CS 543: Computer Graphics
Lecture 11 (Part II): Raytracing (Part III)

Emmanuel Agu

Where are we?

```
Define the objects and light sources in the scene
Set up the camera
for(int r = 0; r < nRows; r+= blockSize){
  for(int c = 0; c < nCols; c+= blockSize){
      1. Build the rc-th ray
      2. Find all object intersections with rc-th ray
      3. Identify closest object intersection
      4. Compute the “hit point” where the ray hits the
          object, and normal vector at that point
      5. Find  color (clr) of light to eye along ray
      glColor3f(clr.red, clr.green, clr.blue);
      glRecti(c, r, c + blockSize, r + blockSize);
  }
}
```

Find Object Intersections with rc-th ray

- Much of work in ray tracing lies in finding intersections with generic objects
- Break into two parts
  - Deal with untransformed, generic (dimension 1) shape
  - Then embellish to deal with transformed shape
- Ray generic object intersection best found by using implicit form of each shape. E.g. generic sphere is

$$F(x, y, z) = x^2 + y^2 + z^2 - 1$$

- Approach: ray r(t) hits a surface when its implicit eqn = 0
- So for ray with starting point S and direction **c**

$$r(t) = S + \mathbf{c}t$$

$$F(S + \mathbf{c}t_{hit}) = 0$$

Ray Intersection with Generic Plane

- Generic Plane?
- Yes! Floors, walls, in a room, etc
- Generic plane is *xy*-plane, or $z = 0$
- For ray

$$r(t) = S + \mathbf{c}t$$

- There exists a t_{hit} such that

$$S_z + \mathbf{c}_z t_h = 0$$

- Solving,

$$t_{hit} = -\frac{S_z}{c_z}$$

Ray Intersection with Generic Plane

- Hit point P_{hit} is given by

$$P_{hit} = S - \mathbf{c}(S_z / c_z)$$

- Numerical example?
- Where does the ray *r(t)* = (4, 1, 3) + (-3, -5, -3)*t* hit the generic plane?
- Soln:

$$t_{hit} = -\frac{S_z}{c_z} = \frac{-3}{-3} = 1$$

- And hit point is given by

$$S + \mathbf{c} = (1, -4, 0)$$

Ray Intersection with Generic Sphere

- Generic sphere has form

$$x^2 + y^2 + z^2 = 1$$

$$x^2 + y^2 + z^2 - 1 = 0$$

$$F(x, y, z) = x^2 + y^2 + z^2 - 1$$

$$F(P) = |P|^2 - 1$$

- Substituting S + **c**t in *F(P) = 0,* we get

$$|S + \mathbf{c}t|^2 - 1 = 0$$

$$|\mathbf{c}|^2 t^2 + 2(S \cdot \mathbf{c})t + (|S|^2 - 1) = 0$$

- This is a quadratic equation of the form $At^2 + 2Bt + C = 0$ where $A = |\mathbf{c}|^2$, $B = S.\mathbf{c}$ and $C = |S|^2 - 1$

Ray Intersection with Generic Sphere

- Solving

$$t_h = \frac{B}{A} \pm \frac{\sqrt{B^2 - AC}}{A}$$

- If discrimant $B^2 - AC$ is negative, no solutions, ray misses sphere
- If discriminant is zero, ray grazes sphere at one point and hit time is –B/A
- If discriminant is +ve, two hit times t1 and t2 (+ve and –ve) discriminant
- Numerical example? See example 14.4.2 on pg. 739

What about transformed Objects

- Generic objects are untransformed:
  - No translation, scaling, rotation
- Real scene: generic objects instantiated, then transformed by a composite matrix *T*,
- We can easily find the inverse transform *T'*
- **Problem definition:** We want to find ray intersection with transformed object
- Easy by just simply finding the implicit form of the transformed object
- May be tough to find implicit form of transformed object
- Hmmm… is there an easier way?

What about transformed Objects

- Yes
- Basic idea: if object is transformed by *T*, then ray–object intersection is the same as inverse transformed ray with generic object
- Algorithm
  - Find *T'* from initial *T* transform matrix of object
  - Inverse transform the ray to get ($S' + \mathbf{c'}t$)
  - Find intersection time, t_{hit} of the ray with the generic object
  - Use the *same* t_{hit} in $S + \mathbf{c}t$ to identify the actual hit point
- This beautiful trick greatly simplifies ray tracing
- We only need to come up with code that intersects ray with *generic* object
- Remember that programmer does transforms anyway, so we can easily track and get *T*

References

- Hill, chapter 14