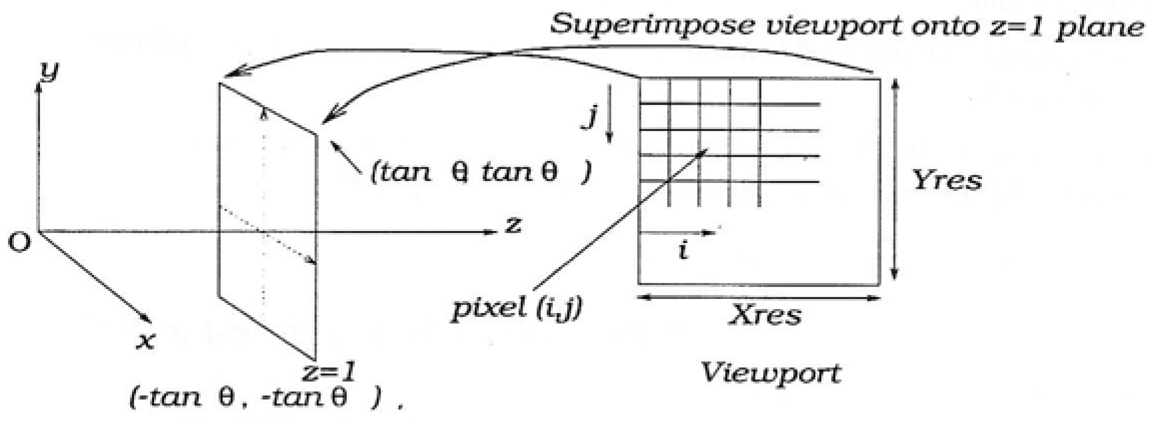
Ray Casting

For each pixel in viewport

1. Generate ray emanating from $Eye(O)$ through pixel
2. Find intersection between ray and objects
3. Pick intersection point closest to $Eye(O)$
4. Plot pixel with color of closest object



Step 1: Generate ray emanating from Eye (O) through pixel



Step 2: Find intersection between ray and objects

- Intersection test: if no intersection, skip part b
- Intersection calculation: for polygons and for spheres

Recursive Ray Tracing Algorithm

Three types of rays:

- A reflection ray is traced in the mirror-reflection direction. The closest object it intersects is what will be seen in the reflection.
- Refraction rays traveling through transparent material work similarly, with the addition that a refractive ray could be entering or exiting a material.
- A shadow ray is traced toward each light. If any opaque object is found between the surface and the light, the surface is in shadow and the light does not illuminate it. This recursive ray tracing added more realism to ray traced images.

Reflection and Refractiong

- Reflected Ray Direction

$$r = i - 2(i \cdot n)n$$

- Refracted Ray Direction
  Snell's Law $\eta_1 \sin\theta_i = \eta_2 \sin\theta_t$

Ray Tree

- Formed of primary and secondary rays
- Evaluated bottom up

Tree terminates if

- No intersection for a ray
- Tree depth has reached a specified level
- Intensity of Ir and It becomes very low

Efficiency Considerations

- Total # of shadow rays spawned = m(2^n – 1)
  m = # light sources; n = depth of ray-tree
- Total # of rays = (m + 1)(2^n – 1)
- Back faces cannot be culled
- Clipping cannot be done for view volume or behind eye
- 75%-95% of time is spent in intersection calculations
- Use bounding box testing or hierarchies (octrees)

Lighting:

- Light sources (specular, diffuse, ambient)

Shading

- Phong / Gourad
- Flat / Smooth
- Barycentric and Trilinear interpolation
- Normal vector calculation for phong reflection model

Texture Mappings

- Texture, bump map, displacement map, environment map
- UV coordinate system

CS-174A Discussion 1B, Fall 2019

@ Yunqi Guo

@ guoyunqi@gmail.com (mailto:guoyunqi@gmail.com)

@ https://github.com/luckiday/cs174a-1b-2019f (https://github.com/luckiday/cs174a-1b-2019f) (Short link: https://bit.ly/32Zt3sg (https://bit.ly/32Zt3sg))