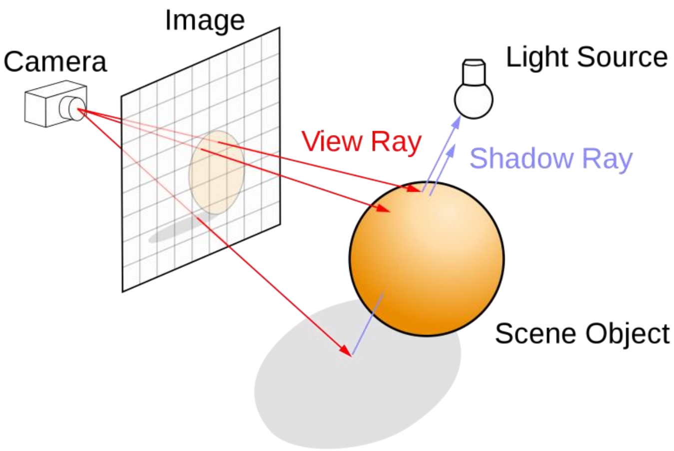
CENG 477
Introduction to Computer Graphics

Ray Tracing: Geometry

Ray Tracing

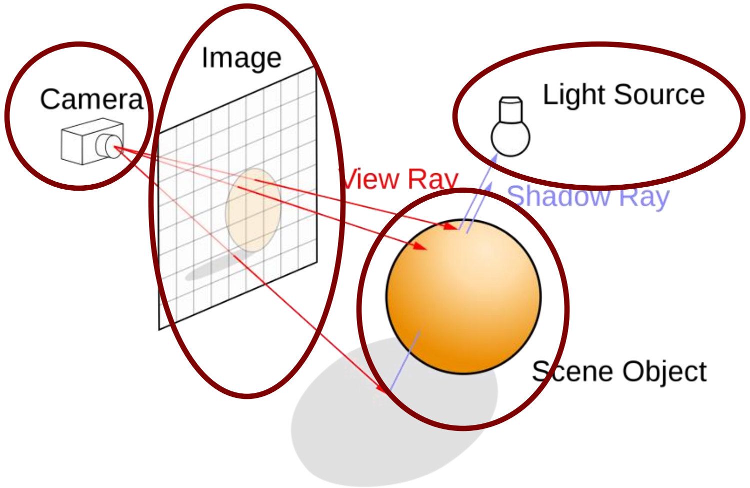
- In ray tracing, we model the propagation of light and its interaction with materials to create realistic images
- Different from reality, we assume that the rays originate from the eye (or the camera)
- This allows us to avoid processing rays that will not be visible to the eye (or the camera)



wikipedia.com

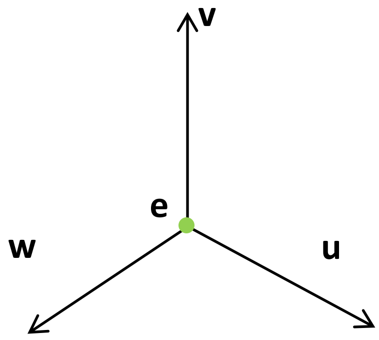
Components

- In RT, we have the following components:
  - Camera (or eye)
  - Image plane
  - Objects
  - Light sources
  - and lots of rays!



Camera

- Camera represents the origin of the rays that we will trace
- It is represented by a position (**e**) and orientation (**u**, **v**, **w**)
- These vectors are orthogonal to each other



- The position and the orientation are defined with respect to a global (or world) coordinate system
- This global system has origin at (0, 0, 0) and its three axis are: (1, 0, 0), (0, 1, 0), (0, 0, 1)

Camera

- The **u**, **v**, **w** vectors of the camera have the following meaning:
  - **v**: up vector
  - **w**: opposite of gaze vector
  - **u**: **v** x **w** with x representing cross-product

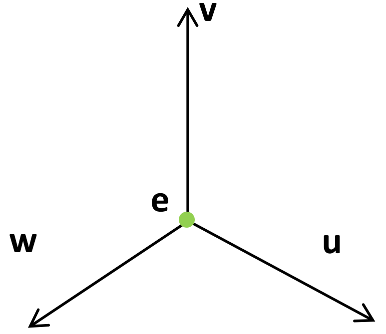
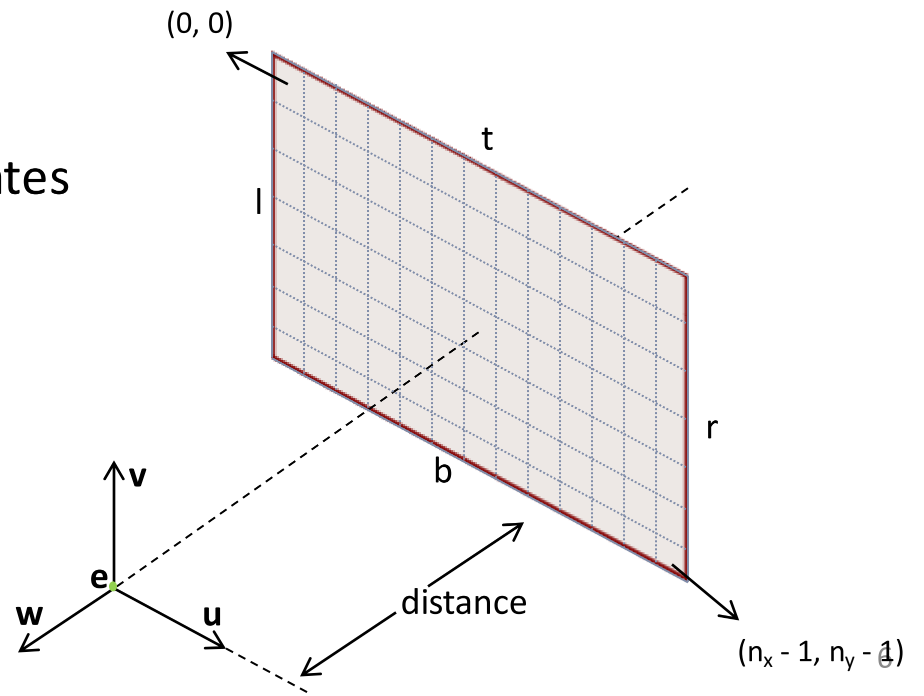


Image Plane

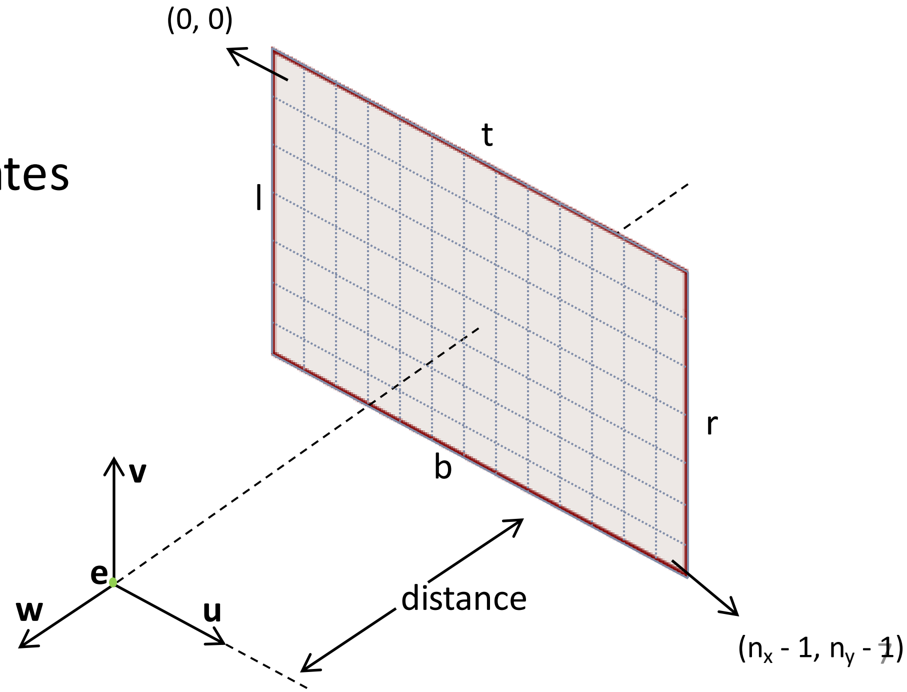
- Image plane is a surface on which the final image is formed
- It is divided into pixels through which rays will be cast
- It is represented by its:
  - Resolution (n_x, n_y)
  - Distance to the camera
  - Left, right, top, bottom coordinates



- The image plane is typically centered and orthogonal with respect to the camera

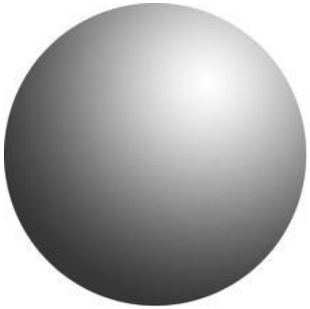
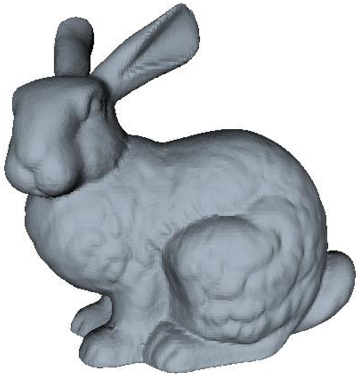
Image Plane

- Image plane is a surface on which the final image is formed
- It is divided into pixels through which rays will be cast
- It is represented by its:
  - Resolution (n_x, n_y)
  - Distance to the camera
  - Left, right, top, bottom coordinates



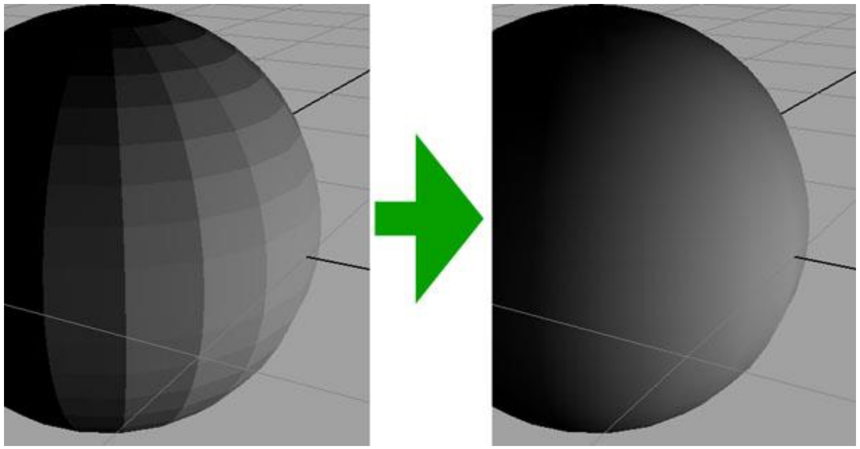
- l, r, t, b are with coordinates with respect to the camera coordinate system (not the world coordinate system)

Objects

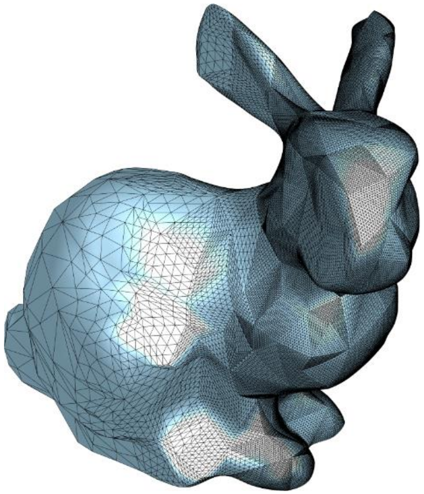
  

- Objects consist of mathematically defined geometrical shapes or meshes made up of triangles

Objects



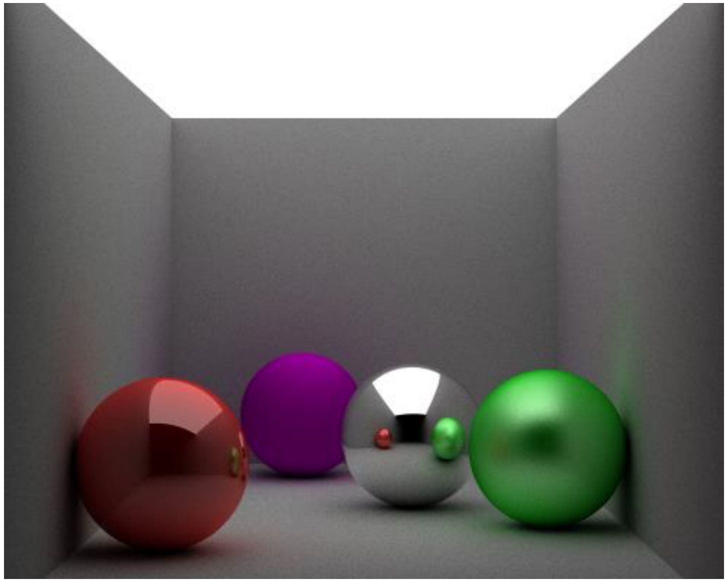
gamedev.ru



cr4.globalspec.com/

- It is possible to model complex shapes using a large number of small triangles or quadrilaterals

Objects

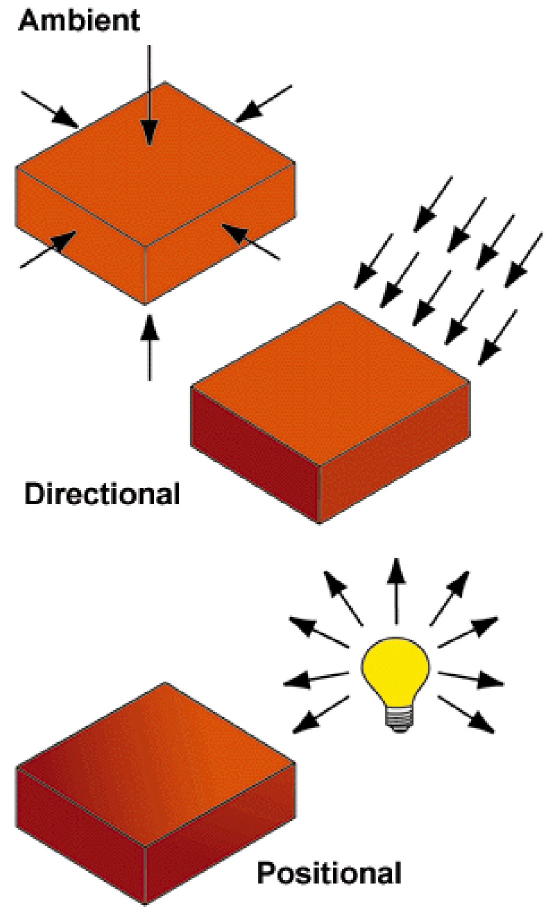


James P. O'Shea
Left to right: High-gloss, diffuse, mirrored, semi-gloss

- Objects may have different materials that affect their appearance

Light Sources

- Light sources provide the illumination in the scene
- The geometrical relationship between objects and light sources may produce shadows
- Typically, three types of light sources are used:
  – Ambient
  – Directional
  – Positional (Point)



Rays

- A ray, or half-line, is a 3D parametric line with a half-open interval, usually $[0, \infty)$

$$\boldsymbol{r}(t) = \underbrace{\boldsymbol{o}}_{\text{origin}} + \underbrace{t}_{\text{parameter}}\underbrace{\boldsymbol{d}}_{\text{direction}}$$

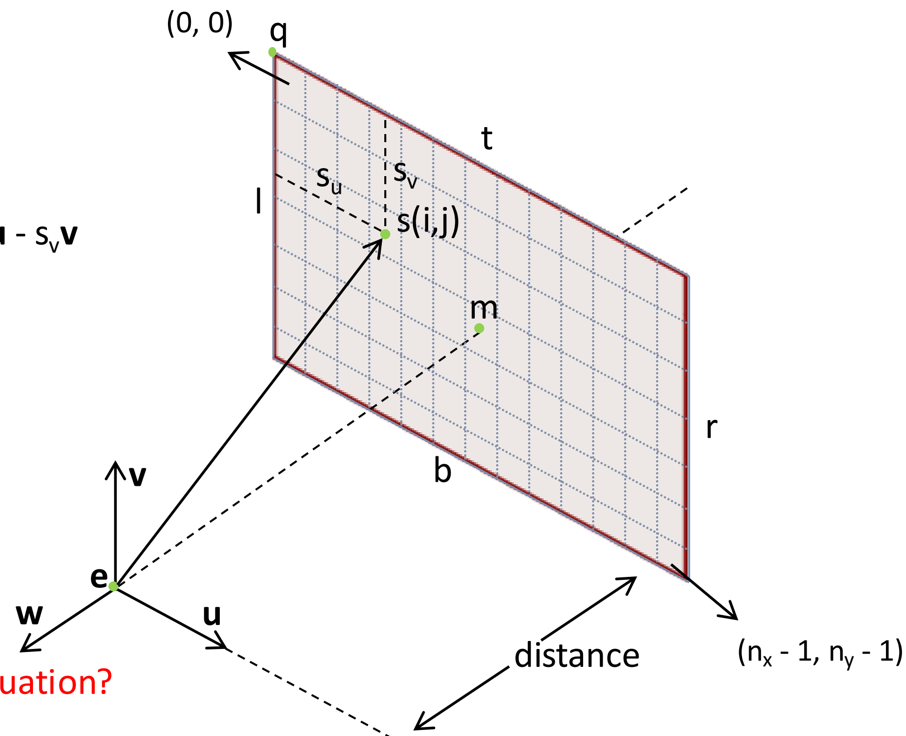
- Both **o** and **d** are elements of $\mathbb{R}^3$ and t is in range $[0, \infty)$
- For a fixed t, **r(**t**)** represents a point on this line

Ray Tracing

- The basic algorithm:

for each pixel **do**
  compute viewing (eye, primary) rays
  find the first object hit by ray and its surface normal **n**
  set pixel color to value computed from hit point, light, and **n**

Computing Eye Rays



m = **e** + -**w**distance

q = **m** + l**u** + t**v**

$\Big\}$ **s** = **q** + s_u**u** - s_v**v**

How to find s_u and s_v?

$$s_u = (i + 0.5)\frac{r - l}{n_x}$$

$$s_v = (j + 0.5)\frac{t - b}{n_y}$$

How to write the final eye ray equation?

r(t) = **e** + (**s** – **e**)t = **e** + **d**t

What information did we use to derive this?

e: location of the camera
distance: camera-image plane distance
n_x: image width
n_y: image height
u, **v**, **w**: camera vectors
r, l, t, b: image plane borders (in **uvw** space)

Some Example Values

- **u** = (1, 0, 0)
- **v** = (0, 1, 0)
- **e** = (0, 0, 0)
- n_x = 1024
- n_y = 768
- l = -1, r = 1
- b = -1, t = 1
- distance = 1

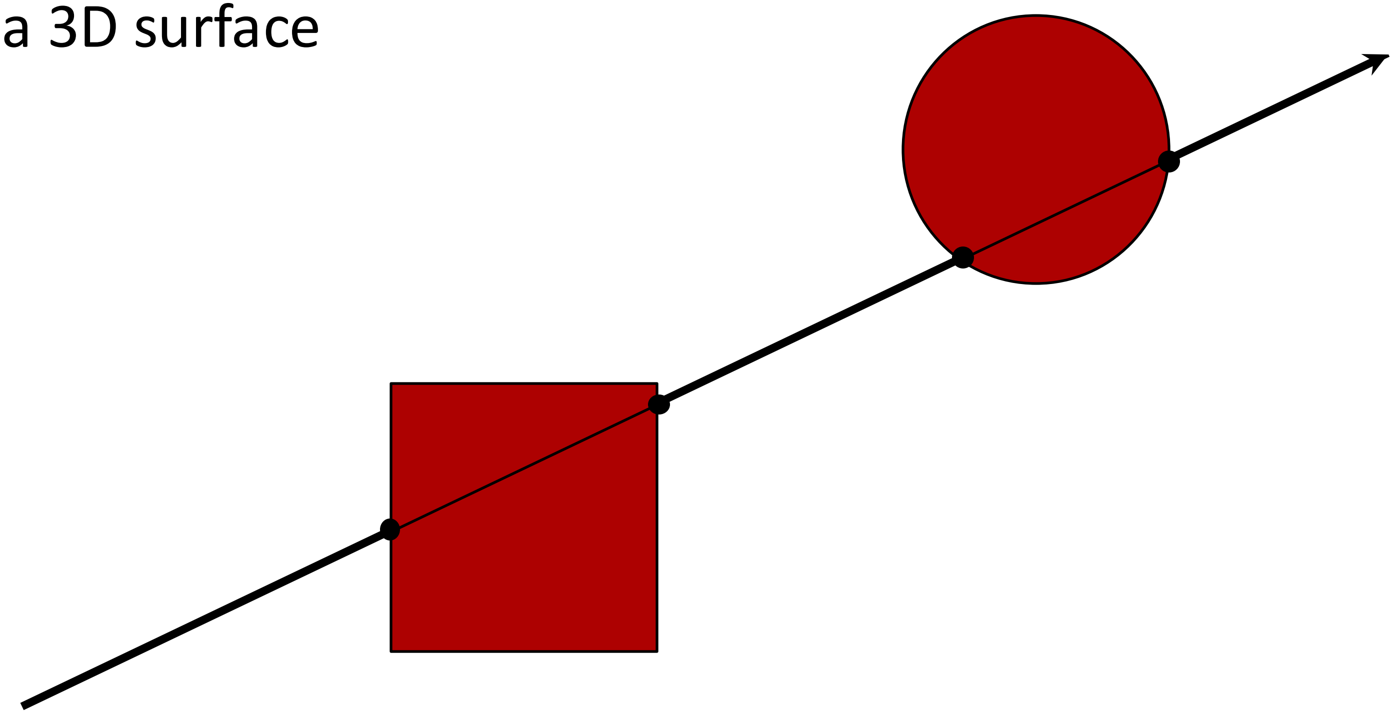
Compute the ray equation passing through the pixel (256, 192):

$$\boldsymbol{r}(\boldsymbol{t}) = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} + t \begin{bmatrix} -0.5 + 1/1024 \\ 0.5 - 1/768 \\ -1 \end{bmatrix}$$

Where is this ray at t = 0, t = 1, t = 2?

Ray-Object Intersections

- **Goal:** To decide at what point, if any, a 3D line (ray) intersects a 3D surface



Parametric Lines

- A 2D line can be represented as: $y - mx - b = 0$. This is called the implicit form
- A parametric 2D line can be represented as:

$$x(t) = 2 + 7t$$
$$y(t) = 1 + 2t$$

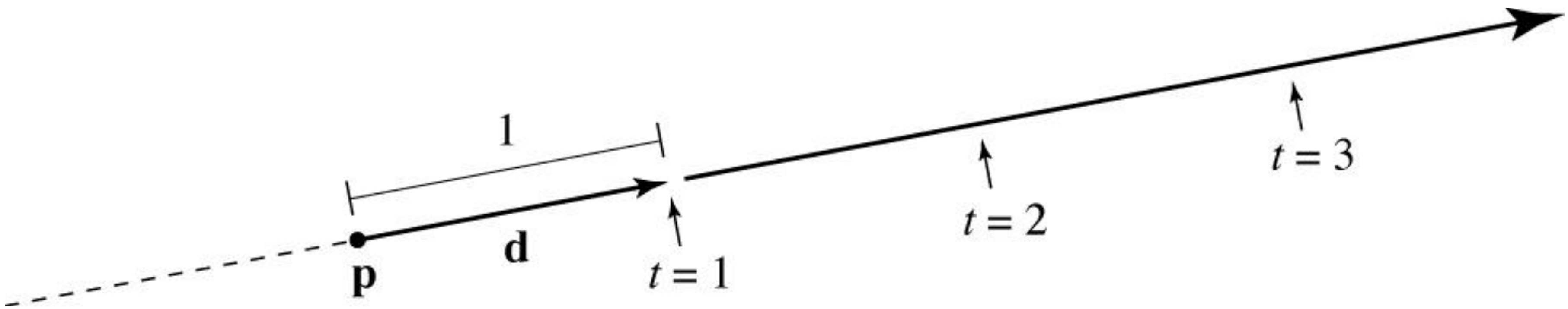
- A 3D parametric line (ray) can be written as:

$$x(t) = 2 + 7t$$
$$y(t) = 1 + 2t$$
$$z(t) = 3 - 5t$$

- Alternatively, in vector form, we can write $\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$ where $\mathbf{o} = (2, 1, 3)$ and $\mathbf{d} = (7, 2, -5)$

Ray (Reminder)

- A ray is a half-line represented by $\boldsymbol{r(t)} = \boldsymbol{o} + t\boldsymbol{d}$ with $t \geq 0$.



- We want to know if a ray intersects an object with t in the interval $[t_{min}, t_{max}]$
  - If $t < t_{min}$, the object is too close (maybe in front of the image plane).
  - If $t > t_{max}$, the object is too far (outside the range we want to consider).

Implicit Surfaces

- Rays will intersect surfaces, so we need to know how to represent surfaces
- In implicit form a surface can be written as $f(x, y, z) = 0$
- Why is it called implicit?
  - You can test whether a point is on the surface, but you cannot generate points on the surface
- Another way to write: $f(\mathbf{p}) = 0$ where $\mathbf{p} = (x, y, z)$
- A ray will intersect this surface if:

$$f(\mathbf{r}(t)) = f(\mathbf{o} + t\mathbf{d}) = 0$$

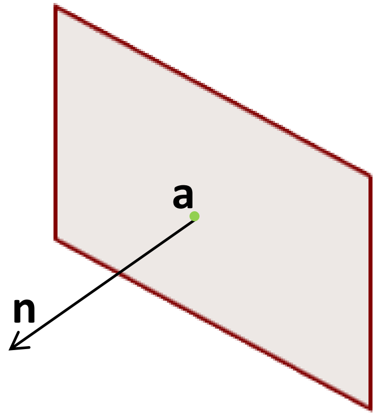
Ray-Plane Intersection

- Consider the plane equation written in vector form as:

$$(\boldsymbol{p} - \boldsymbol{a}) \bullet \boldsymbol{n} = 0$$

If you expand this, you'll get the familiar $Ax + By + Cz + D = 0$ equation

- Here, ***a*** is a point on the plane, ***n*** is the normal vector of the plane



- ***a*** and ***n**,* are known quantities and **p** is the variable.

Ray-Plane Intersection

- Simply plug $\boldsymbol{r}(t) = \boldsymbol{o} + t\boldsymbol{d}$ into the previous equation:

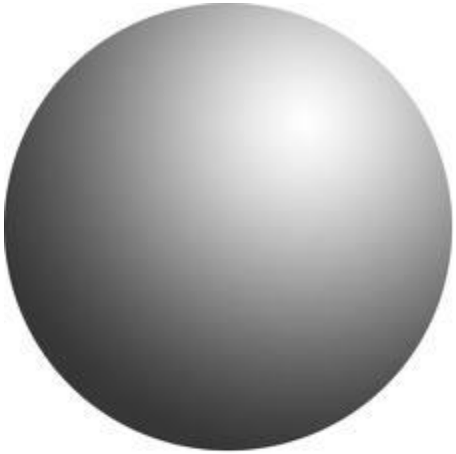
$$(\boldsymbol{o} + t\boldsymbol{d} - \boldsymbol{a}).\boldsymbol{n} = 0$$

- Solving for t, we get:

$$t = (\boldsymbol{a} - \boldsymbol{o}).\boldsymbol{n} \,/\, (\boldsymbol{d}.\boldsymbol{n})$$

- If t is in $[tmin, tmax]$, the ray hits the plane and it is within the limits of our desired viewing range
- What if $\boldsymbol{d}.\boldsymbol{n} = 0$?

Ray-Sphere Intersection



- A sphere can be represented as:

$$(x - c_x)^2 + (y - c_y)^2 + (z - c_z)^2 - R^2 = 0$$

- where $\boldsymbol{c} = (c_x, c_y, c_z)$ is the center and R is the radius
- In vector form, we can rewrite this as:

$$(\mathbf{p} - \mathbf{c}).(\mathbf{p} - \mathbf{c}) - R^2 = 0$$

- Again, plug in the ray equation to find t:

$$(\mathbf{o} + t\mathbf{d} - \mathbf{c}).(\mathbf{o} + t\mathbf{d} - \mathbf{c}) - R^2 = 0$$

Ray-Sphere Intersection

- This gives:

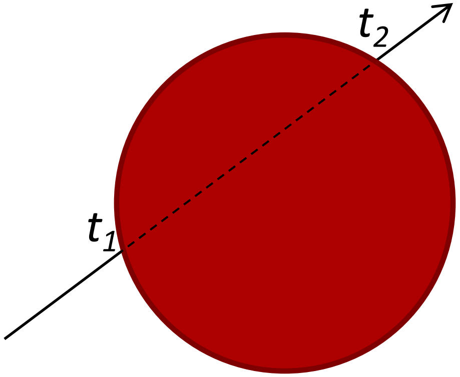
$$(\mathbf{d}.\mathbf{d})t^2 + 2\mathbf{d}.(\mathbf{o} - \mathbf{c})t + (\mathbf{o} - \mathbf{c}).(\mathbf{o} - \mathbf{c}) - R^2 = 0$$

- Note that, this is a quadratic equation in *t*:

$$At^2 + Bt + C = 0$$

- The solution is:

$$t = \frac{-d.(o-c) \pm \sqrt{(d.(o-c))^2 - (d.d)((o-c).(o-c) - R^2)}}{d.d}$$



- What if the discriminant is less than zero?

Ray-Triangle Intersection

- So far, we have been using implicit equations to represent surfaces: $f(x, y, z) = 0$
- Ray-triangle intersection is more efficient if the triangle is represented using a parametric form:

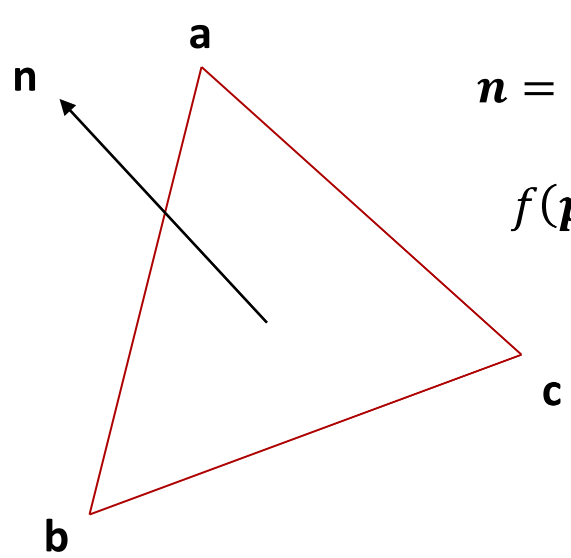
$$x = f(u, v)$$
$$y = g(u, v)$$
$$z = h(u, v)$$

Ray-Triangle Intersection

- Two techniques are possible:
  - Lengthy but simple
  - Shorter but somewhat more complex

Lengthy But Simple Method

- First, intersect the ray with the *plane* of the triangle:
  - The plane normal can be found by cross product
  - The plane equation can be found from the normal and a point

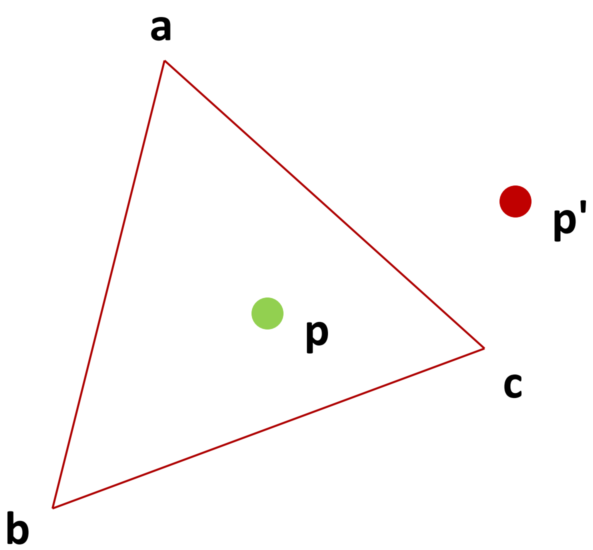


$$\boldsymbol{n} = (\boldsymbol{c} - \boldsymbol{b}) \text{ x } (\mathbf{a} - \mathbf{b})$$

$$f(\boldsymbol{p}) = (\boldsymbol{p} - \boldsymbol{a}).\boldsymbol{n} = 0$$

Lengthy But Simple Method

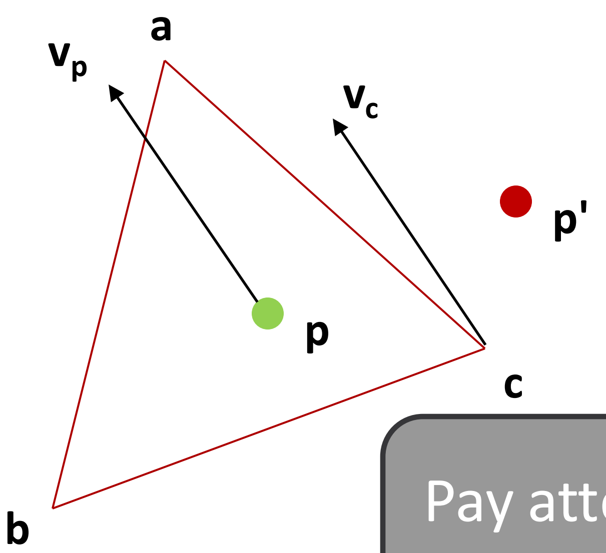
- If the ray intersects triangle's plane, we need to determine if it is inside the triangle
  - How to make an inside check?



- If $\boldsymbol{p}$ is on the same side of $\overline{\boldsymbol{ab}}$ as $\boldsymbol{c}$ and
- If $\boldsymbol{p}$ is on the same side of $\overline{\boldsymbol{bc}}$ as $\boldsymbol{a}$ and
- If $\boldsymbol{p}$ is on the same side of $\overline{\boldsymbol{ac}}$ as $\boldsymbol{b}$
- Then $\boldsymbol{p}$ is inside the triangle
- Otherwise, it is outside the triangle

Lengthy But Simple Method

- To check this, we use cross and dot products



$$\boldsymbol{v}_p = (\boldsymbol{p} - \boldsymbol{b}) \text{ x } (\mathbf{a} - \mathbf{b})$$

$$\boldsymbol{v}_c = (\boldsymbol{c} - \boldsymbol{b}) \text{ x } (\mathbf{a} - \mathbf{b})$$

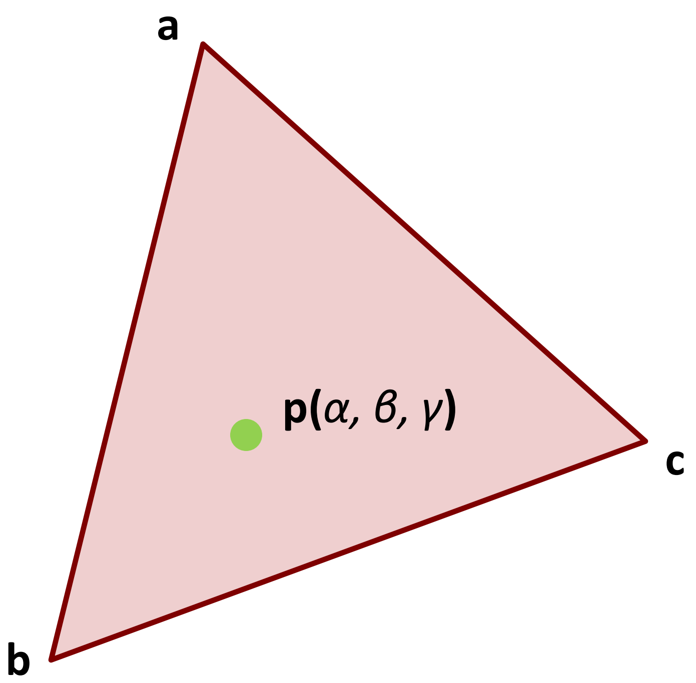
If $\boldsymbol{v_p}.\boldsymbol{vc} > 0$ then $\boldsymbol{p}$ and $\boldsymbol{c}$ are on the same side of the line $\overline{\boldsymbol{ab}}$

Repeat this process for all sides of the triangle!

Pay attention to the order of the terms in cross-products!

Barycentric Coordinates

- The alternative method uses barycentric coordinates
- Any point inside the triangle plane can be parametrized by these three coordinates with the following conditions:



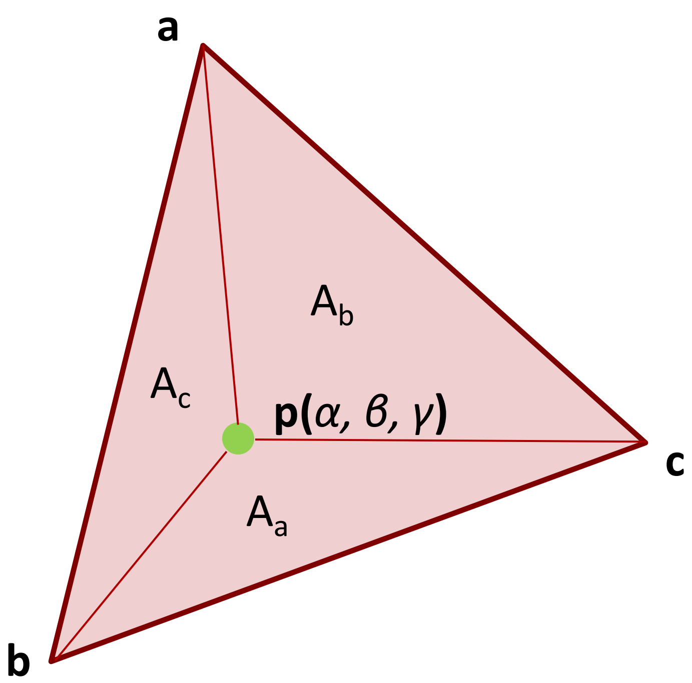
$$\boldsymbol{p}(\alpha, \beta, \gamma) = \alpha\boldsymbol{a} + \beta\boldsymbol{b} + \gamma\boldsymbol{c}$$

with the constraints

$$0 < \alpha < 1$$
$$0 < \beta < 1$$
$$0 < \gamma < 1$$

Barycentric Coordinates

- The coordinates can be computed from area ratios:



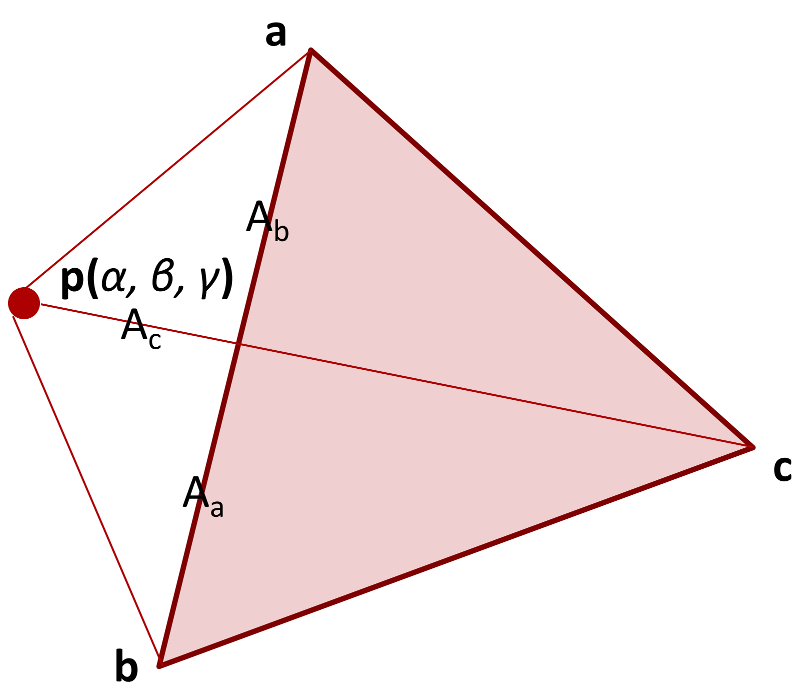
$$\alpha = A_a / A$$
$$\beta = A_b / A$$
$$\gamma = A_c / A$$

where

$$A = A_a + A_b + A_c$$

Barycentric Coordinates

- Outside points also have barycentric coordinates but they can be negative (an outside area is considered to be a negative area):



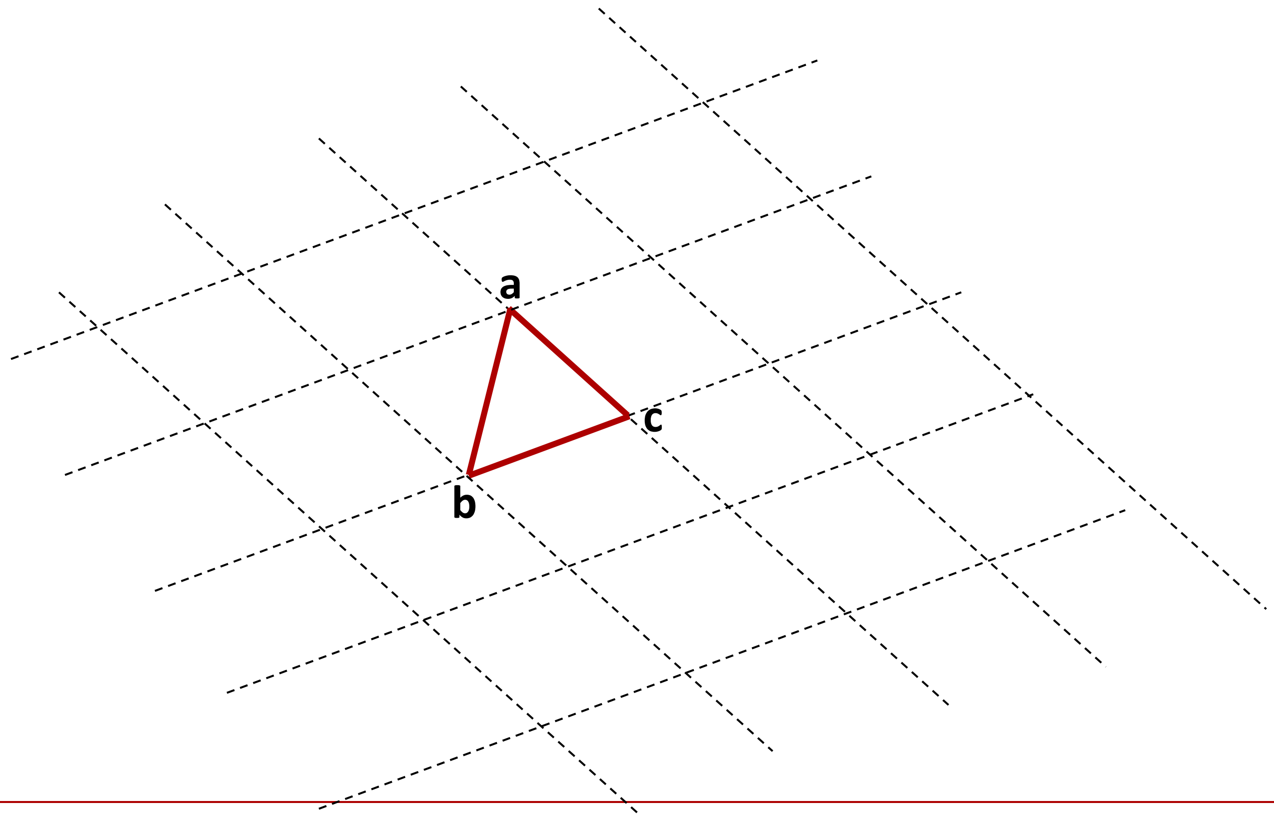
$$\alpha = A_a / A$$
$$\beta = A_b / A$$
$$\gamma = A_c / A$$

where

$$A = A_a + A_b + A_c$$

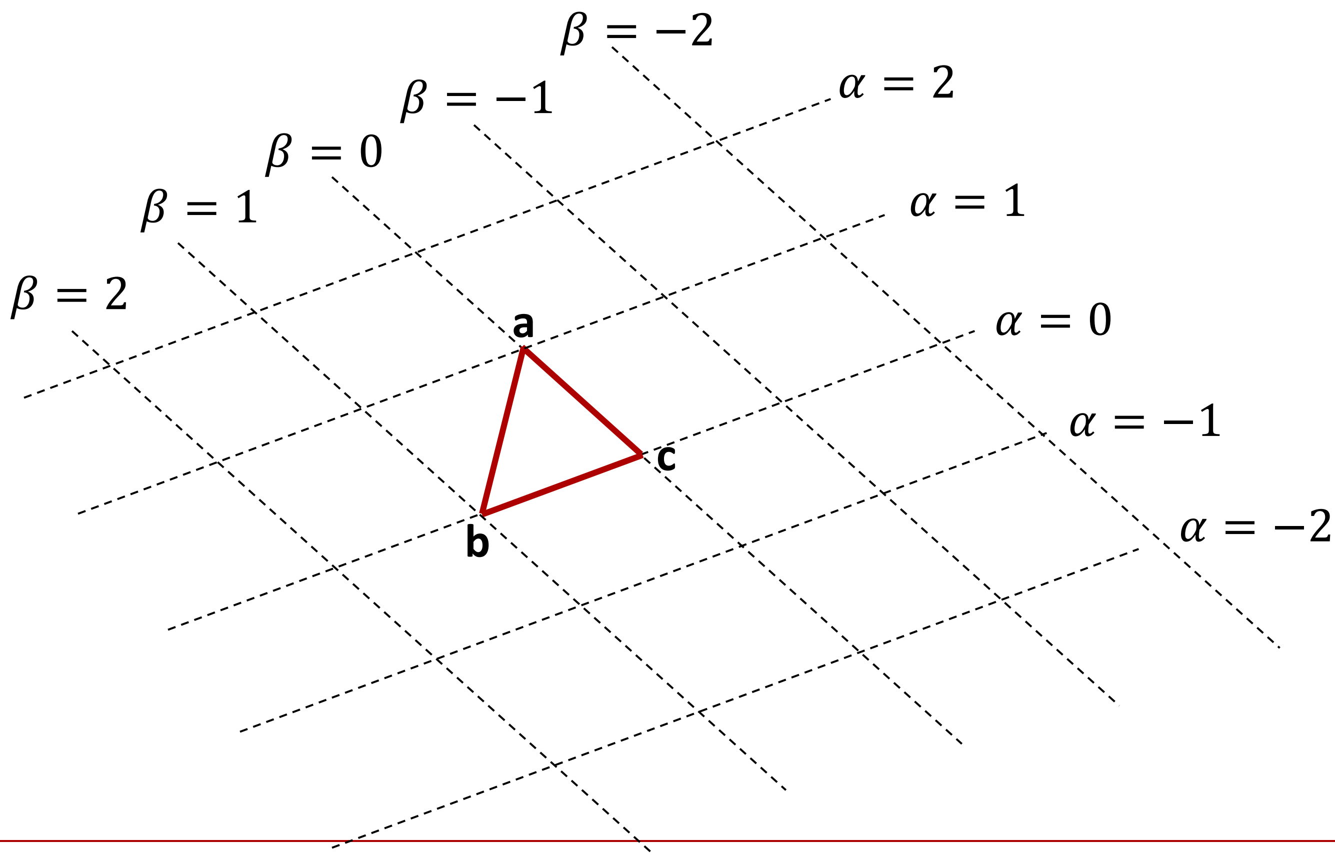
Barycentric Coordinates

- Barycentric coordinates can be shown on a grid that is aligned with the edges of the triangles



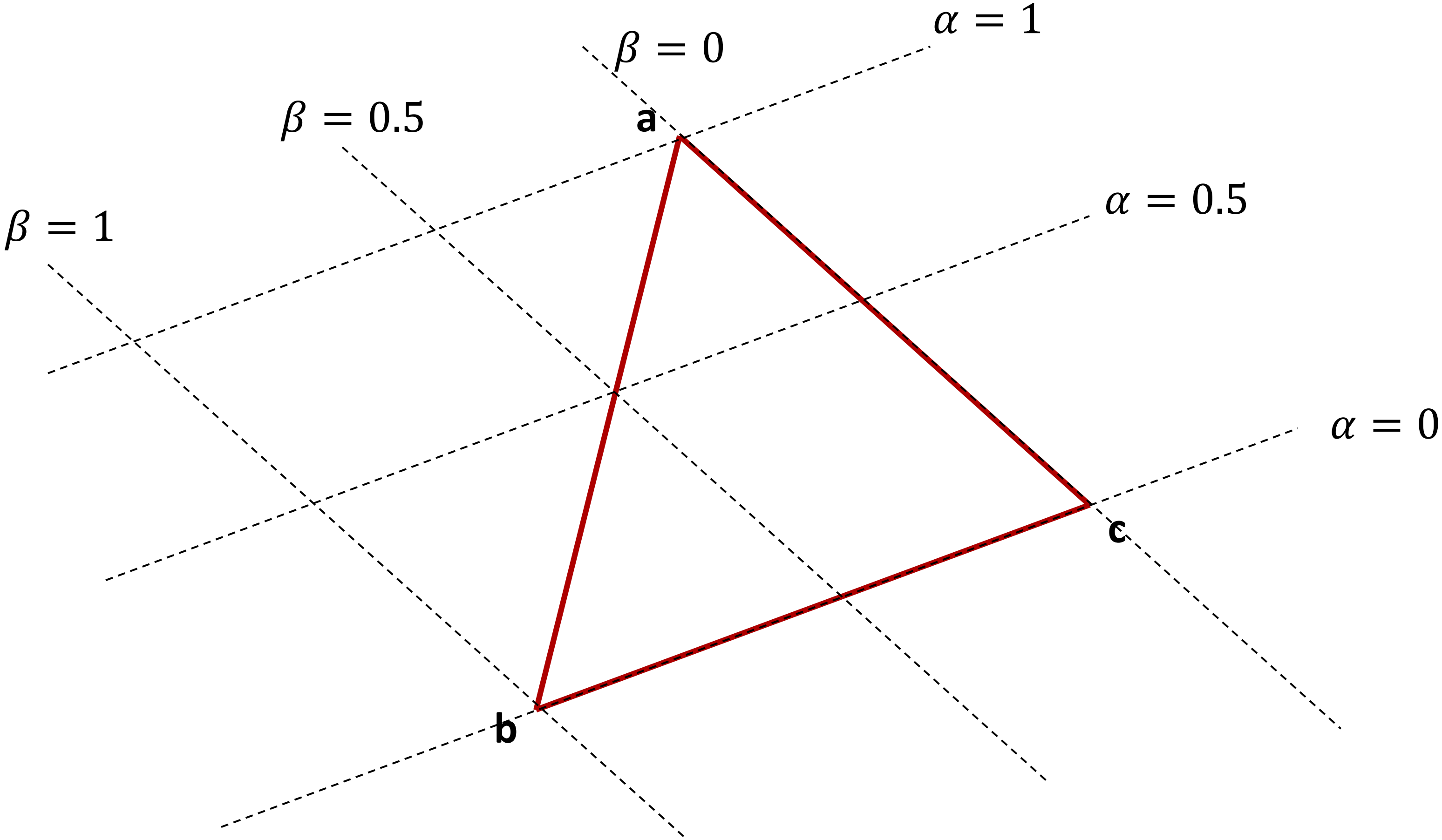
Barycentric Coordinates

- Any point on this grid is characterized by two coordinates



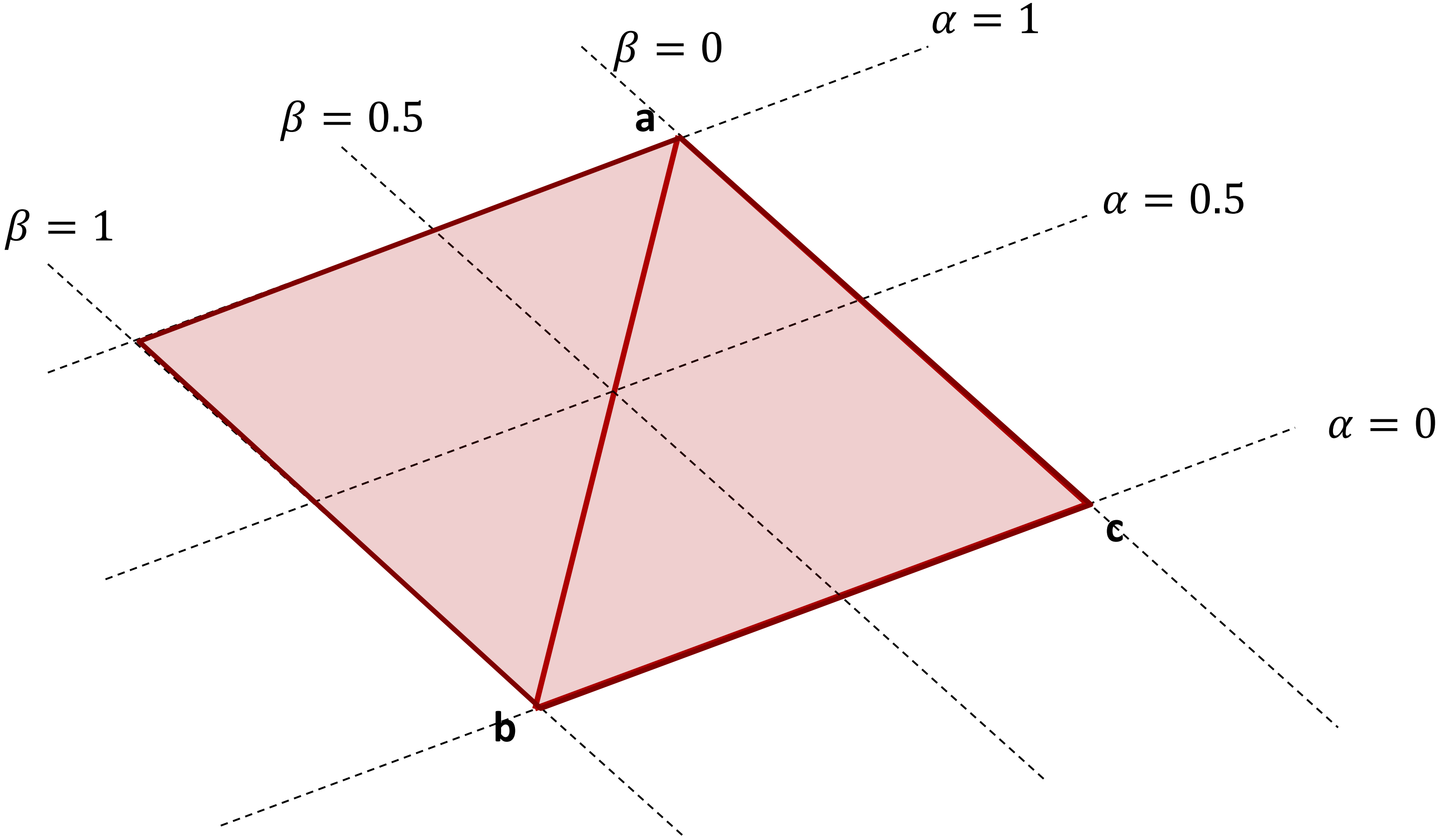
Barycentric Coordinates

- Zooming into our triangle:



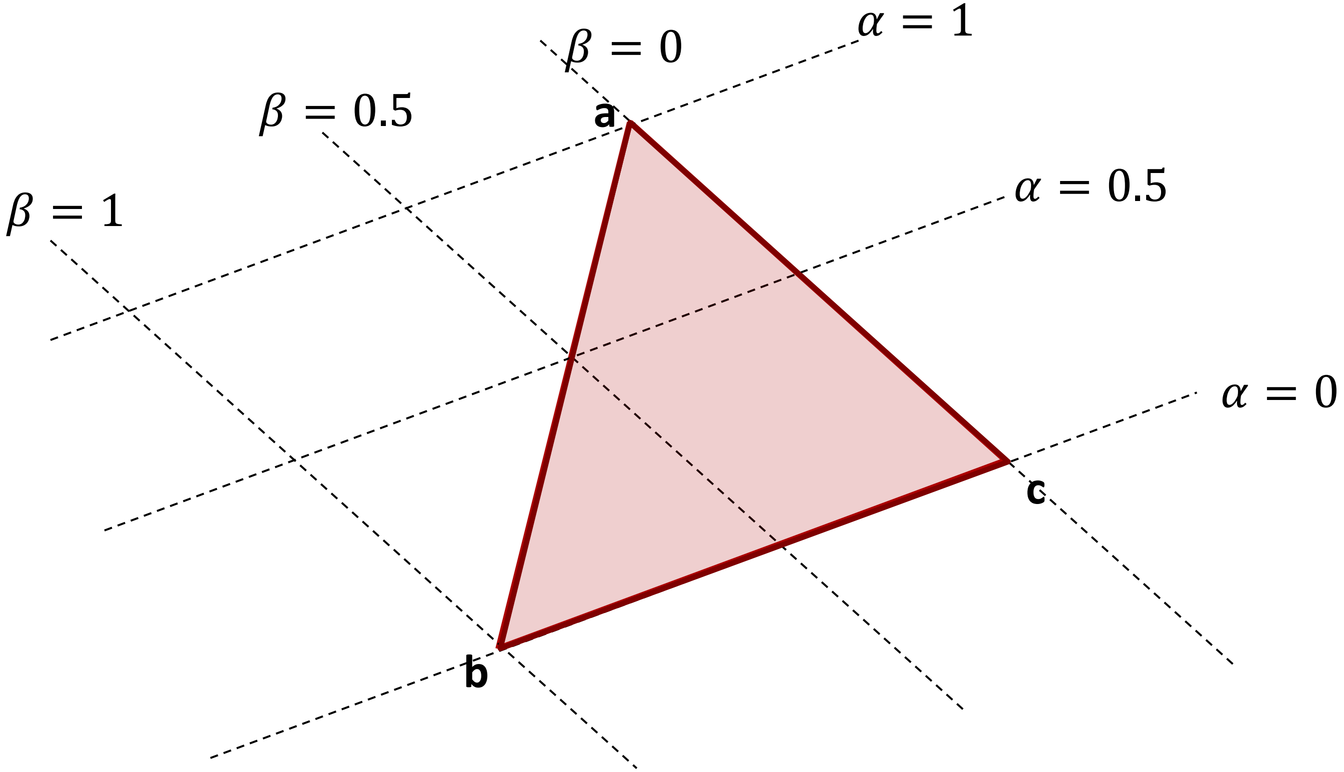
Barycentric Coordinates

- If $0 \leq \alpha \leq 1$ and $0 \leq \beta \leq 1$ then we are in this region:



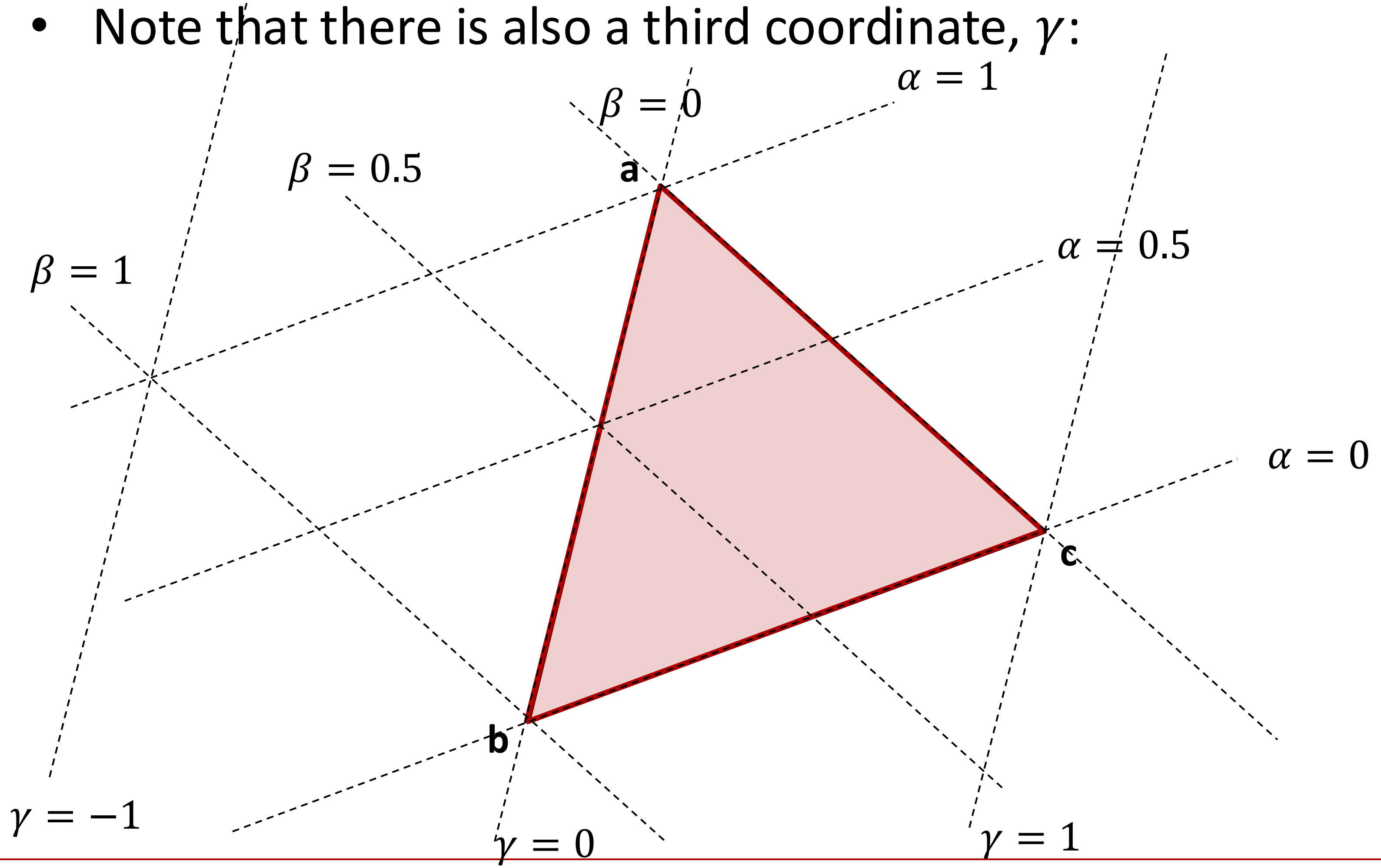
Barycentric Coordinates

- Furthermore, if $\mathbf{0} \leq \boldsymbol{\alpha} + \boldsymbol{\beta} \leq \mathbf{1}$ and $\mathbf{0} \leq \boldsymbol{\alpha}$ and $\mathbf{0} \leq \boldsymbol{\beta}$ then we are inside the triangle



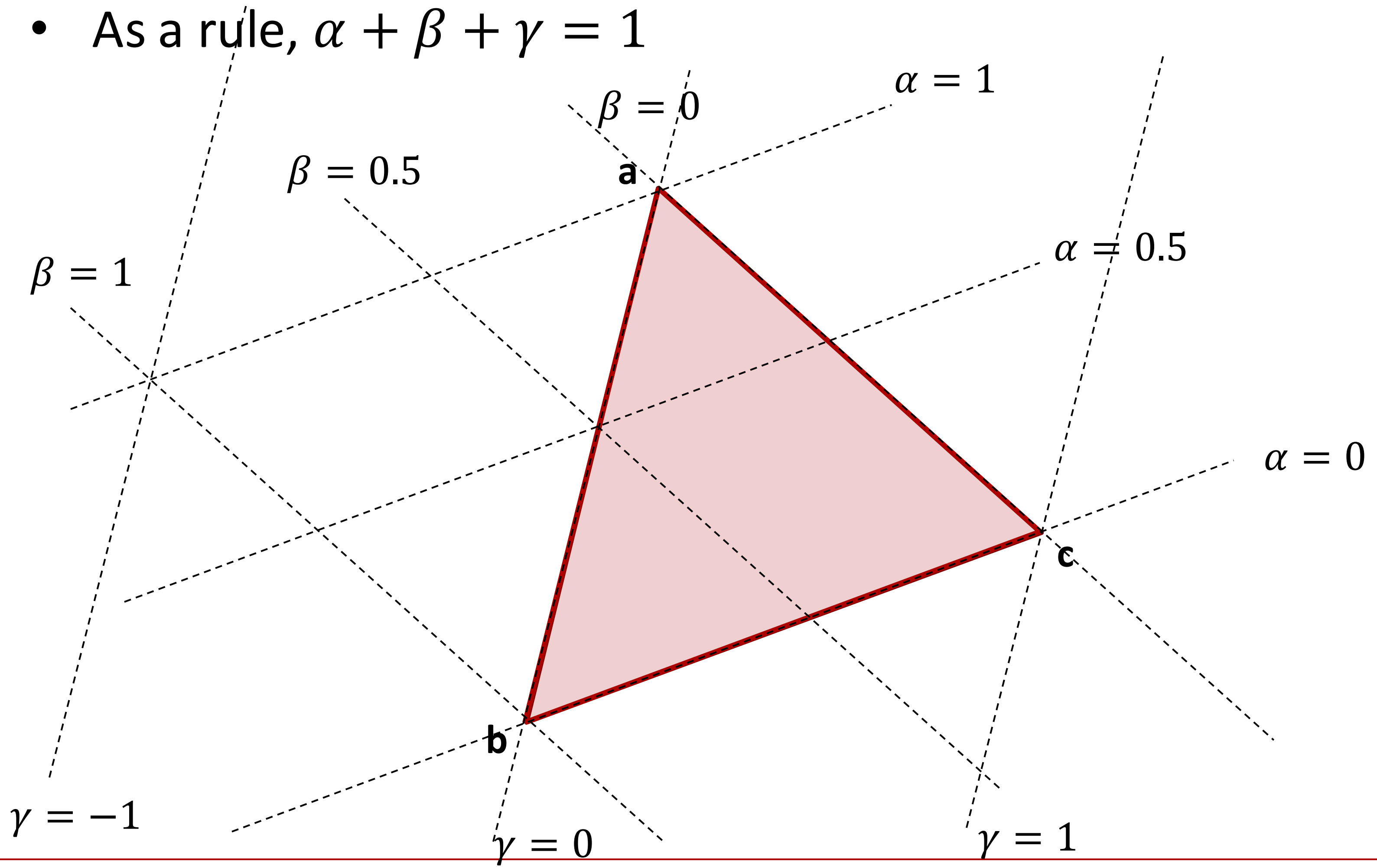
Barycentric Coordinates

- Note that there is also a third coordinate, γ:



Barycentric Coordinates

- As a rule, $\alpha + \beta + \gamma = 1$



Back to Intersection

- Note that we can eliminate one of the parameters:

$$\alpha = 1 - \beta - \gamma$$

$$\mathbf{p}(\beta, \gamma) = \mathbf{a} + \beta(\mathbf{b} - \mathbf{a}) + \gamma(\mathbf{c} - \mathbf{a})$$

- The point ***p*** is inside (or on) the triangle if and only if:

$$\beta + \gamma \leq 1$$
$$0 \leq \beta$$
$$0 \leq \gamma$$

- Plug the ray equation $\mathbf{r}(t) = \mathbf{o} + t\mathbf{d}$ into **p**:

$$\mathbf{o} + t\mathbf{d} = \mathbf{a} + \beta(\mathbf{b} - \mathbf{a}) + \gamma(\mathbf{c} - \mathbf{a})$$

Ray-Triangle Intersection

- How to solve for t?
- Expand from the vector form into individual coordinates:

$$o_x + td_x = a_x + \beta(b_x - a_x) + \gamma(c_x - a_x)$$
$$o_y + td_y = a_y + \beta(b_y - a_y) + \gamma(c_y - a_y)$$
$$o_z + td_z = a_z + \beta(b_z - a_z) + \gamma(c_z - a_z)$$

- The unknowns here are t, β, and γ
- We have 3 equations and 3 unknowns

Ray-Triangle Intersection

- Rewrite this system in matrix form:

$$\begin{bmatrix} a_x - b_x & a_x - c_x & d_x \\ a_y - b_y & a_y - c_y & d_y \\ a_z - b_z & a_z - c_z & d_z \end{bmatrix} \begin{bmatrix} \beta \\ \gamma \\ t \end{bmatrix} = \begin{bmatrix} a_x - o_x \\ a_y - o_y \\ a_z - o_z \end{bmatrix}$$

$$A \begin{bmatrix} \beta \\ \gamma \\ t \end{bmatrix} = \begin{bmatrix} a_x - o_x \\ a_y - o_y \\ a_z - o_z \end{bmatrix}$$

- And solve for *t*, *β*, and *γ* using Cramer's rule

Ray-Triangle Intersection

- Cramer's rule:

$$\beta = \frac{\begin{vmatrix} a_x - o_x & a_x - c_x & d_x \\ a_y - o_y & a_y - c_y & d_y \\ a_z - o_z & a_z - c_z & d_z \end{vmatrix}}{|A|} \qquad \gamma = \frac{\begin{vmatrix} a_x - b_x & a_x - o_x & d_x \\ a_y - b_y & a_y - o_y & d_y \\ a_z - b_z & a_z - o_z & d_z \end{vmatrix}}{|A|}$$

$$t = \frac{\begin{vmatrix} a_x - b_x & a_x - c_x & a_x - o_x \\ a_y - b_y & a_y - c_y & a_y - o_y \\ a_z - b_z & a_z - c_z & a_z - o_z \end{vmatrix}}{|A|}$$

where |.| denotes the determinant

Finding the Determinant

- If A is equal to:

$$A = \begin{bmatrix} a & d & g \\ b & e & h \\ c & f & i \end{bmatrix}$$

- Then |A| is given by:

$$|A| = a(ei - hf) + b(gf - di) + c(dh - eg)$$

Ray-triangle Intersection

- Use this to find the determinants of the other terms and compute *t*, *β*, and *γ*.
- The ray will intersect the triangle if:

$$t_{min} \leq t \leq t_{max}$$
$$\beta + \gamma \leq 1$$
$$0 \leq \beta$$
$$0 \leq \gamma$$

- Ray-triangle intersection is the most important as any complex object can be represented using a set of triangles

Ray Tracing Algorithm

- The basic algorithm (reminder):

for each pixel **do**
compute viewing (eye, primary) rays
find the first object hit by ray and its surface normal **n**
set pixel color to value computed from hit point, light, and **n**

Complex Models

- Bunny model composed of 725,000 triangles

From Stanford University

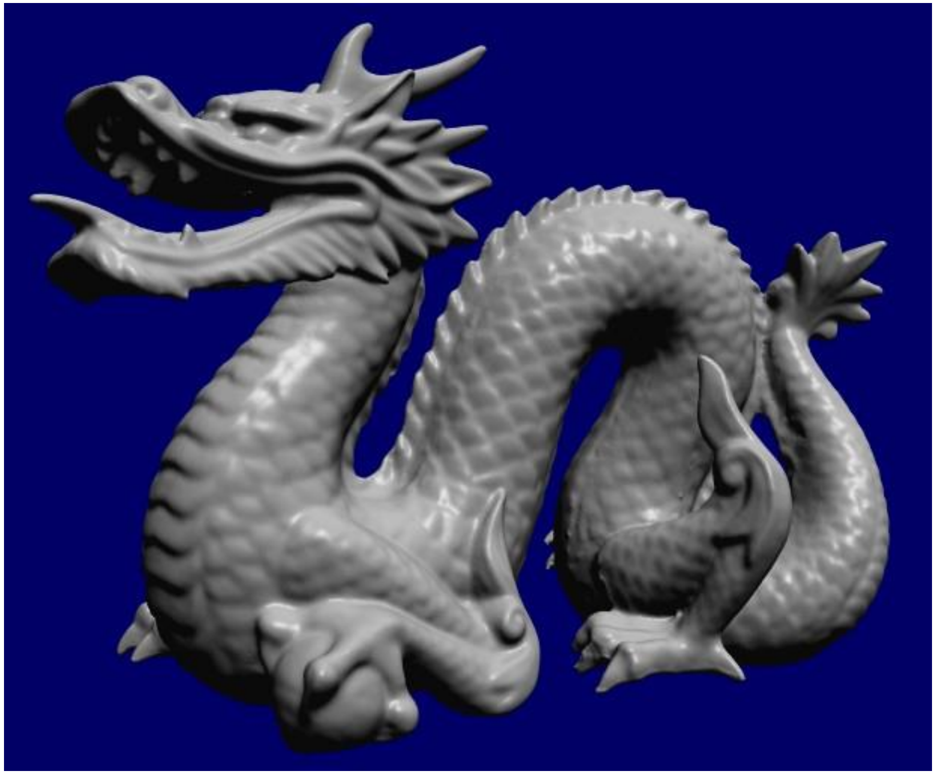
Complex Models

- Buddha model composed of 9,200,000 triangles

From Stanford University

Complex Models

- Dragon model composed of 5,500,000 triangles



From Stanford University

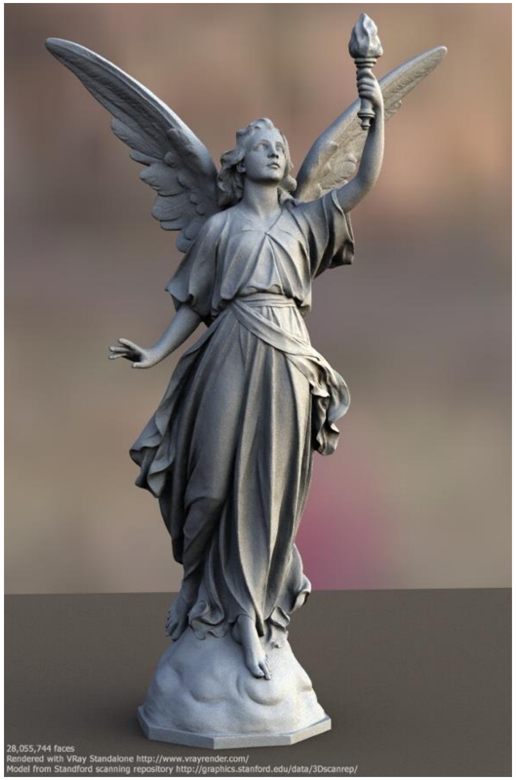
Complex Models

- Armadillo model composed of 7,500,000 triangles

From Stanford University

Complex Models

- Lucy model composed of 116,000,000 triangles



From Stanford University

Intersection Cost

- Let’s compute how many intersection tests we need to perform to render a model composed of 1,000,000 triangles with an image size of 1024x1024
- $1{,}000{,}000 * 1024 * 1024 \approx 1{,}000{,}000{,}000{,}000$ (one trillion)
- And this is just the intersection – realistic shading is generally more costly (next week)
- That’s why ray tracing is very slow
- However, ray tracing can be accelerated by using:
  - Multiple computers
  - GPUs
  - Acceleration structures

Realism

- Intersection tests give us the surface position that is hit by a ray
- To create realistic images, we need to compute realistic models of light-surface interaction at that point on the surface
- This will be the topic of the next week

From ACM Siggraph