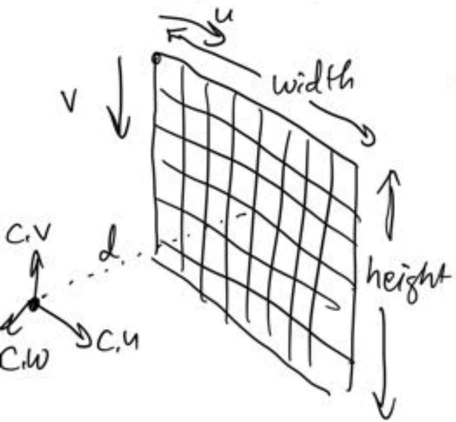
CSC 317

Tutorial: Ray Casting

src/viewing_ray.cpp

- Output: ray.origin, ray.direction (in worldspace)
- Input:
  - (i, j): pixel index of the pixel we want to shoot a ray through
  - (height, width): pixel size of the image
  - (camera.height, camera.width): worldspace (physical) size of the image
  - camera.d: focal length (we want to make sure ray.direction lands on the image plane)
  - (camera.u, camera.v): camera axes oriented in worldspace
- Ray origin is at the camera location
  - Look at Camera.h
- Careful:
  - We want the ray going through the MIDDLE of the pixel (not the top left corner), need to have an offset

viewing_ray:



Suppose we want (u,v) of the top-left corner.

$$u = -\frac{c.width}{2}$$

$$v = +\frac{c.height}{2}$$

Now, shift it by half a pixel to get (u,v) of the pixel $(0,0)$:

$$u = -\frac{c.width}{2} + \underbrace{\frac{c.width}{width}}_{\text{pixel width}} * 0.5$$

$$v = +\frac{c.height}{2} - \underbrace{\frac{c.height}{height}}_{\text{pixel height}} * 0.5$$

Now, shift it by (i,j) pixels:

Well, we already know pixel size so:

$$u = -\frac{c.width}{2} + \frac{c.width}{width}(j+0.5)$$

$$v = +\frac{c.height}{2} - \frac{c.height}{height}(i+0.5)$$

But u,v are scalars along the direction of c.u, c.v

(and c.d is along c.w)

So:

$$direction = -c.d * c.w + \underbrace{u}_{\text{scalars}} * \underbrace{c.u}_{\text{vectors}} + v * c.v$$

src/first_hit.cpp

- Output:
  - hit_id: index of object first hit
  - t: parametric distance of the ray hit (ie ray.origin+t*ray.direction gives the surface hit)
  - n: surface normal at the location of first hit
  - return bool: if the ray hit any object (false if not)
- Input:
  - ray: ray to search
  - min_t: minimum value of t to accept (culls surfaces behind and too close to the camera)
  - objects: list of objects in scene
- Loop through all the objects, call the object intersect with the ray, and see which object gives the smallest (greater than min_t) t value
- Note: no actual intersection checks should be done here, just call the object intersects and compare

src/Sphere.cpp

Ray-Sphere Intersect

p = point
c = center
r = radius

Implicit eqn of a sphere

$$(\bar{p}-\bar{c})^2 - r^2 = 0$$

$$(\bar{p}-\bar{c})\cdot(\bar{p}-\bar{c}) - r^2 = 0$$

$$(\bar{e}+\bar{d}t-\bar{c})\cdot(\bar{e}+\bar{d}t-\bar{c}) - r^2 = 0$$

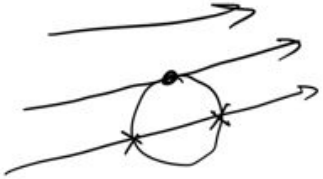
$$\underbrace{(\bar{d}\cdot\bar{d})}_{a}t^2 + \underbrace{(2\bar{d}\cdot(\bar{e}-\bar{c}))}_{b}t + \underbrace{(\bar{e}-\bar{c})\cdot(\bar{e}-\bar{c}) - r^2}_{c} = 0$$

$$at^2 + bt + c = 0$$

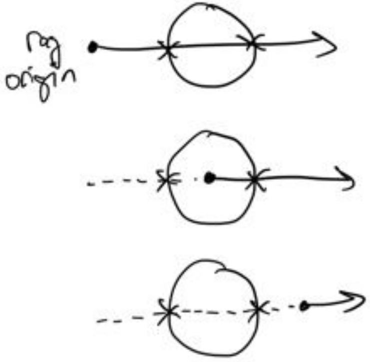
Quadratic Formula

$$t = \frac{-b \pm \sqrt{b^2-4ac}}{2a}$$

Discriminant
$b^2 - 4ac$
- $< 0 \Rightarrow$ no hits
- $= 0 \Rightarrow$ one hit
- $> 0 \Rightarrow$ two hits



Note with two hits you have three cases:



both in front ($> min\_t$)

one in front

none in front

Check t against min_t!

src/Plane.cpp

<u>Ray-Plane Intersect</u>

$\bar{p}_0$: point on the plane

$\bar{n}$: plane normal

$$(\bar{p} - \bar{p}_0) \cdot \bar{n} = 0$$

$$(\bar{e} + \bar{d}t - \bar{p}_0) \cdot \bar{n} = 0$$

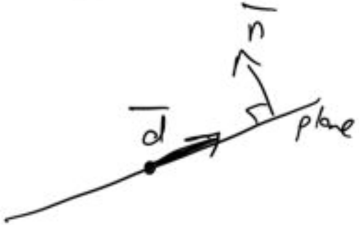
solve for t

$$\Rightarrow \bar{e} \cdot \bar{n} + (\bar{d} \cdot \bar{n})t - \bar{p}_0 \cdot \bar{n} = 0$$

$$\boxed{t = \frac{\bar{p}_0 \cdot \bar{n} - \bar{e} \cdot \bar{n}}{\bar{d} \cdot \bar{n}}}$$

<u>Note</u> if $\bar{d} \cdot \bar{n} = 0$ this is undefined!
(either no hit or the camera is in the plane)

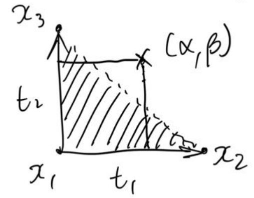
or



src/Triangle.cpp

Ray-Triangle Intersect

Canonical Triangle:



Any 3 corners $x_1 x_2 x_3$ form two vectors t_1, t_2.

"origin" of the canonical triangle

Then $\bar{p} = \alpha \bar{t}_1 + \beta \bar{t}_2 + x_1$

$\bar{e} + \bar{d}\,t = \alpha \bar{t}_1 + \beta \bar{t}_2 + x_1$

$-x_1 + \bar{e} = \alpha \bar{t}_1 + \beta \bar{t}_2 - t \bar{d}$

unknown scalars

$$\begin{bmatrix} | & | & | \\ \bar{t}_1 & \bar{t}_2 & \bar{d} \\ | & | & | \end{bmatrix} \begin{bmatrix} \alpha \\ \beta \\ t \end{bmatrix} = \begin{bmatrix} | \\ \bar{e} - x_1 \\ | \end{bmatrix}$$

Linear System $A \cdot x = b$

Solve using eigen inverse

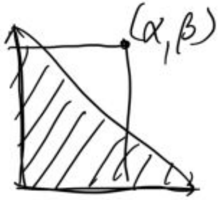
$$\boxed{x = A.\text{inverse} * b}$$

Then Make sure $\alpha \geq 0$

$\beta \geq 0$

$\alpha + \beta \leq 1$

$t \geq min\_t$



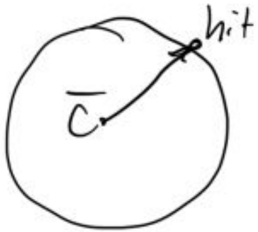
to check if the intersection is inside the triangle.

Note what happens if A is non-invertible?

Normals

Normals

Sphere



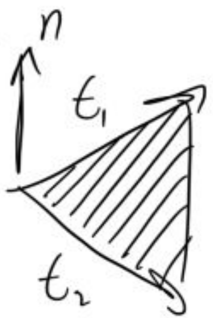
$$\frac{hit - \bar{c}}{radius}$$

normalization

Plane

plane.normal (normalize just in case)

Triangle



$$\frac{\bar{t}_1 \times \bar{t}_2}{\| \bar{t}_1 \times \bar{t}_2 \|}$$

normalization.

src/TriangleSoup.cpp

- Just call first hit on the list of triangles