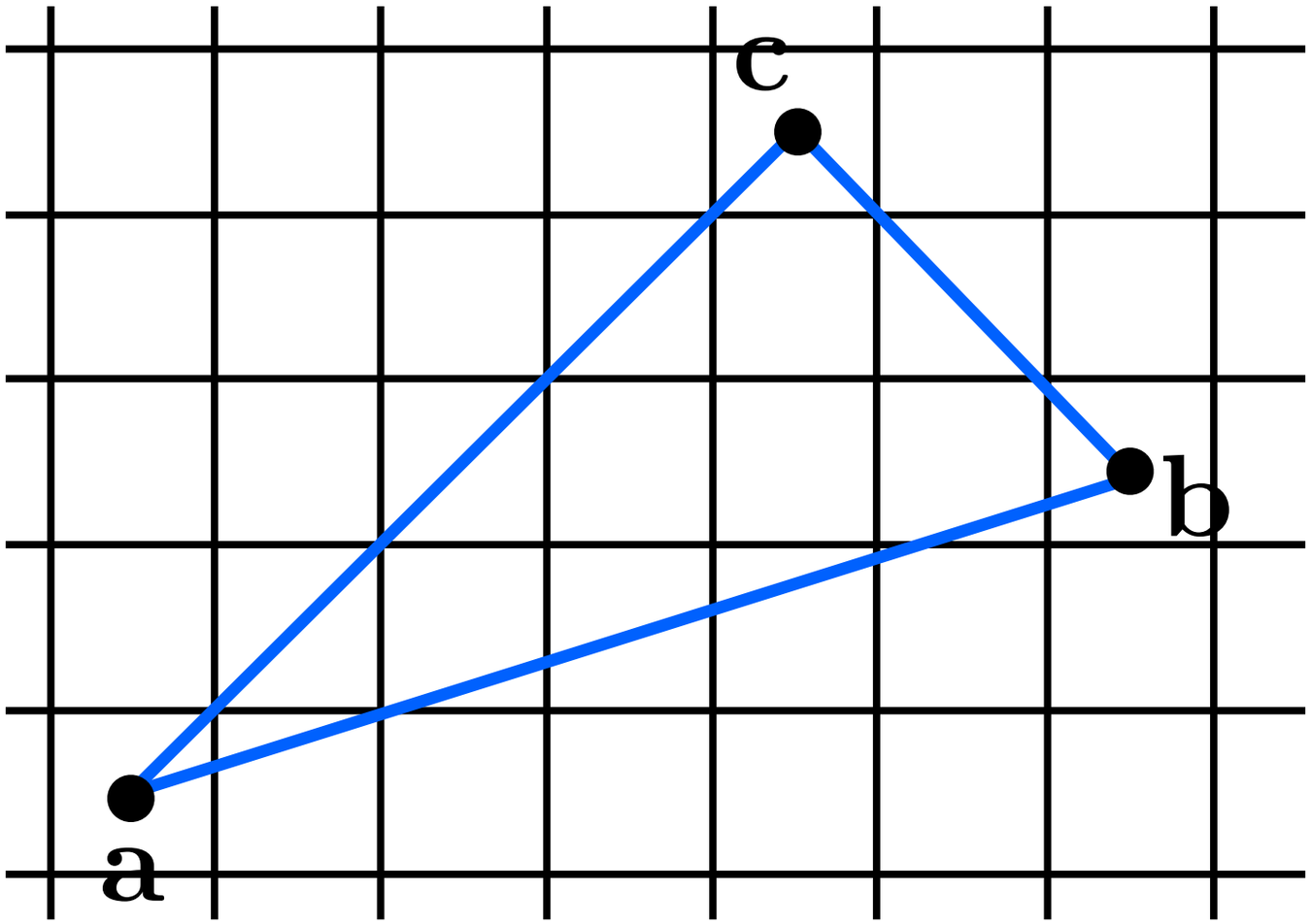
CS130 : Computer Graphics

Lecture 4: Rasterizing Triangles and Graphics Pipeline (cont.)

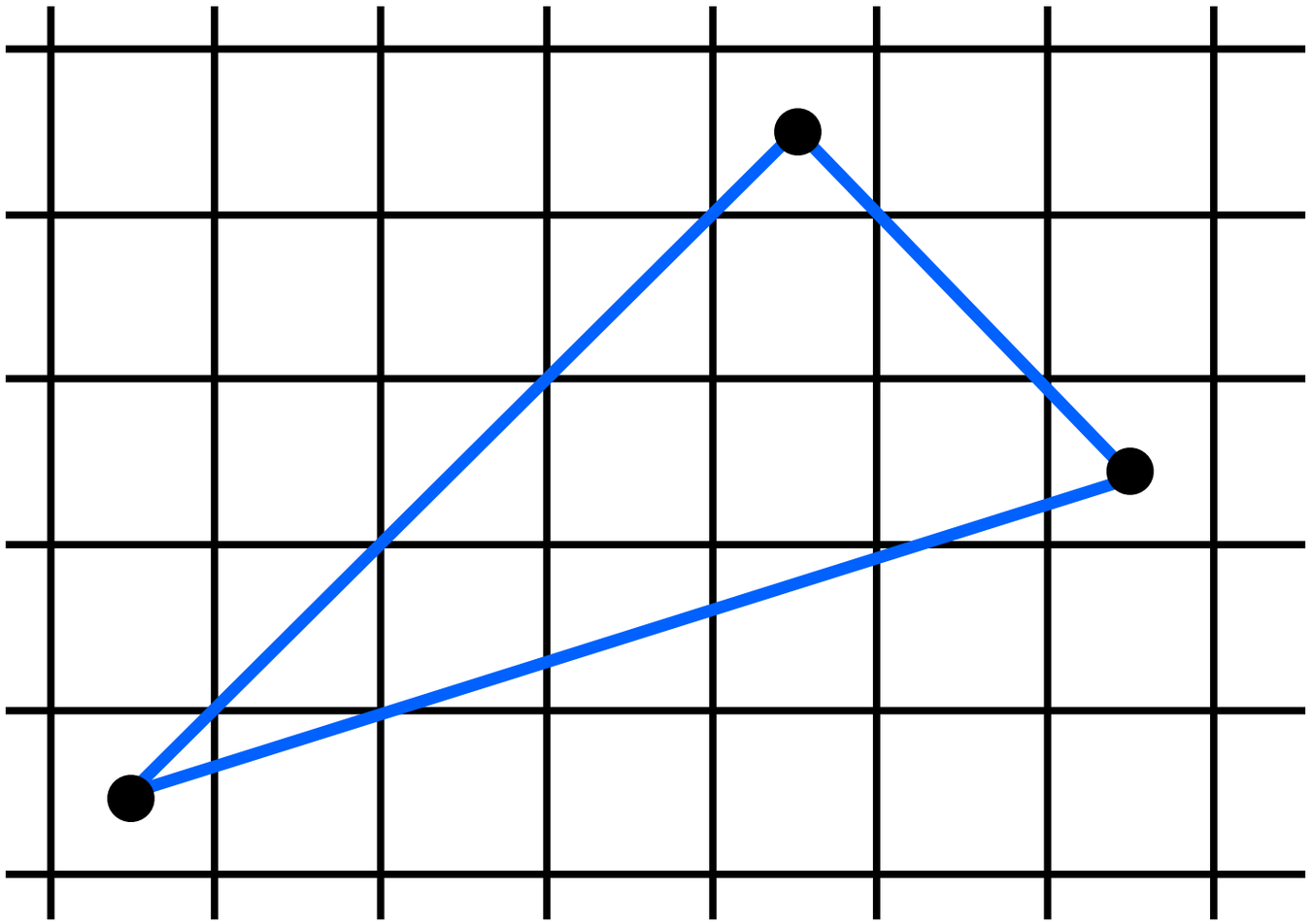
Tamar Shinar
Computer Science & Engineering
UC Riverside

Triangle rasterization

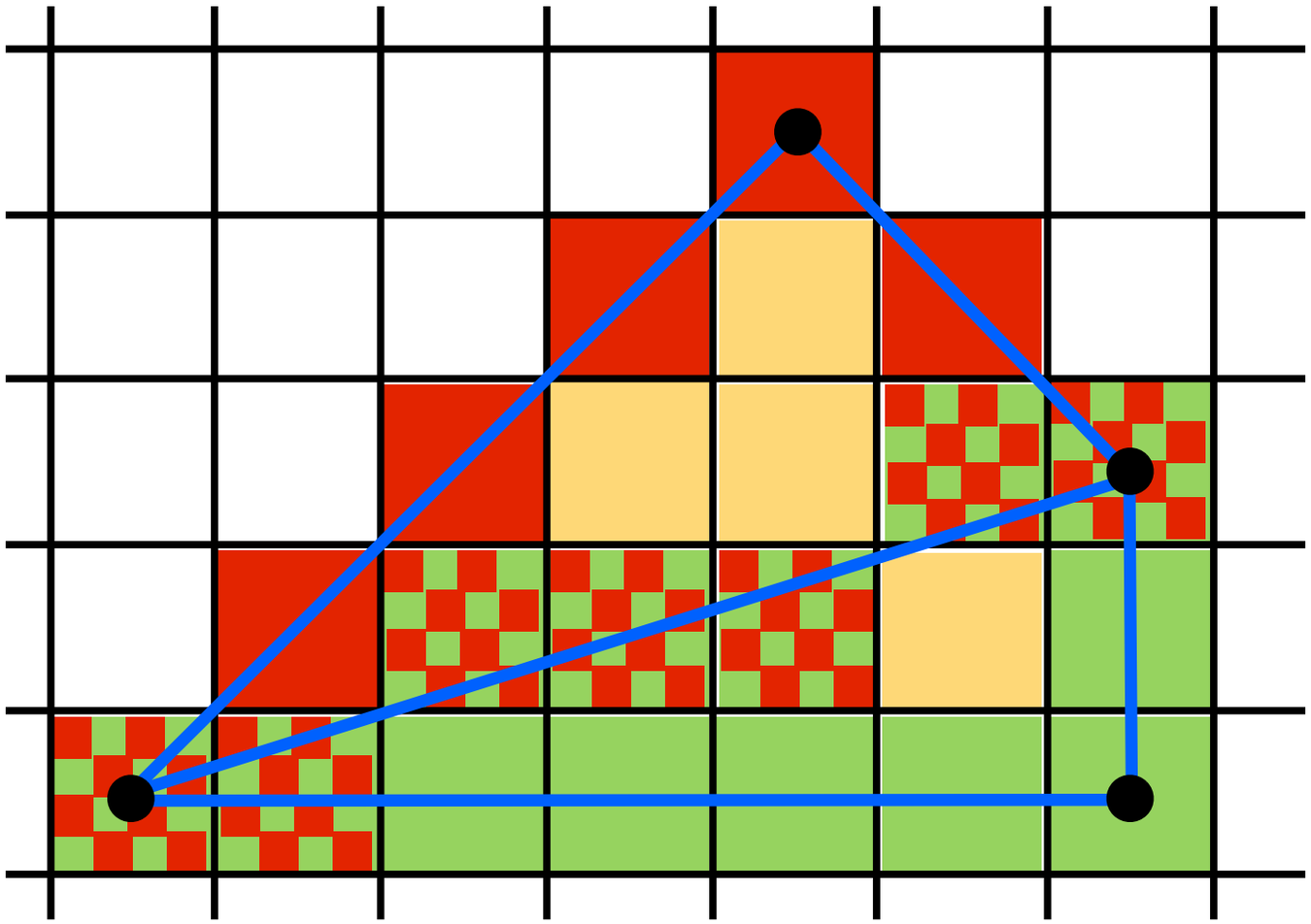
Which pixels should be used to approximate a triangle?



Triangle rasterization issues

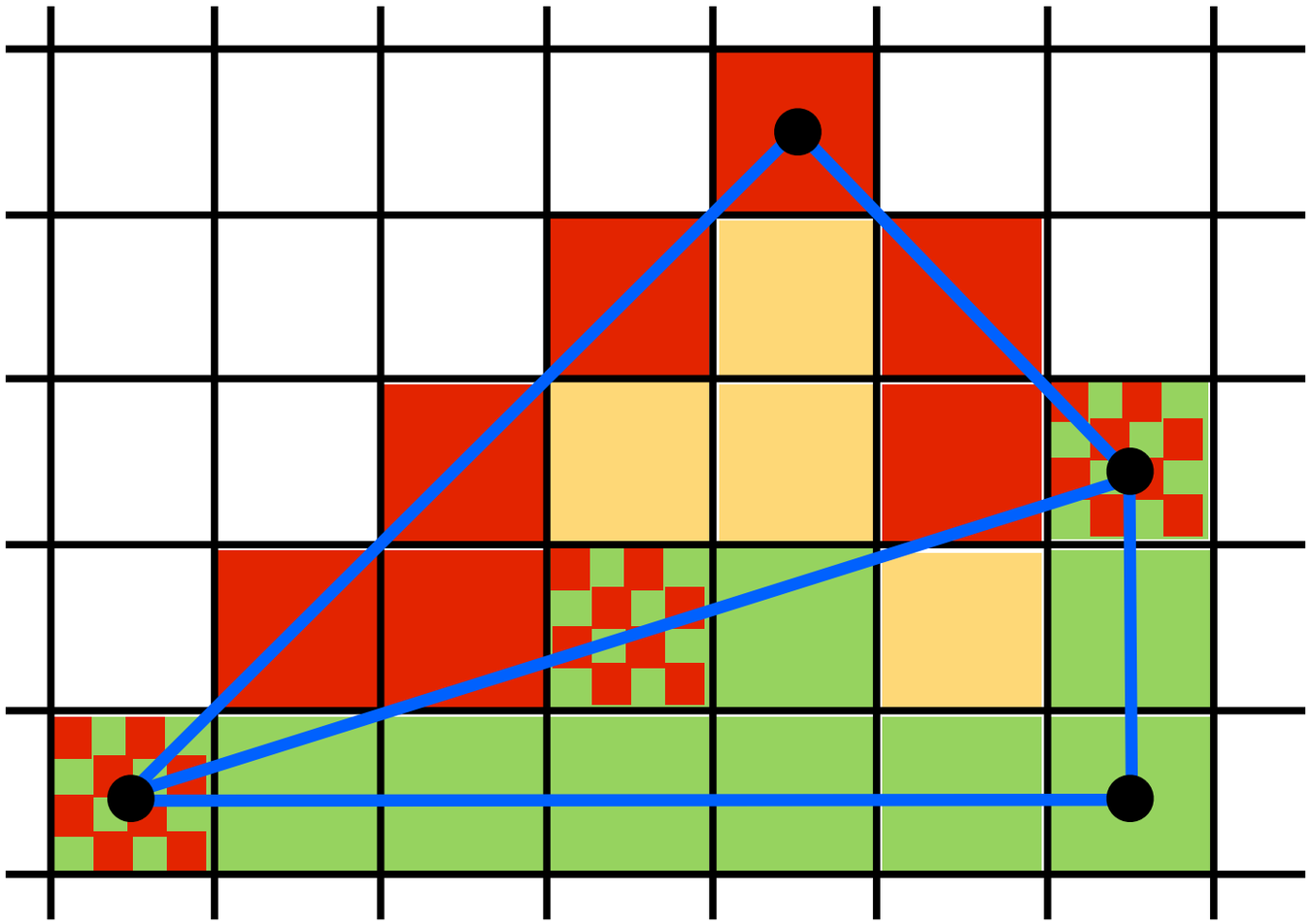


Which pixels should be used to approximate a triangle?



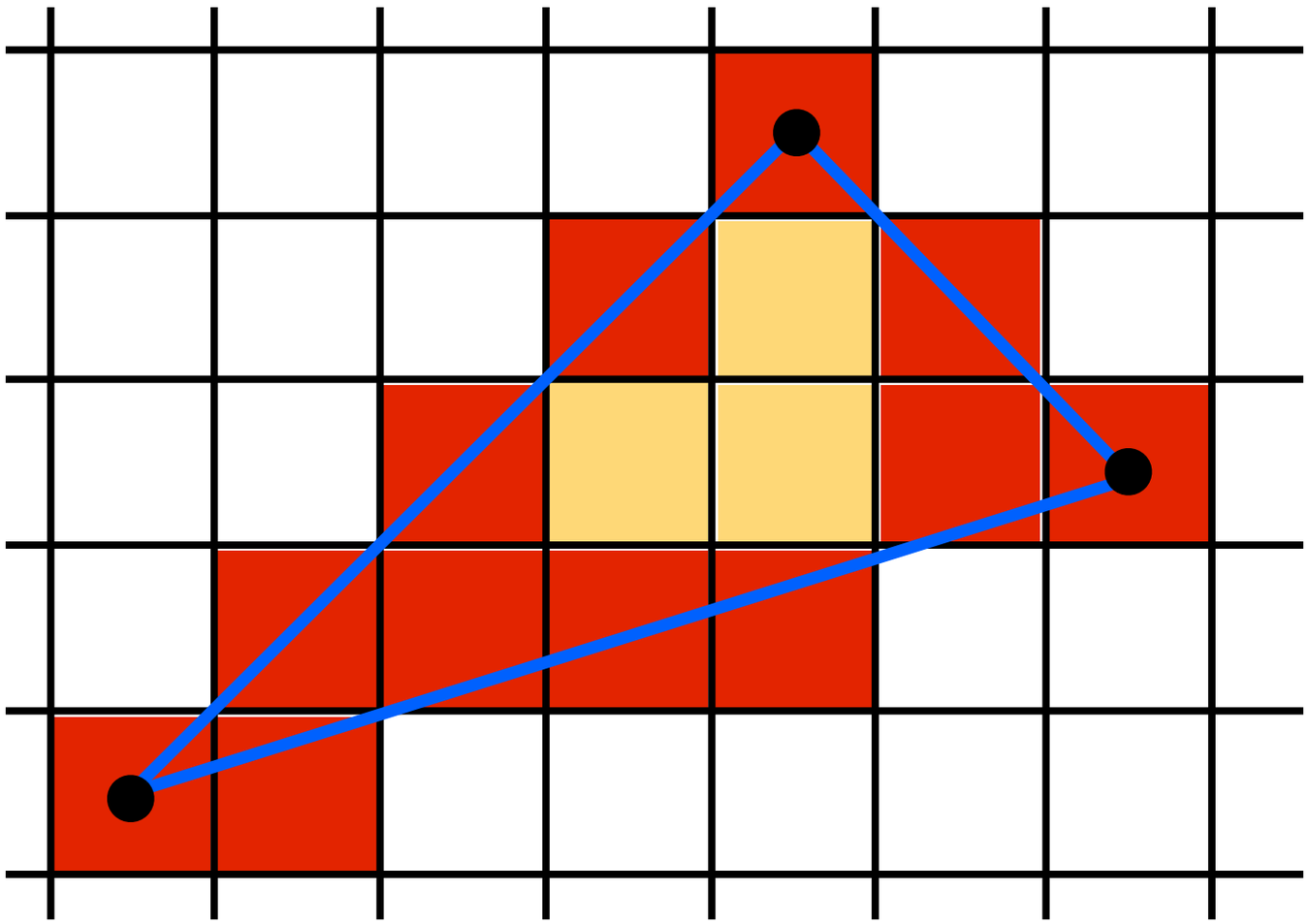
Who should fill in shared edge?

Which pixels should be used to approximate a triangle?



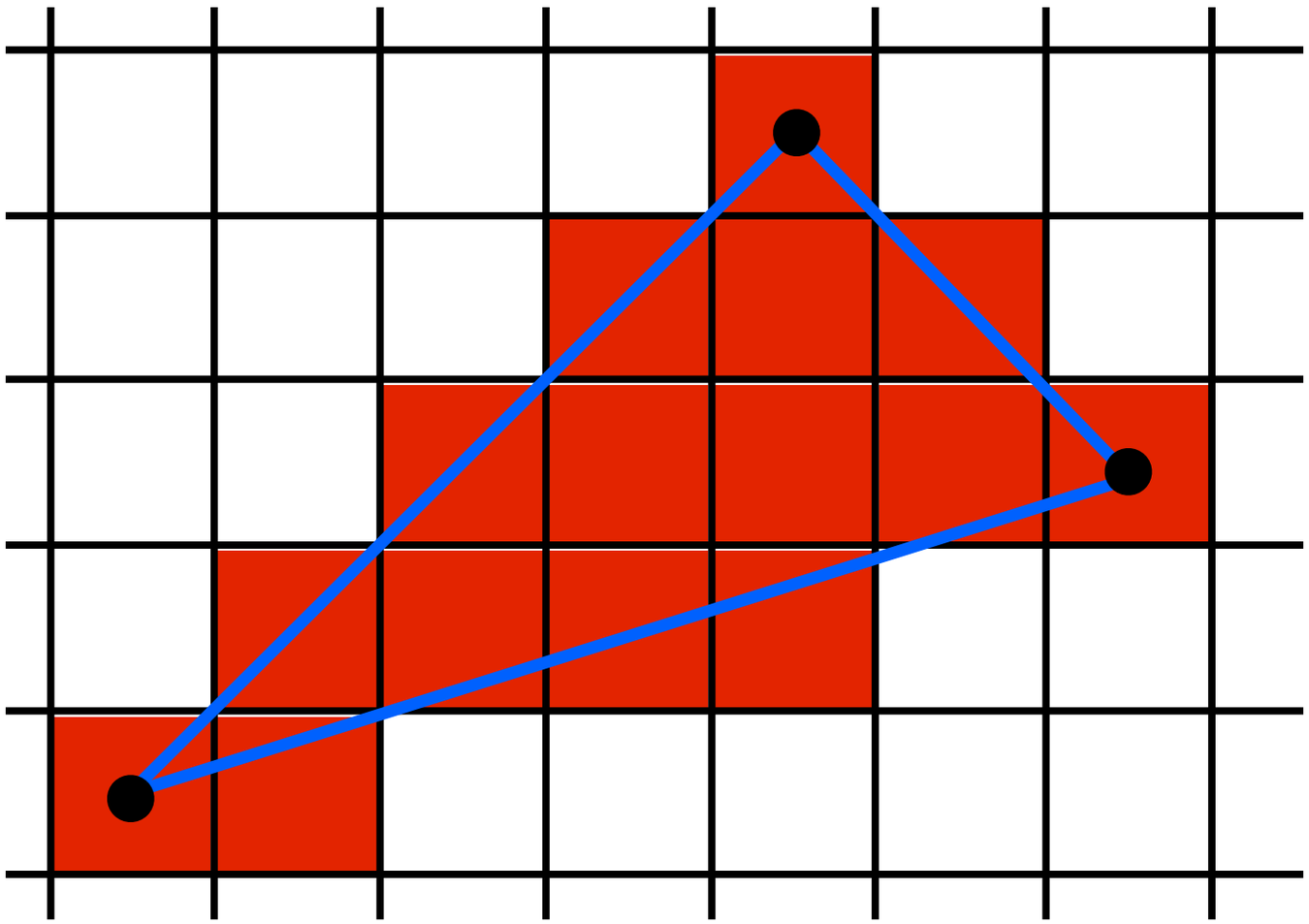
Who should fill in shared edge?

Which pixels should be used to approximate a triangle?



Use Midpoint Algorithm for edges and fill in?

Which pixels should be used to approximate a triangle?



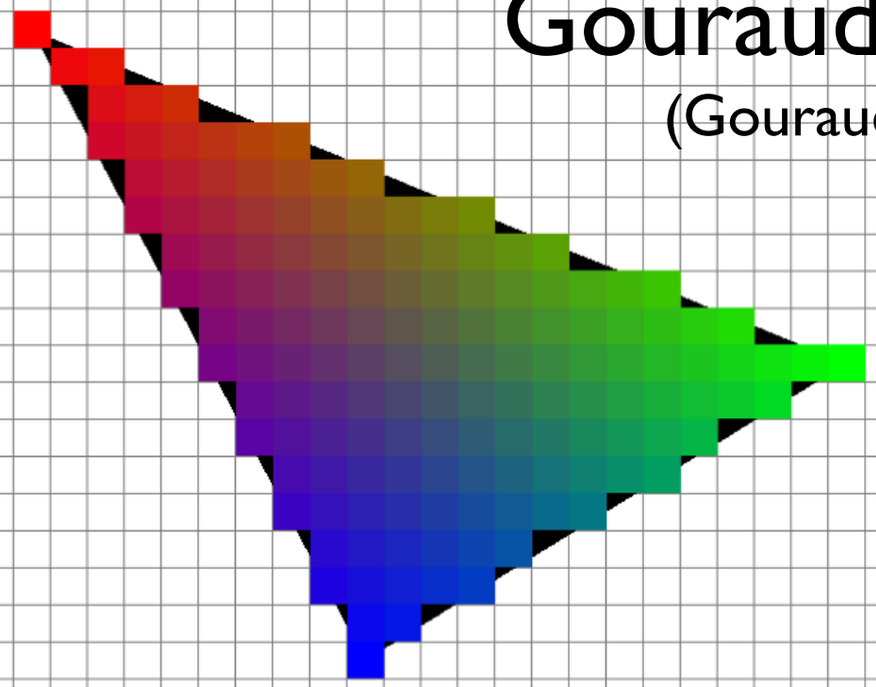
Use an approach based on barycentric coordinates

We can interpolate attributes using barycentric coordinates

$$\mathbf{c} = \alpha\mathbf{c}_0 + \beta\mathbf{c}_1 + \gamma\mathbf{c}_2$$

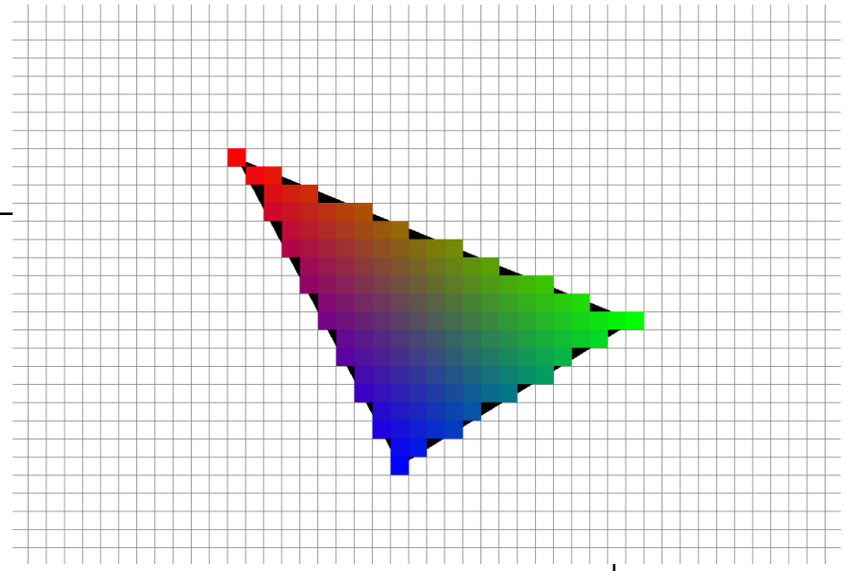
Gouraud shading

(Gouraud, 1971)



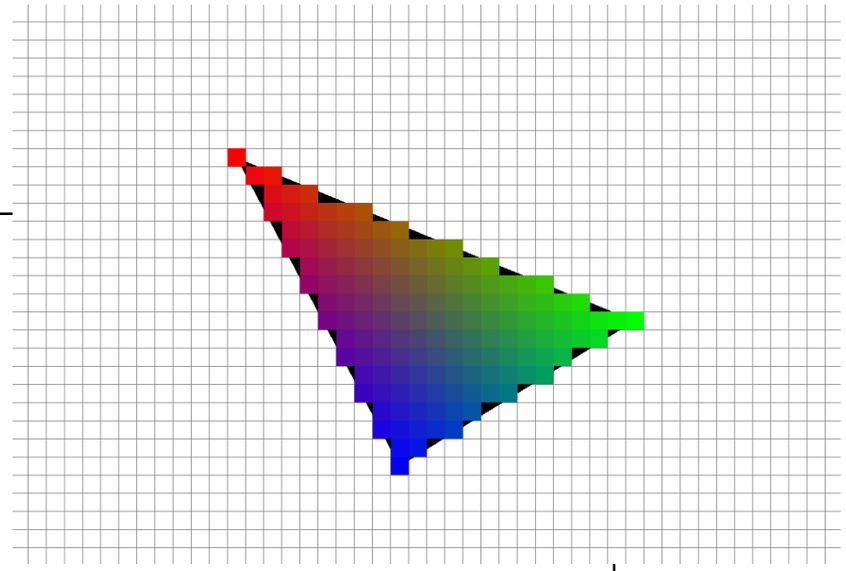
http://jtibble.dyndns.org/graphics/eecs487/eecs487.html

Triangle rasterization algorithm



```
for all x do
    for all y do
        compute (α, β, γ) for (x,y)
        if (α ∈ [0, 1] and β ∈ [0, 1] and γ ∈ [0, 1]) then
            c = αc0 + βc1 + γc2
            drawpixel(x,y) with color c
```

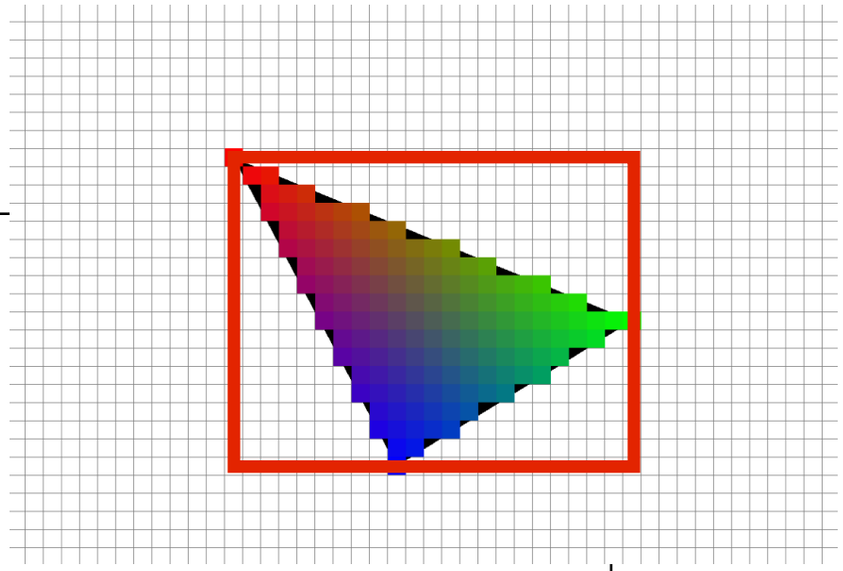
Triangle rasterization algorithm



for all x do
  for all y do
    compute (α, β, γ) for (x,y)
    if $(\alpha \in [0,1]$ and $\beta \in [0,1]$ and $\gamma \in [0,1])$ then
      $\mathbf{c} = \alpha\mathbf{c}_0 + \beta\mathbf{c}_1 + \gamma\mathbf{c}_2$
      drawpixel(x,y) with color c

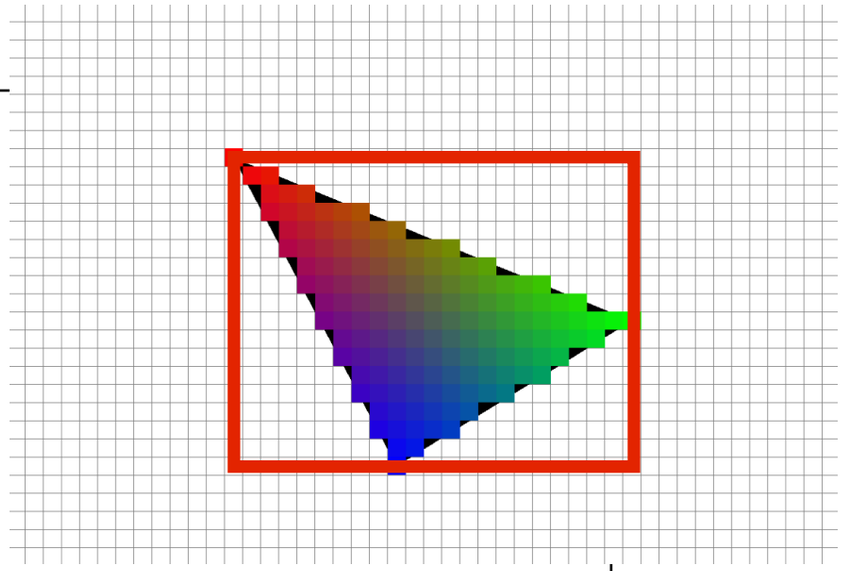
Triangle rasterization algorithm

use a bounding rectangle



for x in [x_min, x_max]
  for y in [y_min, y_max]
    compute (α, β, γ) for (x,y)
    if $(\alpha \in [0,1]$ and $\beta \in [0,1]$ and $\gamma \in [0,1])$ then
      $\mathbf{c} = \alpha\mathbf{c}_0 + \beta\mathbf{c}_1 + \gamma\mathbf{c}_2$
      drawpixel(x,y) with color c

Triangle rasterization algorithm



for x in [x_min, x_max]
  for y in [y_min, y_max]

$$\alpha = f_{bc}(x, y)/f_{bc}(x_a, y_a)$$

$$\beta = f_{ca}(x, y)/f_{ca}(x_b, y_b)$$

$$\gamma = f_{ab}(x, y)/f_{ab}(x_c, y_c)$$

if $(\alpha \in [0, 1]$ and $\beta \in [0, 1]$ and $\gamma \in [0, 1])$ then

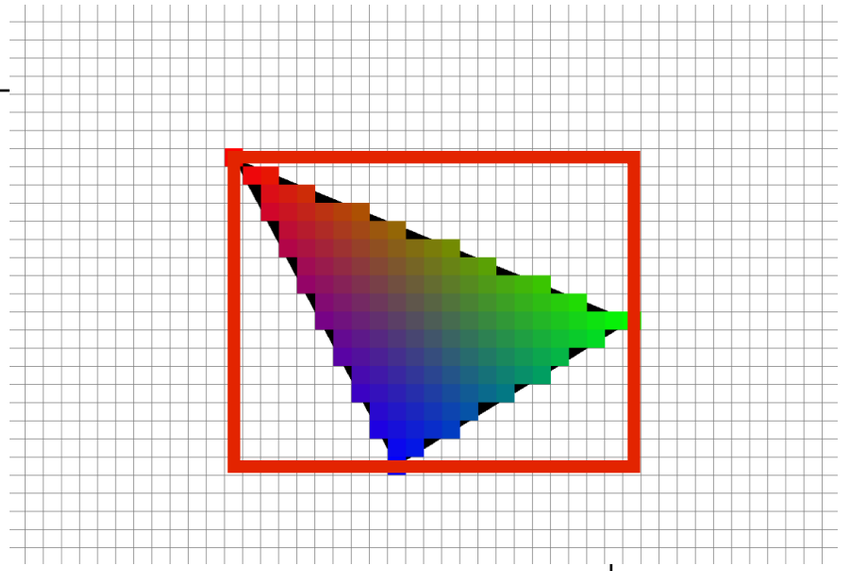
$$\mathbf{c} = \alpha\mathbf{c}_0 + \beta\mathbf{c}_1 + \gamma\mathbf{c}_2$$

drawpixel(x,y) with color c

<whiteboard>

Triangle rasterization algorithm

Optimizations?



for x in [x_min, x_max]
  for y in [y_min, y_max]

$$\alpha = f_{bc}(x, y) / f_{bc}(x_a, y_a)$$

$$\beta = f_{ca}(x, y) / f_{ca}(x_b, y_b)$$

$$\gamma = f_{ab}(x, y) / f_{ab}(x_c, y_c)$$

if $(\alpha \in [0, 1]$ and $\beta \in [0, 1]$ and $\gamma \in [0, 1])$ then

$$\mathbf{c} = \alpha \mathbf{c}_0 + \beta \mathbf{c}_1 + \gamma \mathbf{c}_2$$

drawpixel(x,y) with color c

Triangle rasterization algorithm

Optimizations?

for x in [x_min, x_max]
  for y in [y_min, y_max]

$$\alpha = f_{bc}(x,y)/f_{bc}(x_a,y_a)$$

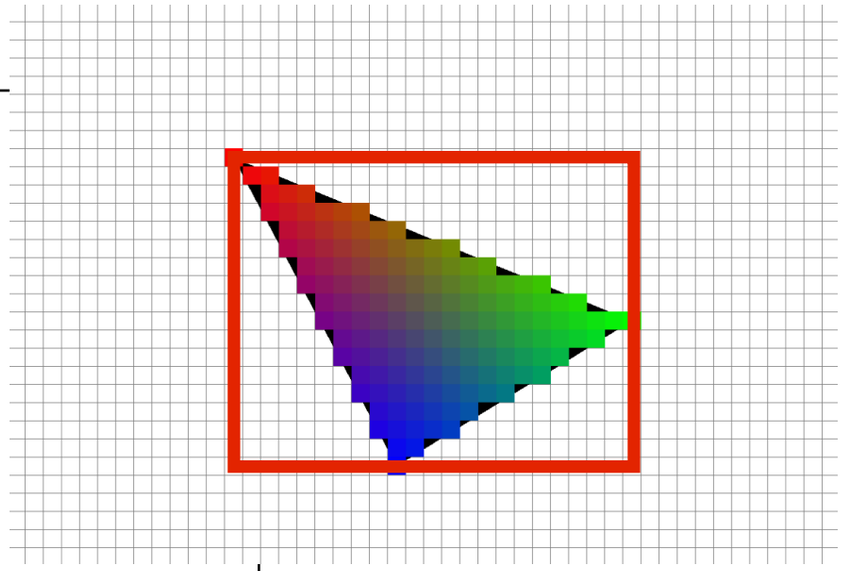
$$\beta = f_{ca}(x,y)/f_{ca}(x_b,y_b)$$

$$\gamma = f_{ab}(x,y)/f_{ab}(x_c,y_c)$$

if $(\alpha \geq 0 \text{ and } \beta \geq 0 \text{ and } \gamma \geq 0)$ then

$$\mathbf{c} = \alpha\mathbf{c}_0 + \beta\mathbf{c}_1 + \gamma\mathbf{c}_2$$

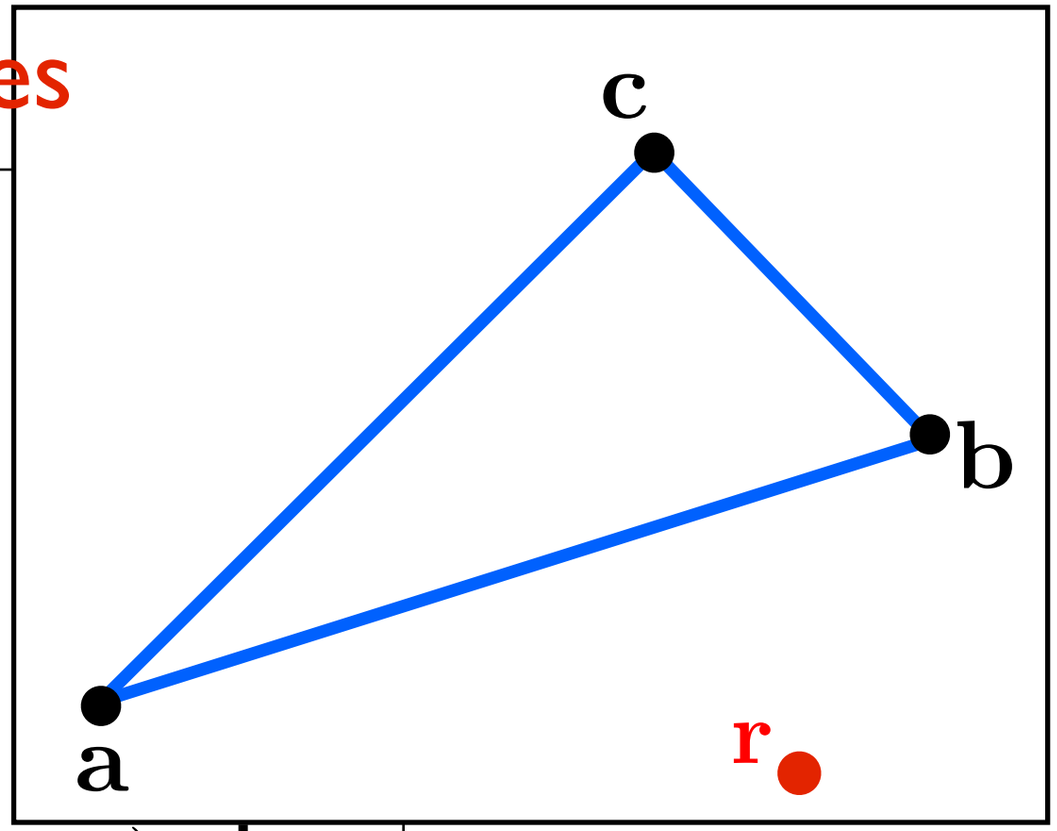
drawpixel(x,y) with color c



make computation of bary. coords. incremental
color can also be computed incrementally
don't need to check upper bound

Triangle rasterization algorithm

dealing with shared triangle edges



for x in [x_min, x_max]

for y in [y_min, y_max]

$$\alpha = f_{bc}(x,y)/f_{bc}(x_a,y_a)$$

$$\beta = f_{ac}(x,y)/f_{ac}(x_b,y_b)$$

$$\gamma = f_{ab}(x,y)/f_{ab}(x_c,y_c)$$

if $(\alpha \geq 0$ and $\beta \geq 0$ and $\gamma \geq 0)$ then

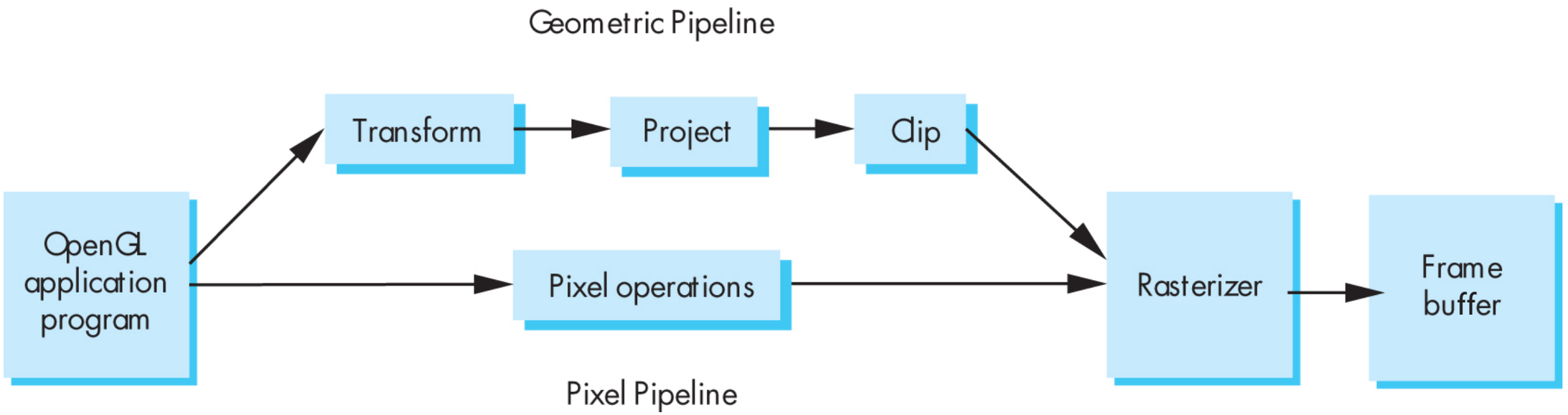
if $(\alpha > 0$ or $f_{bc}(\mathbf{a})f_{bc}(\mathbf{r}) > 0)$ and
$(\beta > 0$ or $f_{ca}(\mathbf{b})f_{ca}(\mathbf{r}) > 0)$ and
$(\gamma > 0$ or $f_{ab}(\mathbf{c})f_{ab}(\mathbf{r}) > 0)$ then

$$\mathbf{c} = \alpha\mathbf{c}_0 + \beta\mathbf{c}_1 + \gamma\mathbf{c}_2$$

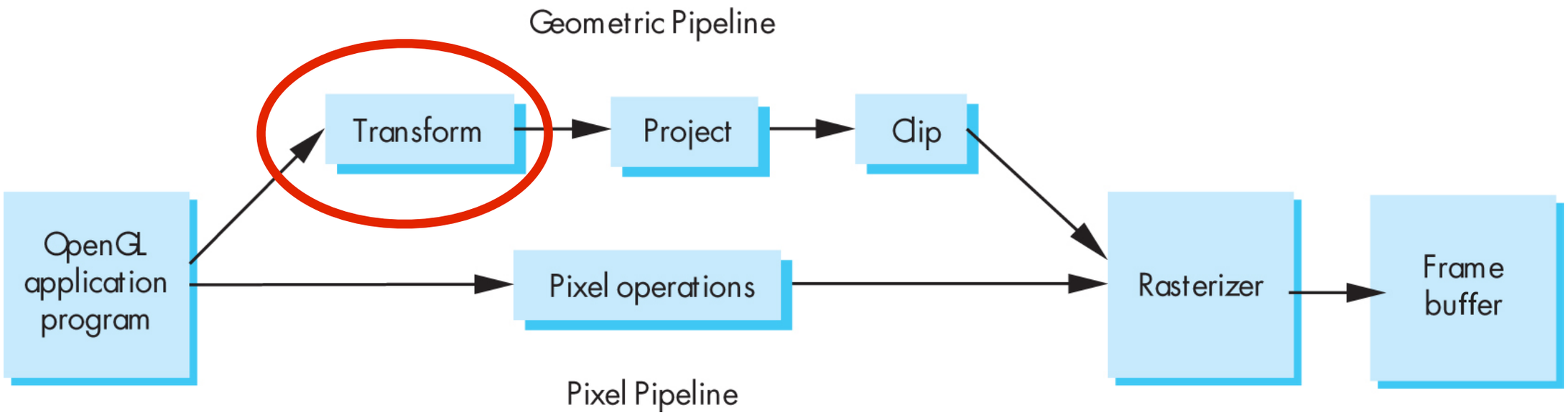
drawpixel(x,y) with color c

Graphics Pipeline (cont.)

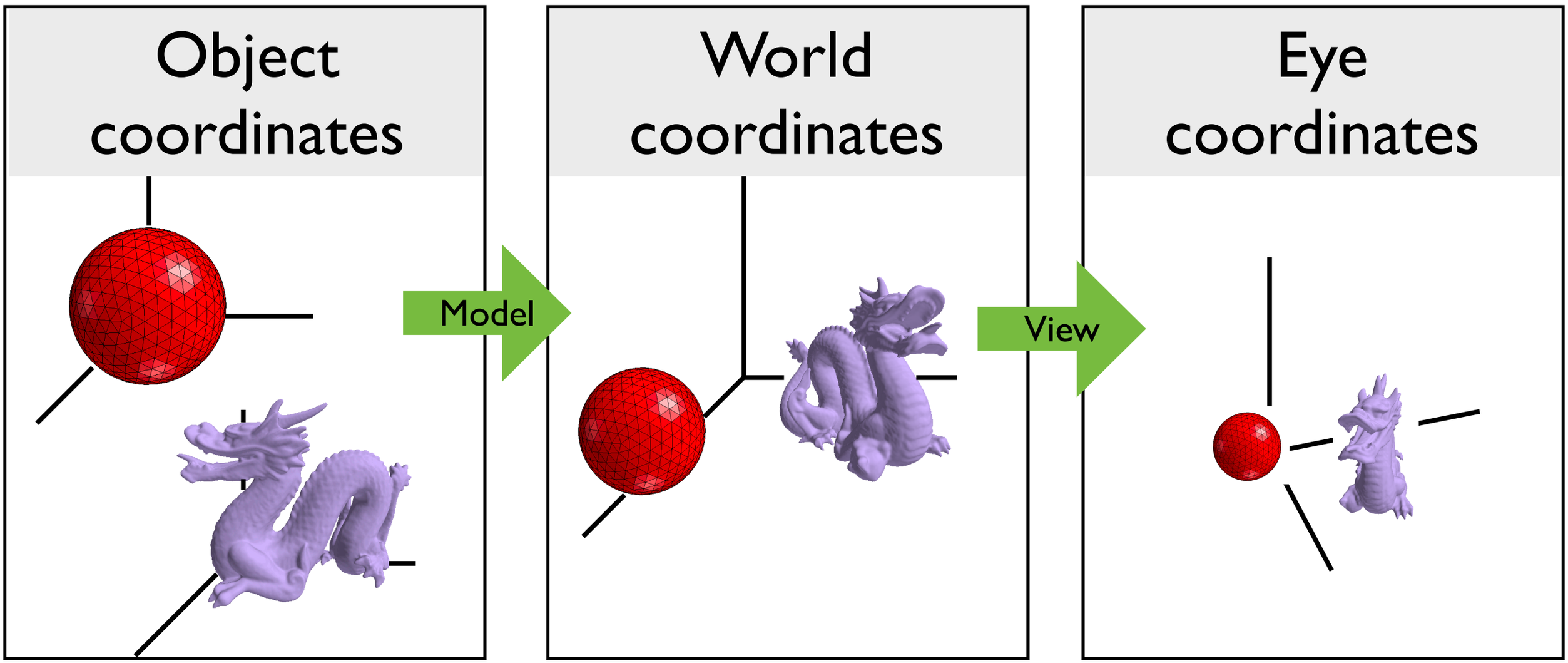
Graphics Pipeline



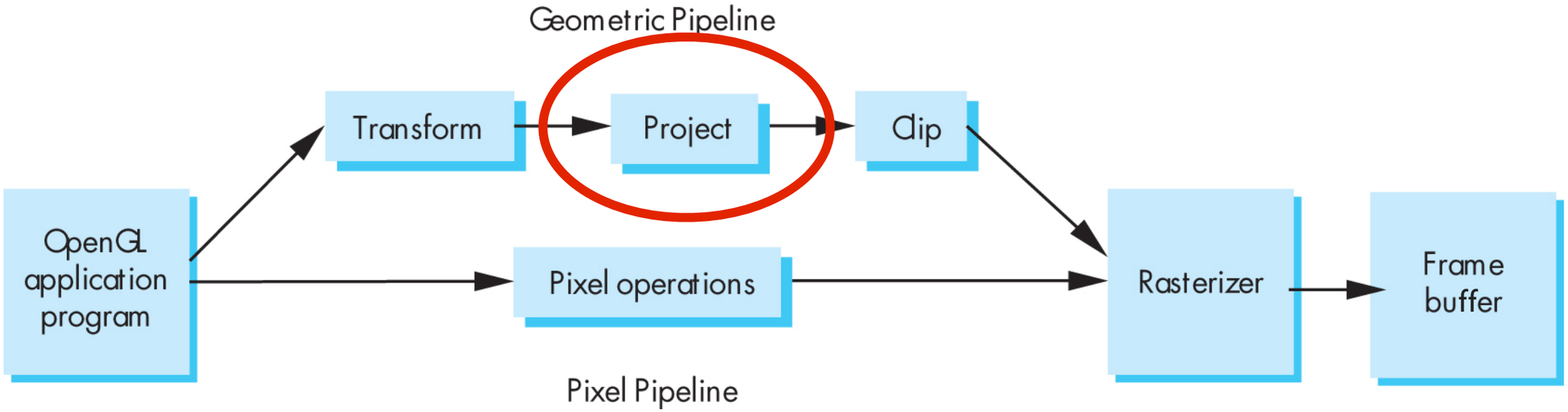
Transform


Geometric Pipeline
Transform
Project
Clip
OpenGL application program
Pixel operations
Rasterizer
Frame buffer
Pixel Pipeline

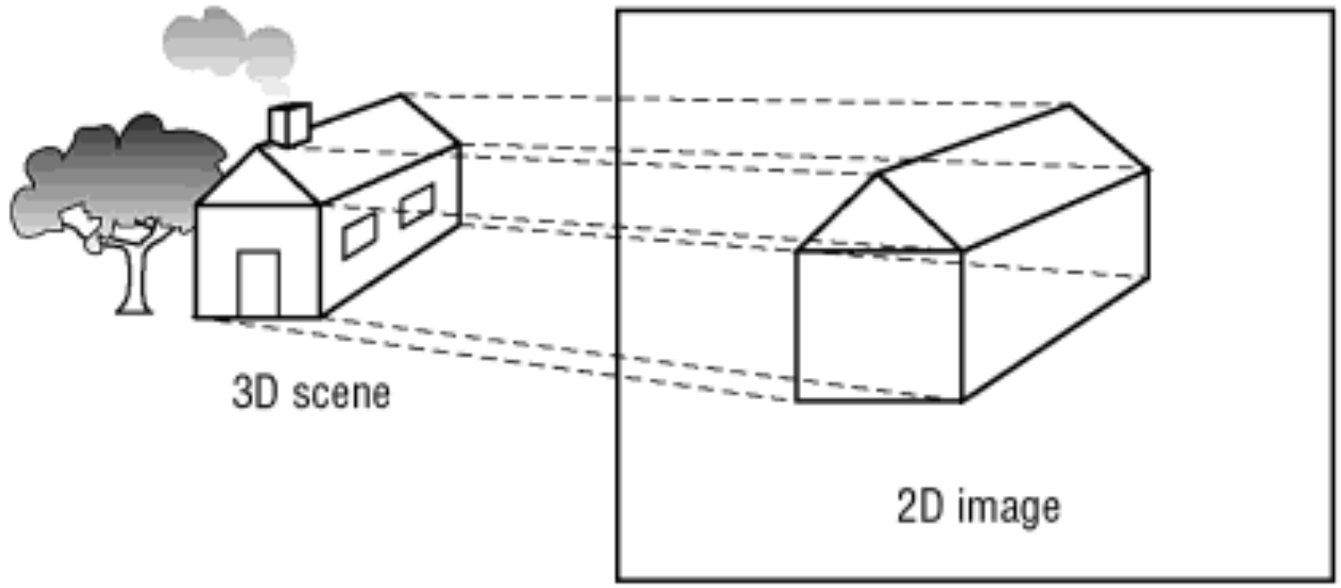
"Modelview" Transformation



Project

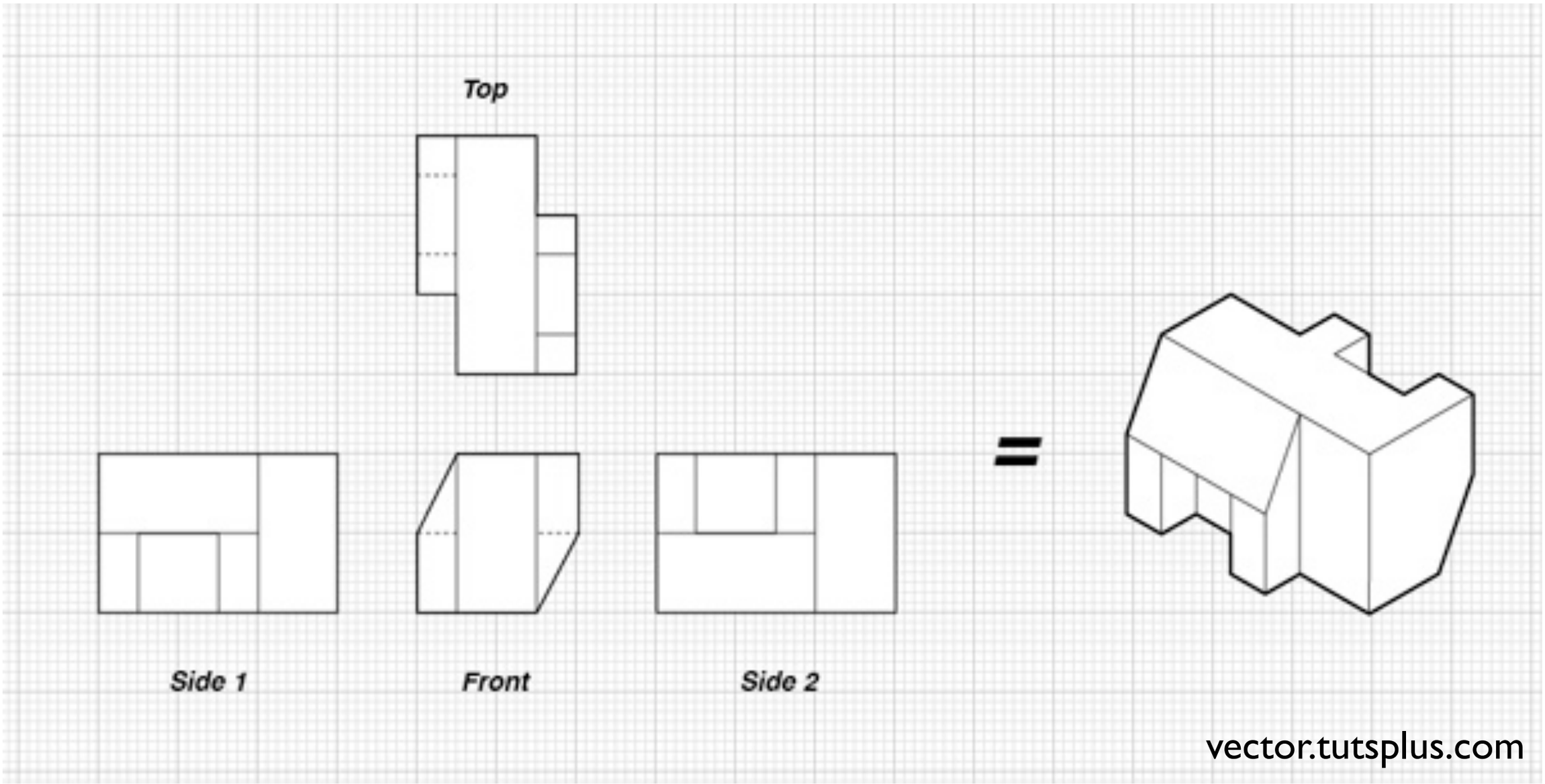

Geometric Pipeline
Transform
Project
Clip
OpenGL application program
Pixel operations
Rasterizer
Frame buffer
Pixel Pipeline

Projection: map 3D scene to 2D image



OpenGL Super Bible, 5th Ed.

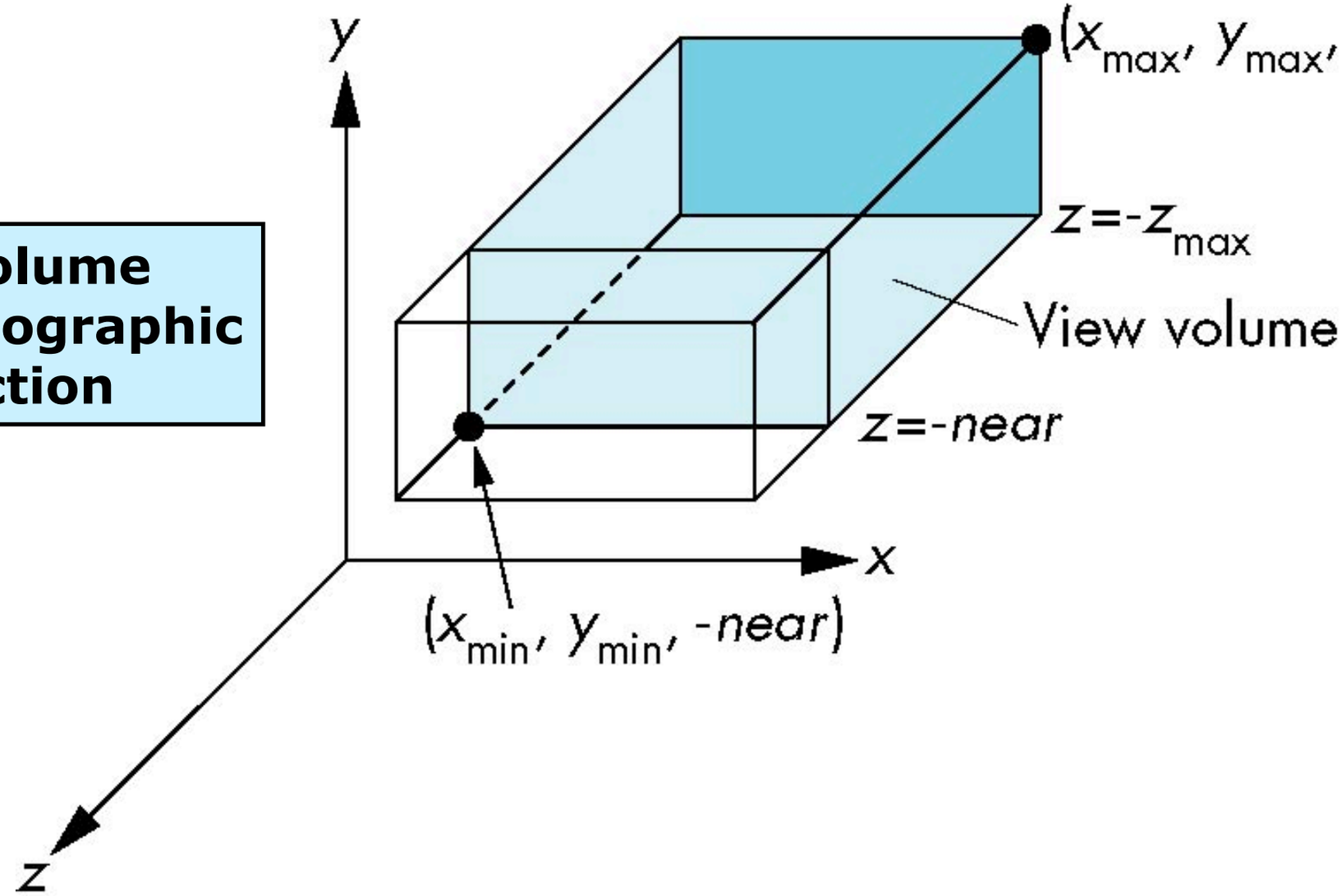
Orthographic projection



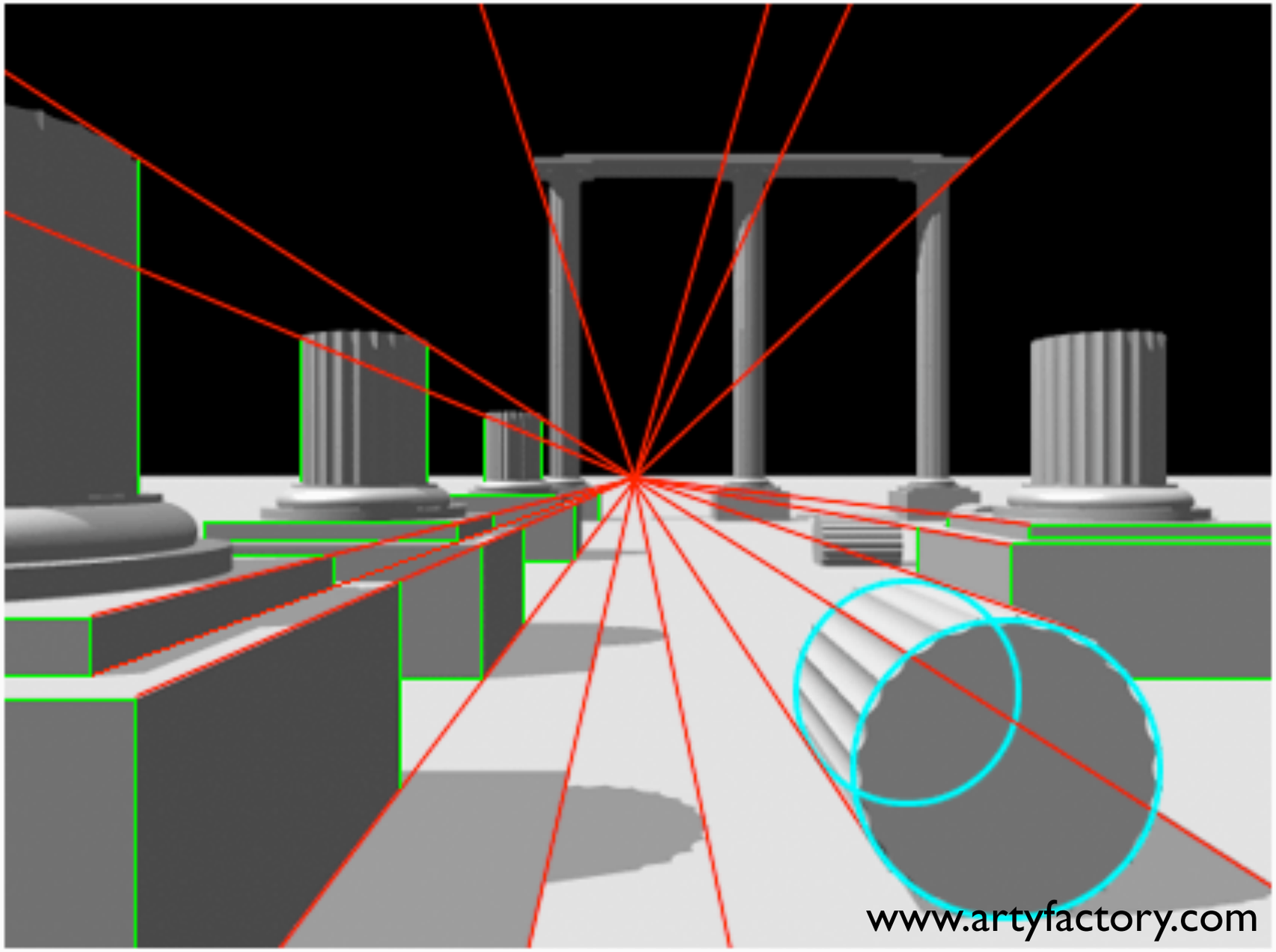
OpenGL Orthogonal Viewing

```
glOrtho(left,right,bottom,top,near,far)
```

View volume for an orthographic projection



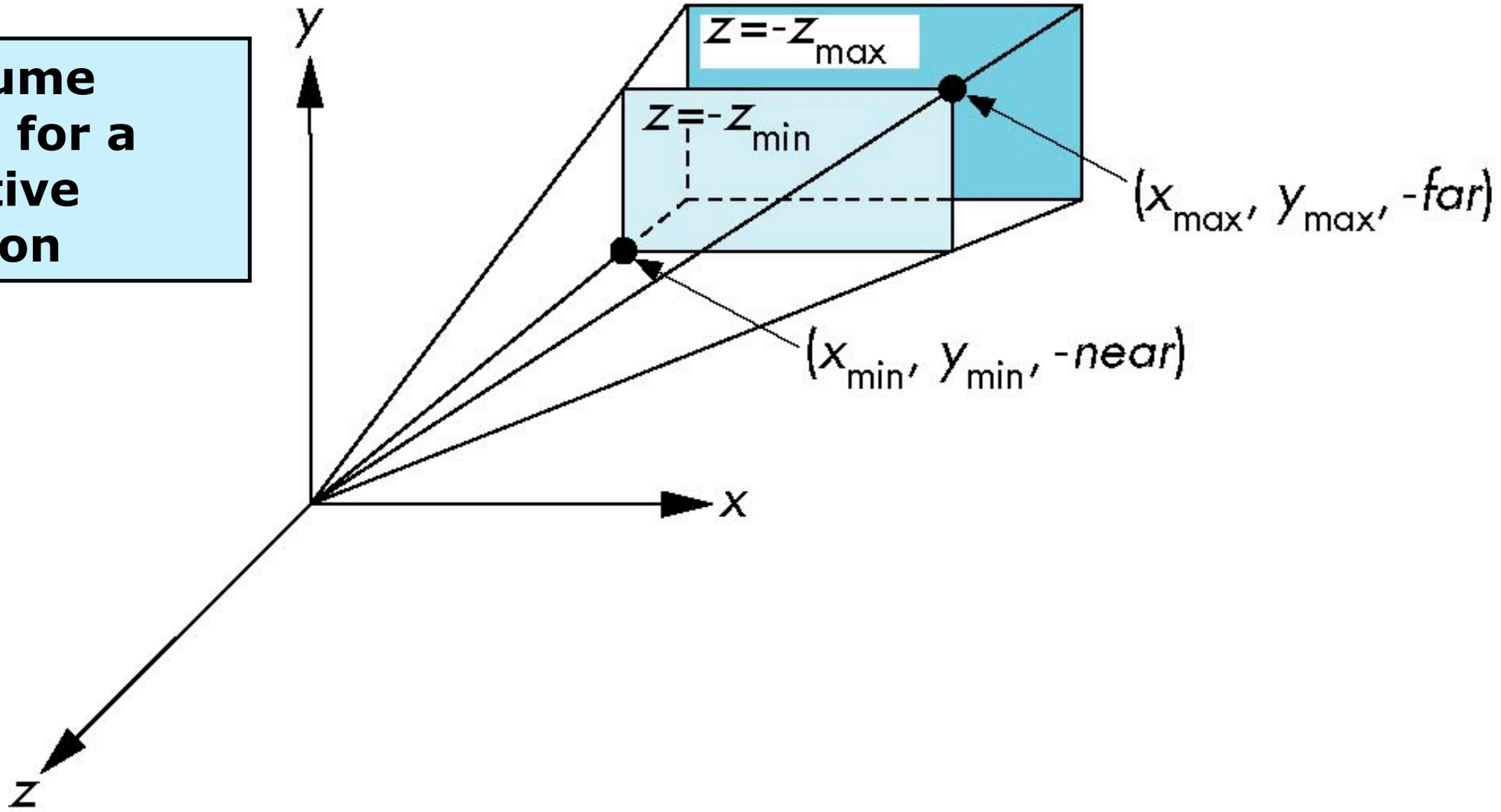
Perspective projection



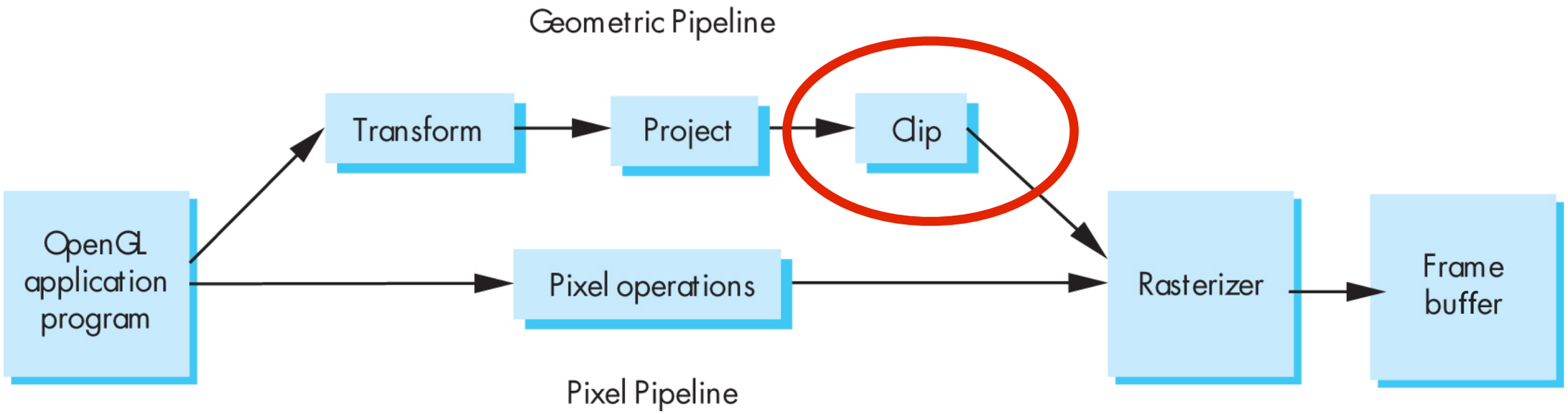
OpenGL Perspective Viewing

`glFrustum(xmin,xmax,ymin,ymax,near,far)`

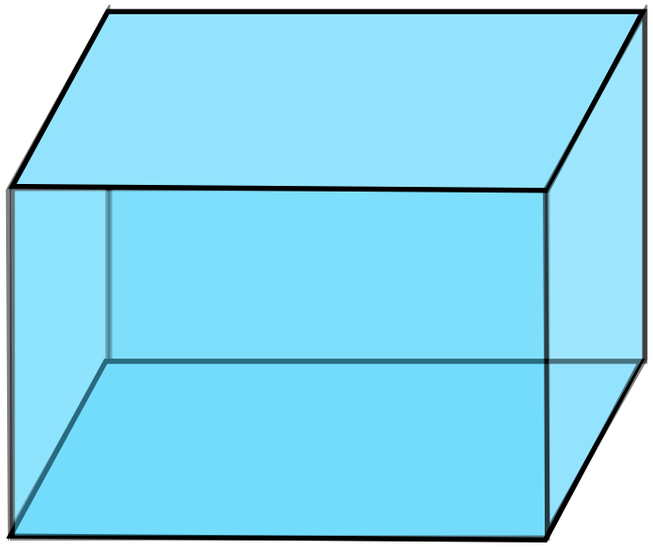
View volume (frustrum) for a perspective projection

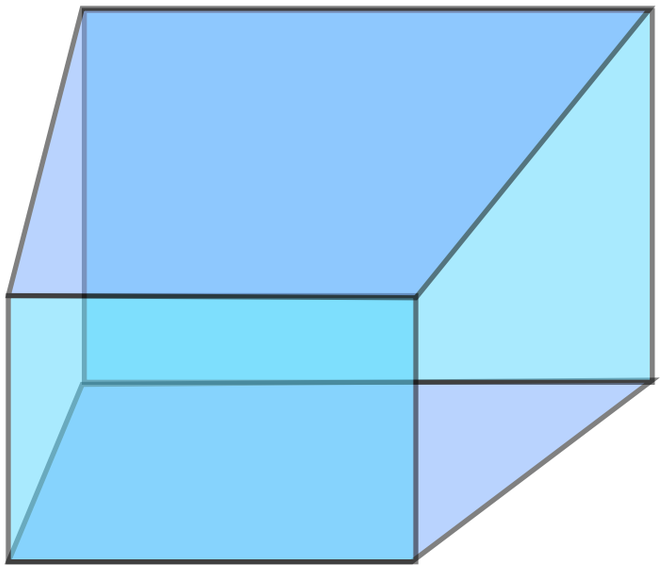


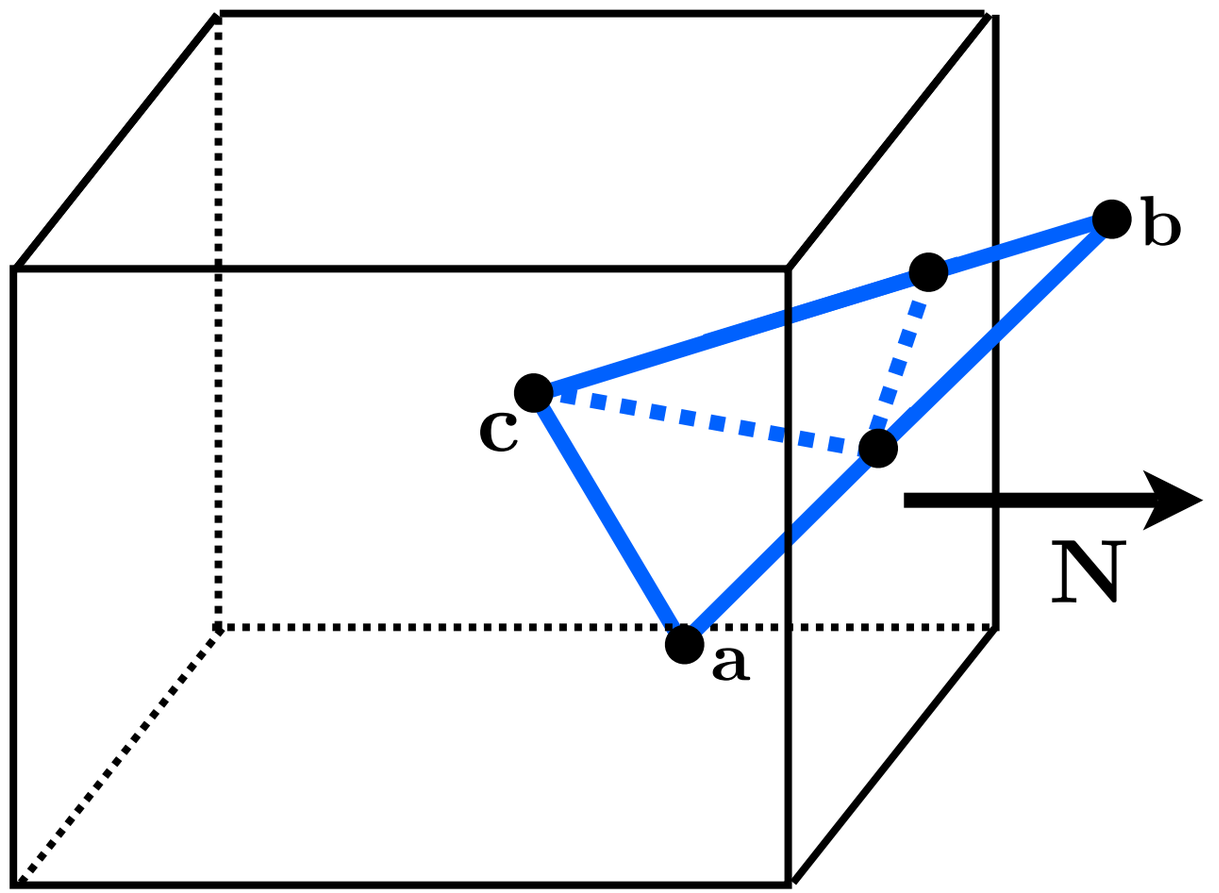
Clip


Geometric Pipeline
Transform
Project
Clip
OpenGL application program
Pixel operations
Rasterizer
Frame buffer
Pixel Pipeline

Clip against view volume

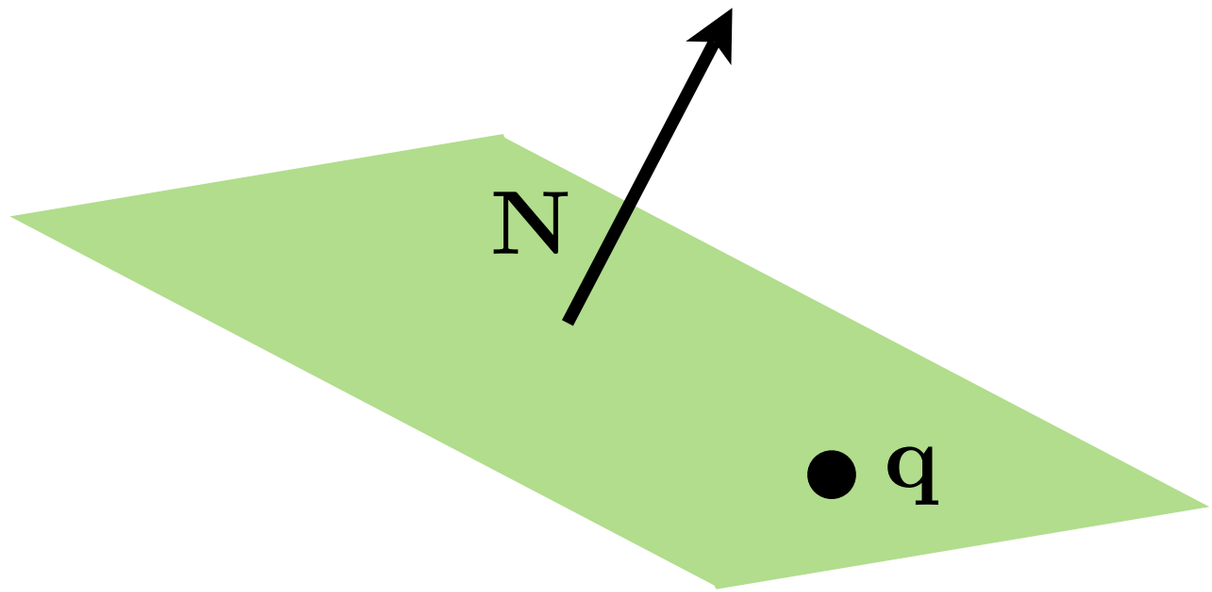


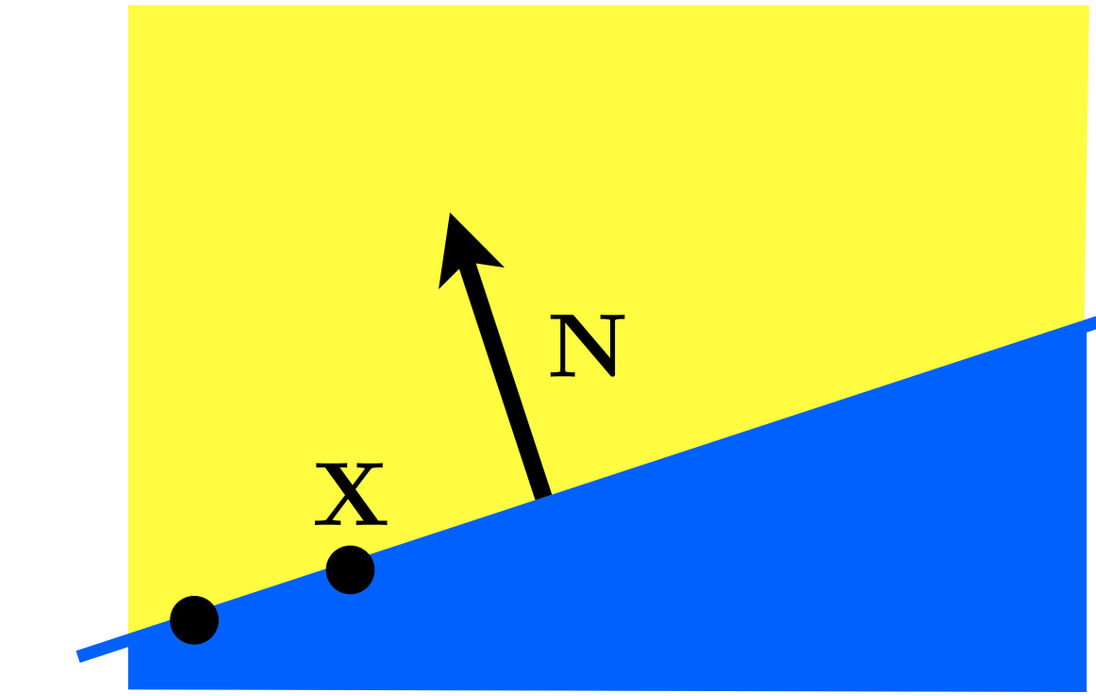




Clipping against a plane

What's the equation for the plane through **q** with normal **N**?





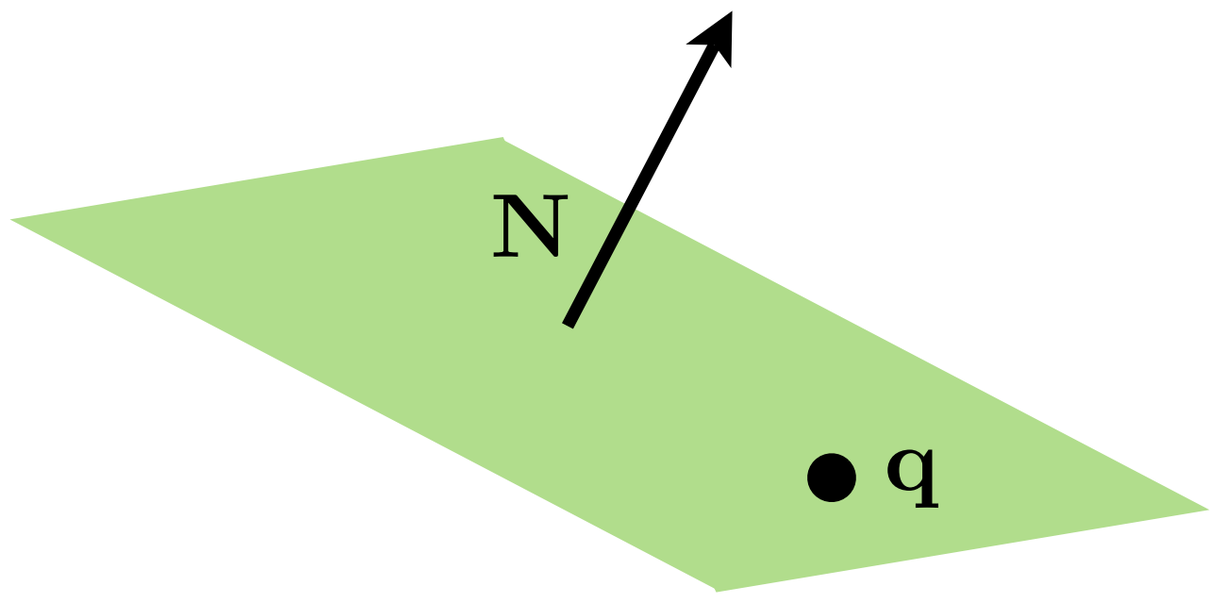
$\mathbf{X}_0 = (x_0, y_0)$

implicit line equation:

$$f(\mathbf{X}) = \mathbf{N} \cdot (\mathbf{X} - \mathbf{X}_0) = 0$$

Clipping against a plane

What's the equation for the plane through **q** with normal **N**?

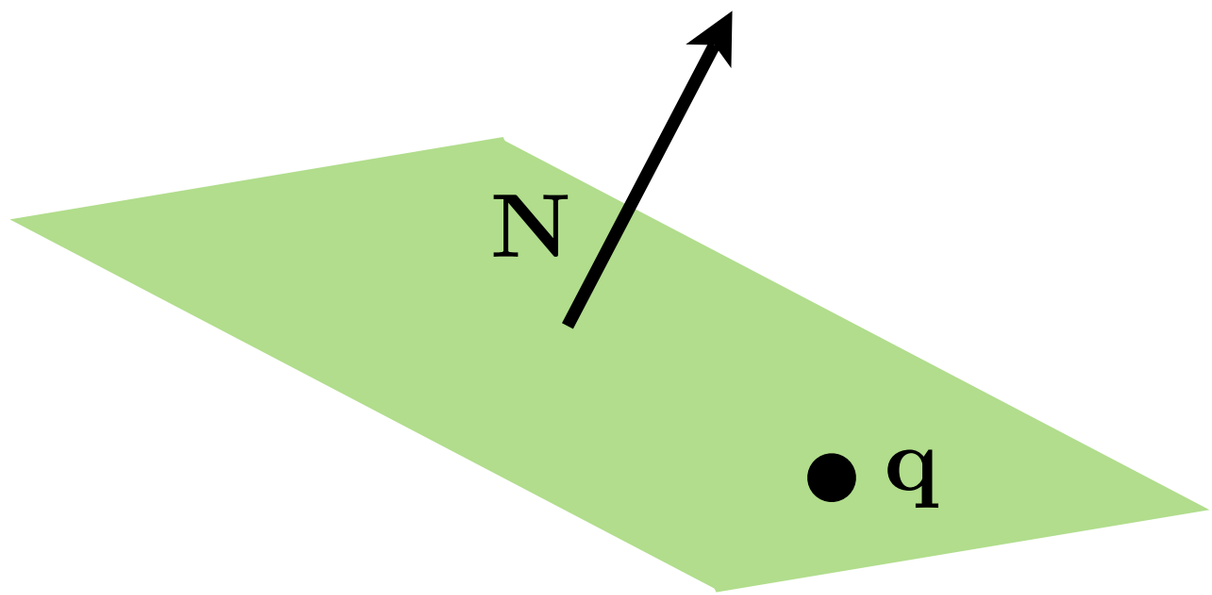


$$f(\mathbf{p}) =? = 0$$

<whiteboard>

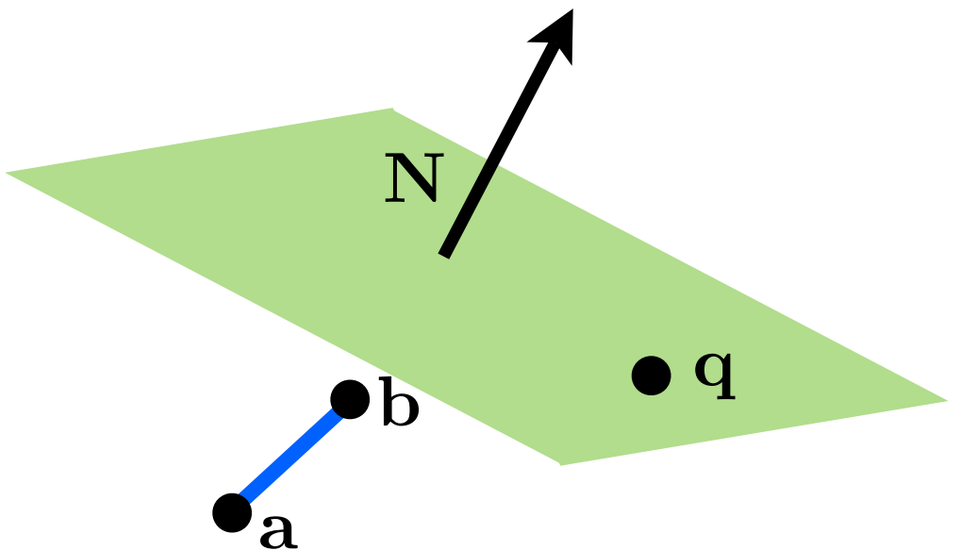
Clipping against a plane

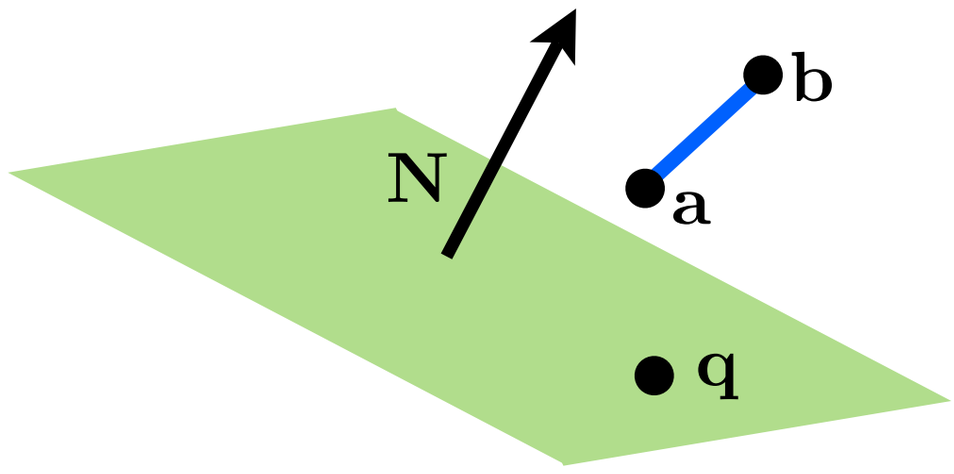
What's the equation for the plane through **q** with normal **N**?



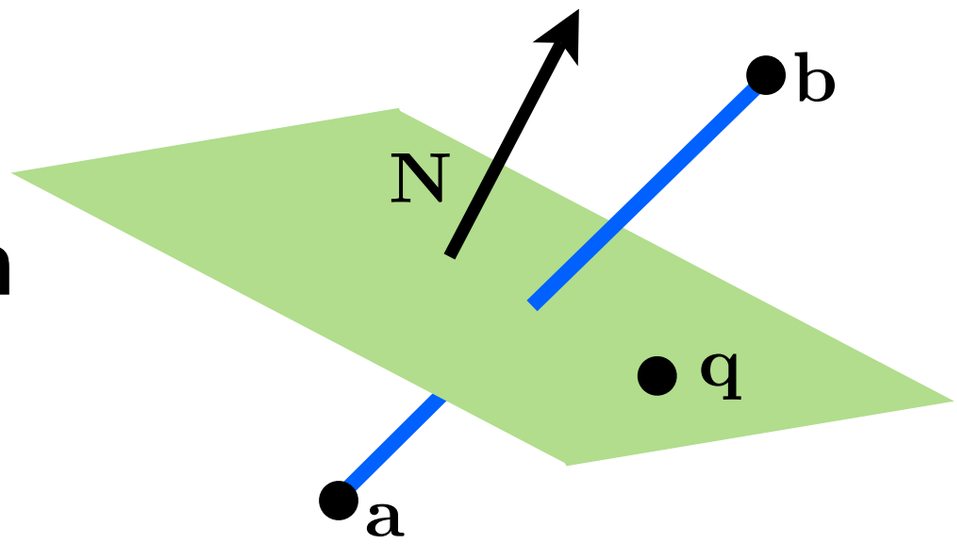
$$f(\mathbf{p}) = \mathbf{N} \cdot (\mathbf{p} - \mathbf{q}) = 0$$

Intersection of line and plane



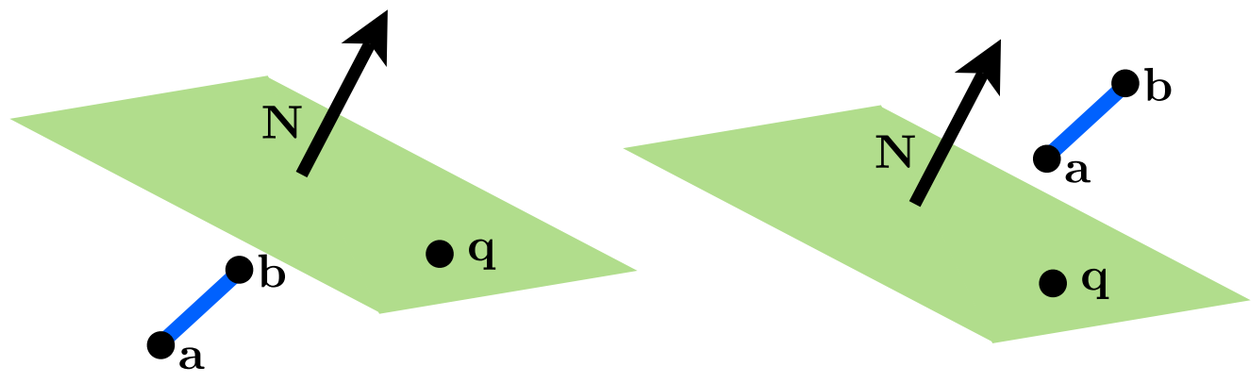


How can we distinguish between these cases?

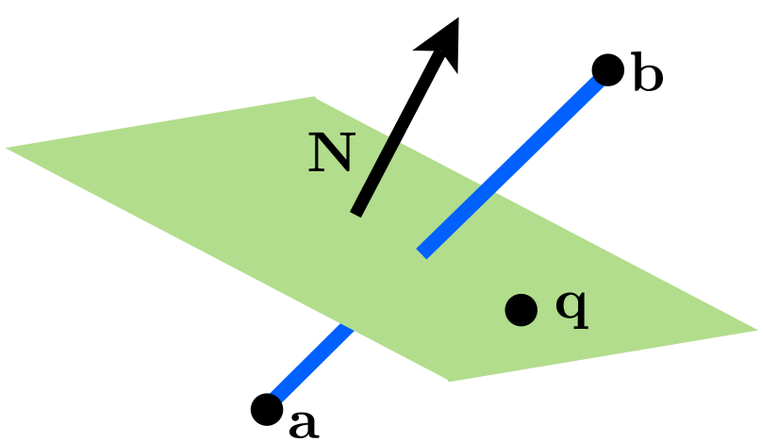


Intersection of line and plane

$$f(\mathbf{a})f(\mathbf{b}) \geq 0$$

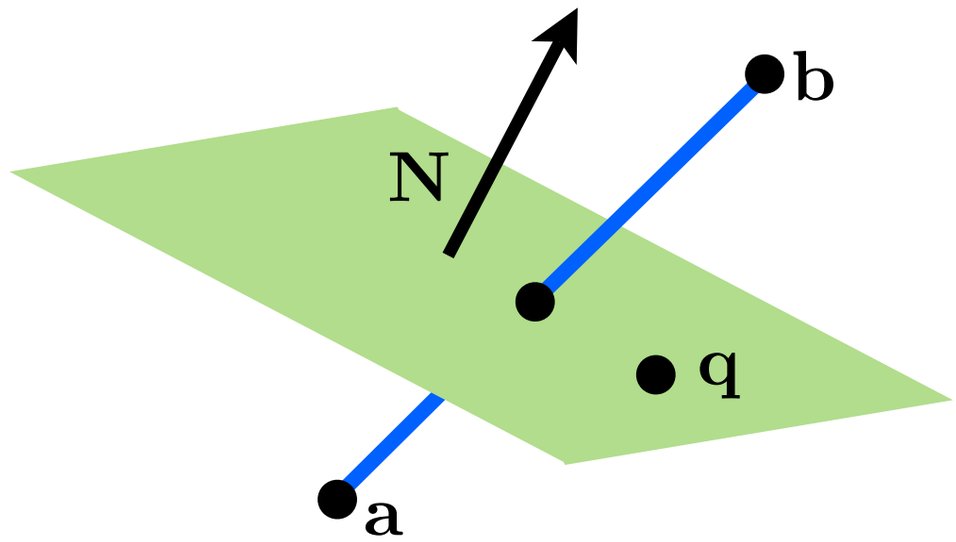


$$f(\mathbf{a})f(\mathbf{b}) < 0$$



Intersection of line and plane

How can we find the intersection point?



<whiteboard>

Clip against view volume

$$s = \frac{\mathbf{N} \cdot (\mathbf{q} - \mathbf{c})}{\mathbf{N} \cdot (\mathbf{b} - \mathbf{c})}$$

$$t = \frac{\mathbf{N} \cdot (\mathbf{q} - \mathbf{a})}{\mathbf{N} \cdot (\mathbf{b} - \mathbf{a})}$$

need to generate new triangles