CPSC 314
15 –RASTERIZATION (CONT.)

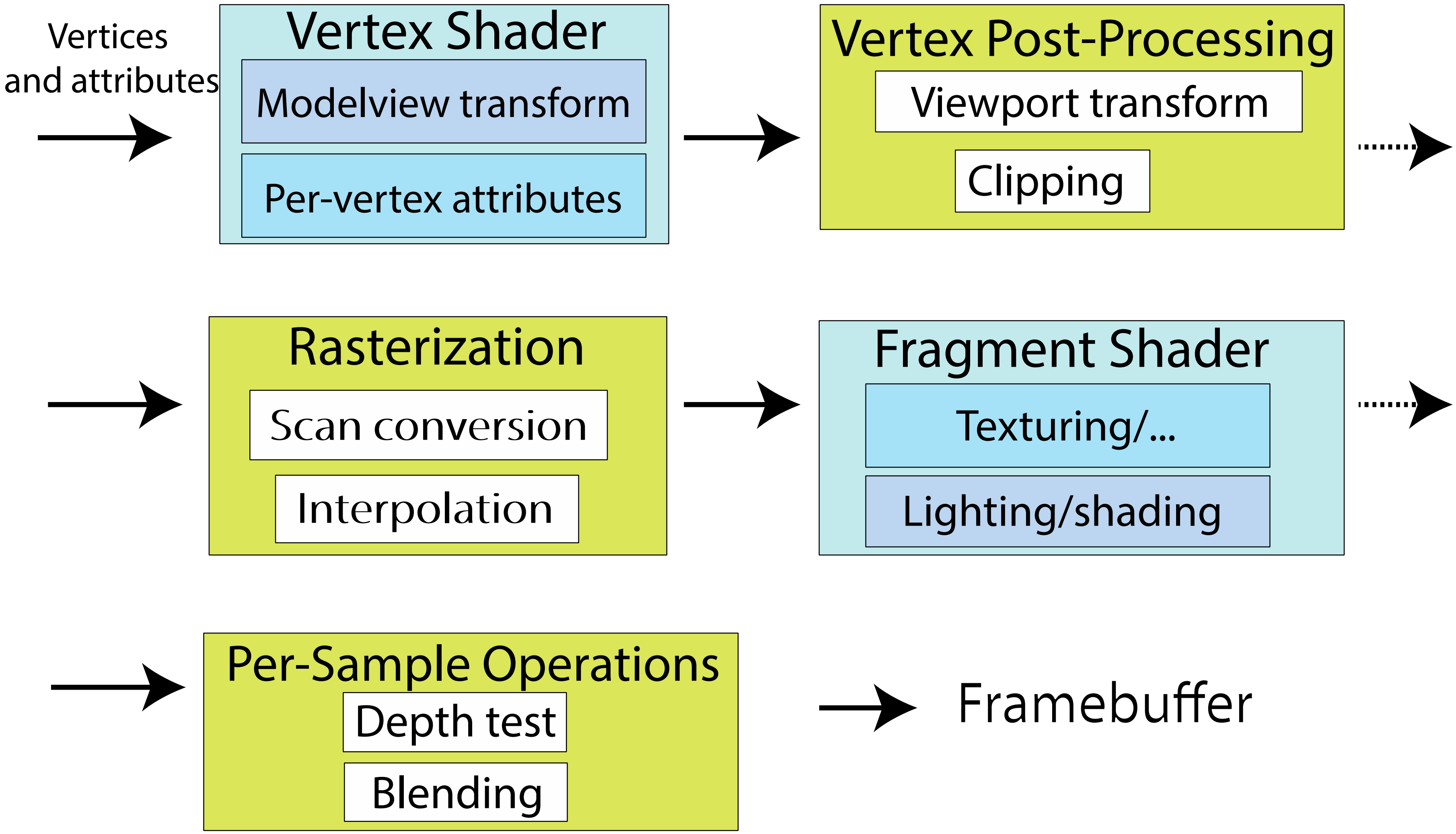
Textbook: 12.4

UGRAD.CS.UBC.CA/~CS314

Alla Sheffer

2016

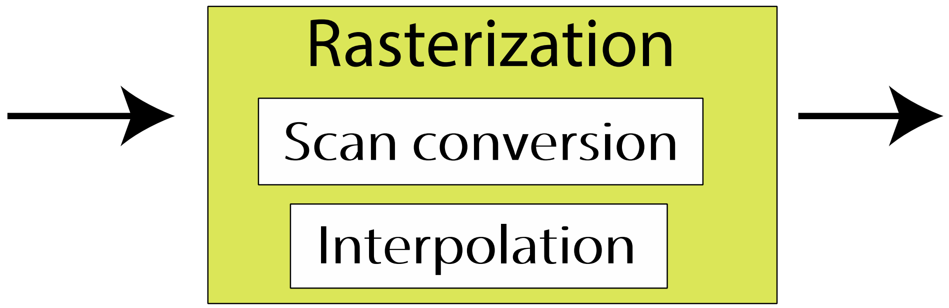
THE RENDERING PIPELINE



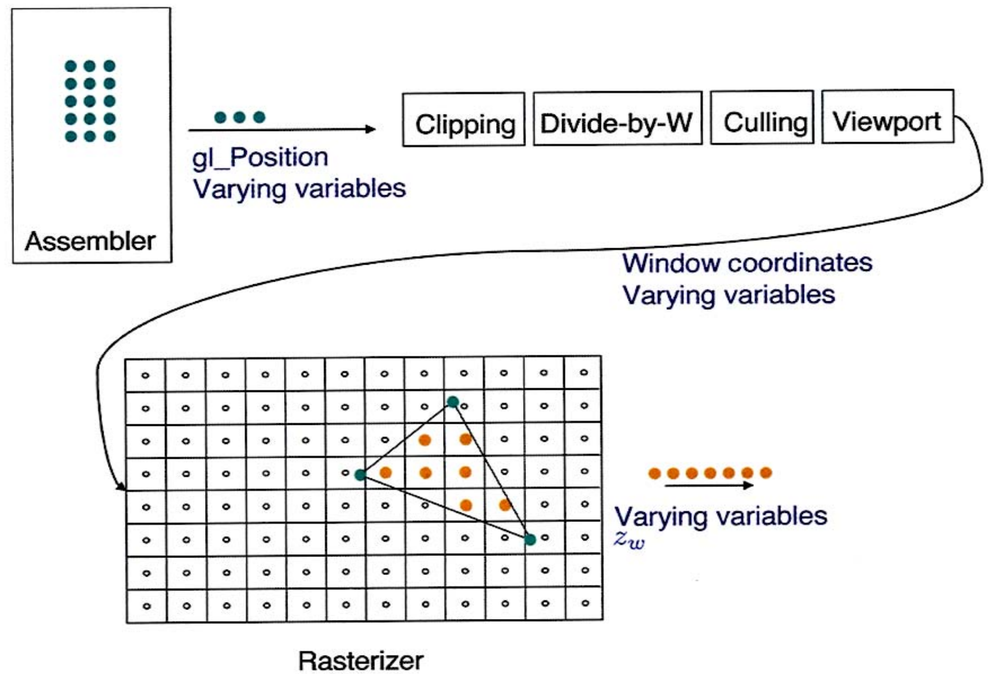
RASTERIZATION

- This is part of the fixed function pipeline

- Input: all polygons are clipped
- Output: fragments (with **varying variables** interpolated)



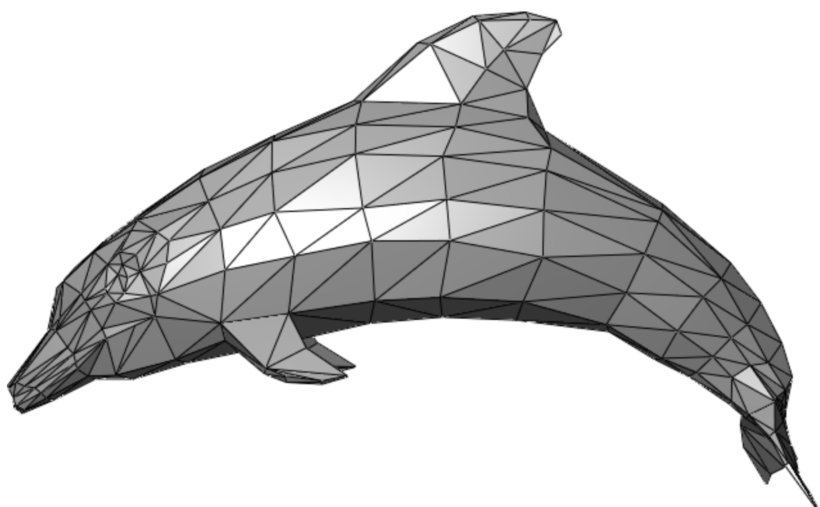
PATH FROM VERTEX TO PIXEL



GEOMETRY: POLYGONS (TRIANGLES++)

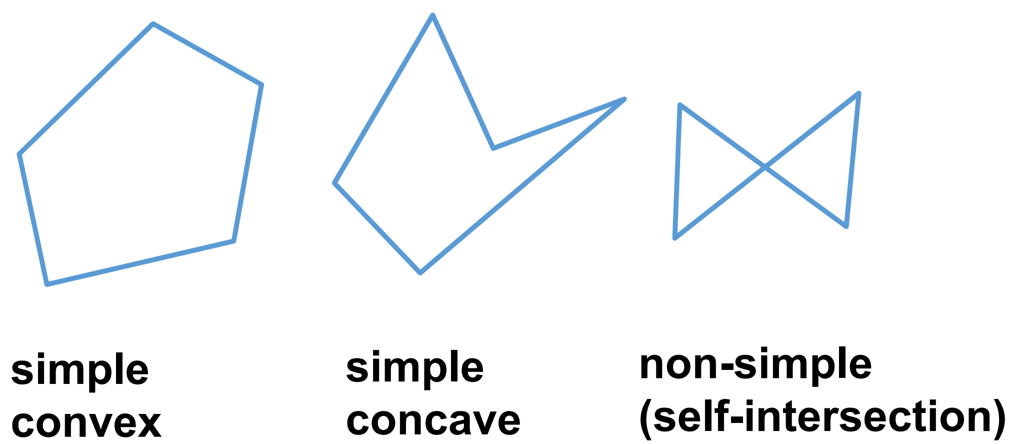
Interactive graphics uses Polygons

- Can represent any surface *with arbitrary accuracy*
  - *Splines, mathematical functions, …*
- simple, regular rendering algorithms
  - *embed well in hardware*



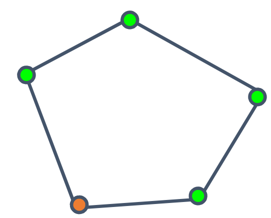
POLYGONS

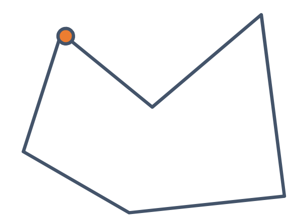
- Basic Types



FROM POLYGONS TO TRIANGLES

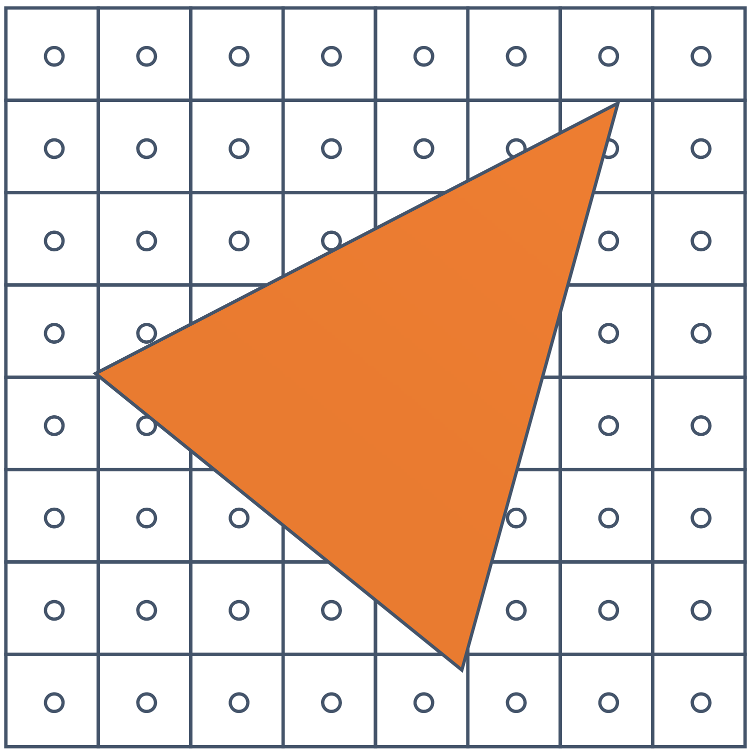
- why? triangles are always planar, always convex
- simple convex polygons
  - trivial to break into triangles
- concave or non-simple polygons
  - more effort to break into triangles



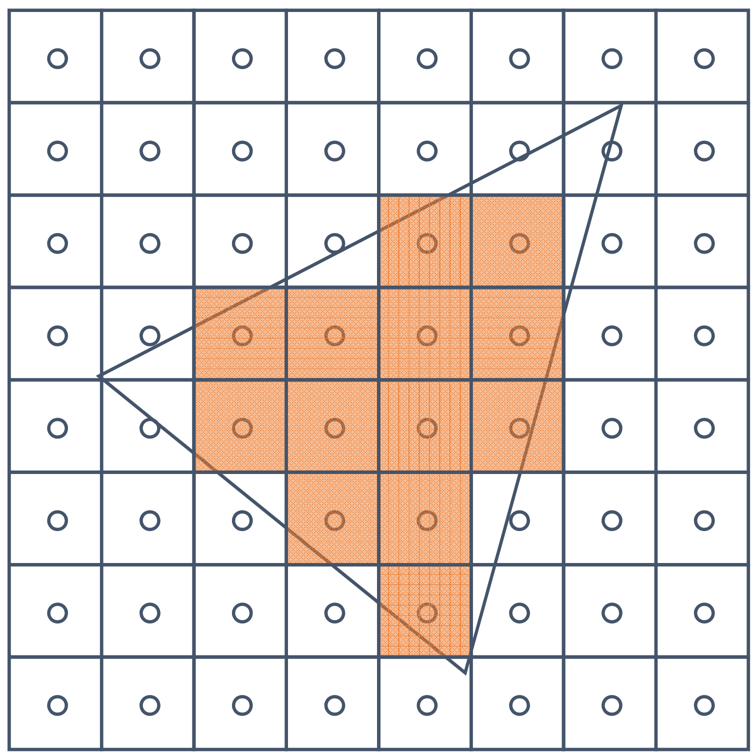


WHAT IS SCAN CONVERSION? (A.K.A. RASTERIZATION)

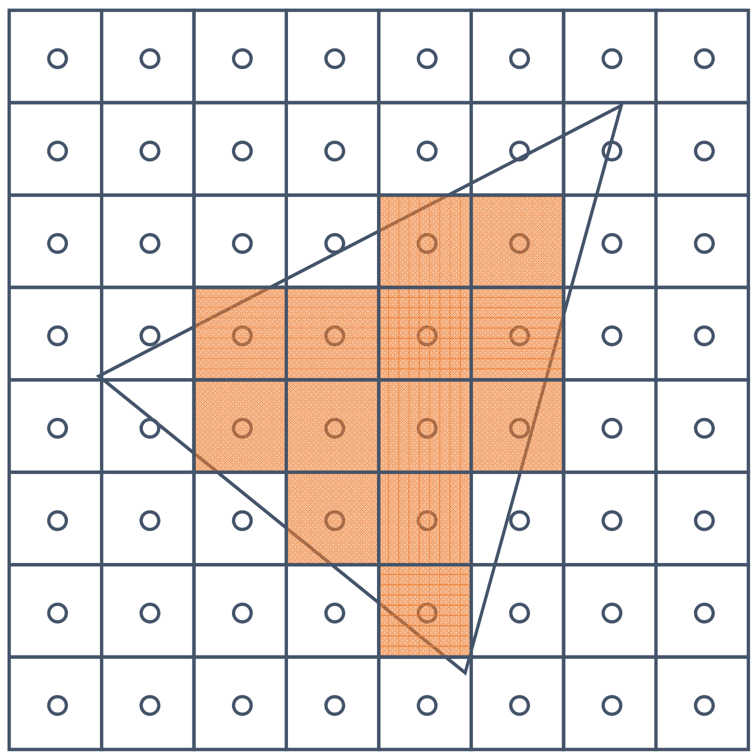
- screen is discrete



•one possible scan conversion

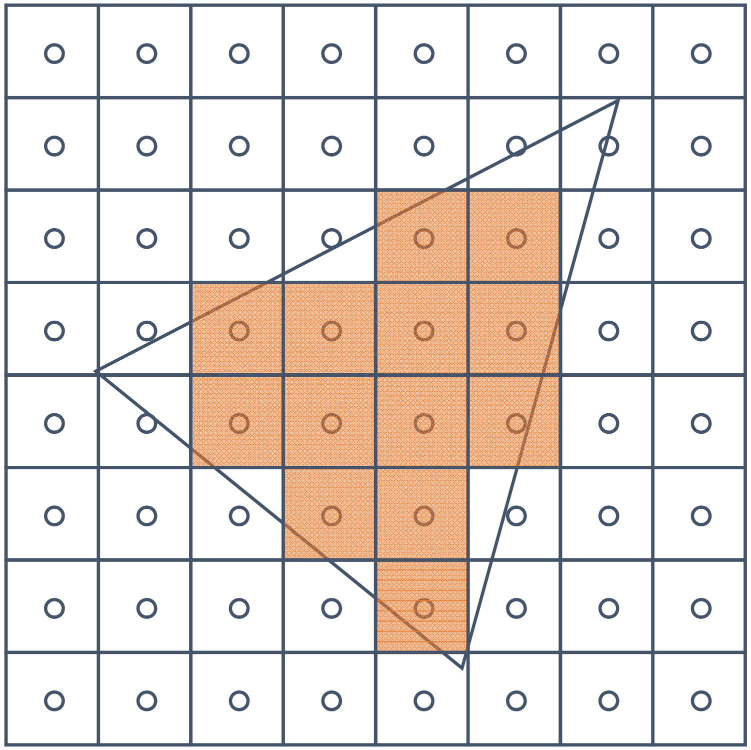


HOW TO CHECK IF A PIXEL IS INSIDE?



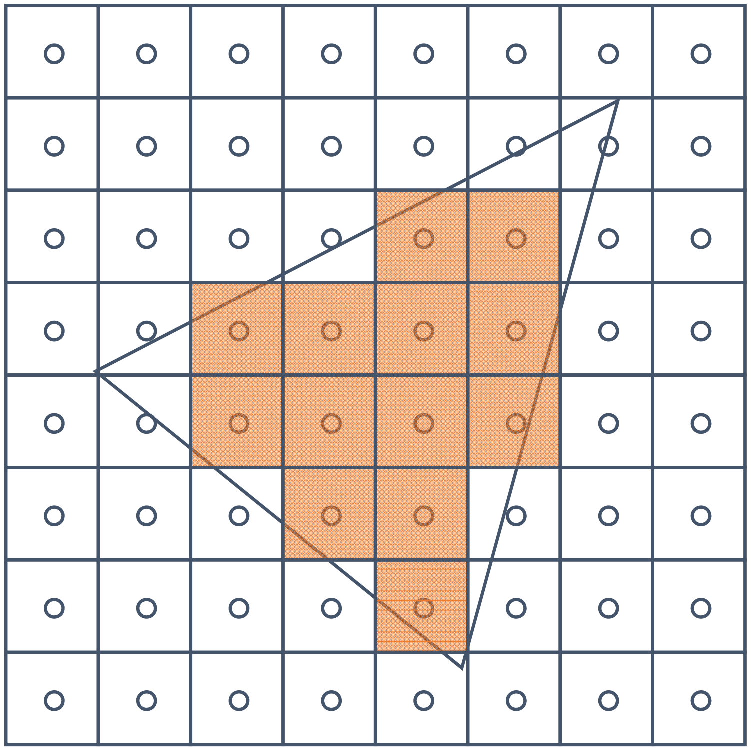
HOW TO CHECK IF A PIXEL IS INSIDE?

- Use implicit line equation:
  - $Ax + By + C = 0$
  - What is geometric meaning of A,B,C?
- How to find A,B,C?
- Orientation?



HOW TO CHECK IF A PIXEL IS INSIDE?

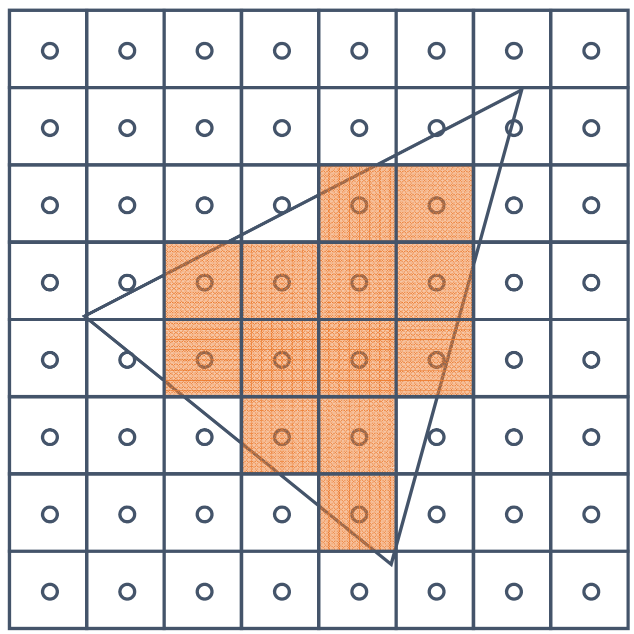
- Use implicit line equation:
  - $Ax + By + C = 0$
  - What is geometric meaning of A,B,C?
    - **(A,B) is a normal (not unit!) to the line**
    - **C is translation of that line**
- How to find A,B,C?
  - Option 1. Solve a system of 2 equations
  - Option 2. Find any normal
- Orientation?
  - Normal points in positive side



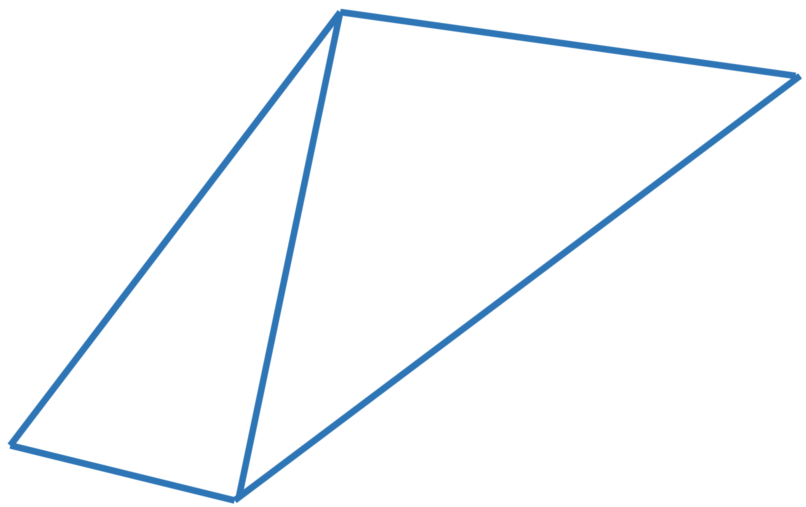
HOW TO CHECK IF A PIXEL IS INSIDE?

A point is inside ⇔

$$A_i x + B_i y + C > 0, i = 1, \ldots, 3$$

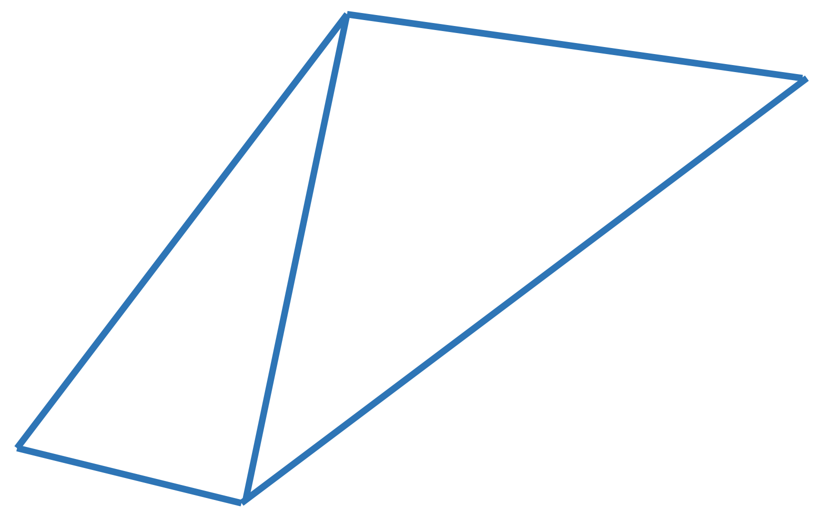


HOW TO TREAT BOUNDARY?



HOW TO TREAT BOUNDARY?

- If two triangles share an edge, scan conversion should be consistent
  - No pixel drawn twice
  - No gaps
- Strategy ideas?

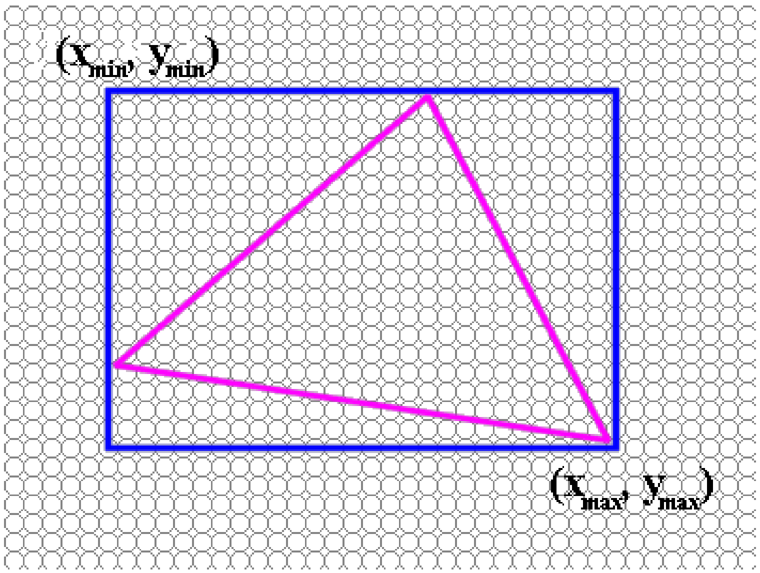


NAÏVE SCAN CONVERSION

- Testing every pixel is suboptimal
- Better ideas?

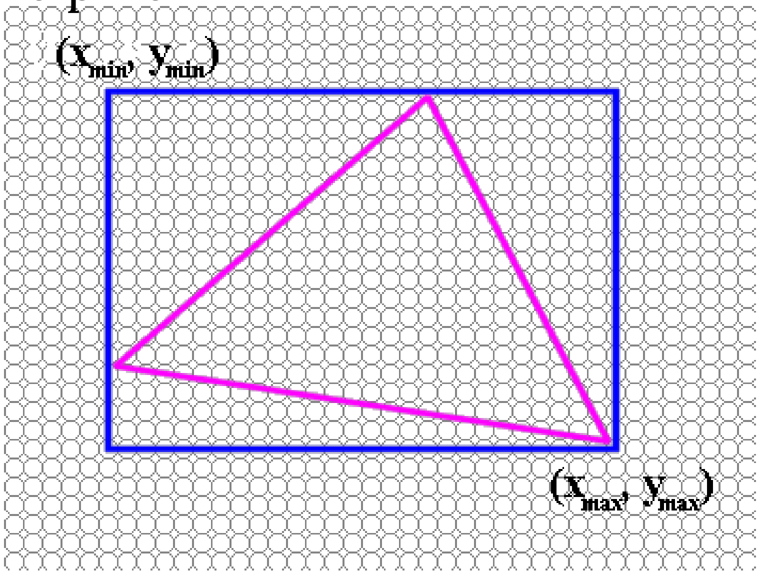
LESS NAÏVE SCAN CONVERSION

- Go over each pixel in bounding rectangle
- Check if pixel is inside/outside of triangle
  - Use sign of edge equations



SCANLINE IDEA (SIMPLIFIED)

- Basic structure of code:
  - Setup: compute edge equations, bounding box
  - (Outer loop) For each scanline in bounding box...
  - (Inner loop) ...check each pixel on scanline, evaluating edge equations and drawing the pixel if all three are positive



SCANLINE: CODE

```
findBoundingBox(xmin, xmax, ymin, ymax);
setupEdges (a0,b0,c0,a1,b1,c1,a2,b2,c2);

for (int y = yMin; y <= yMax; y++) {
 for (int x = xMin; x <= xMax; x++) {
  float e0 = a0*x + b0*y + c0;
  float e1 = a1*x + b1*y + c1;
  float e2 = a2*x + b2*y + c2;
  if (e0 > 0 && e1 > 0 && e2 > 0)
    Image[x][y] = TriangleColor;
 }
}
```

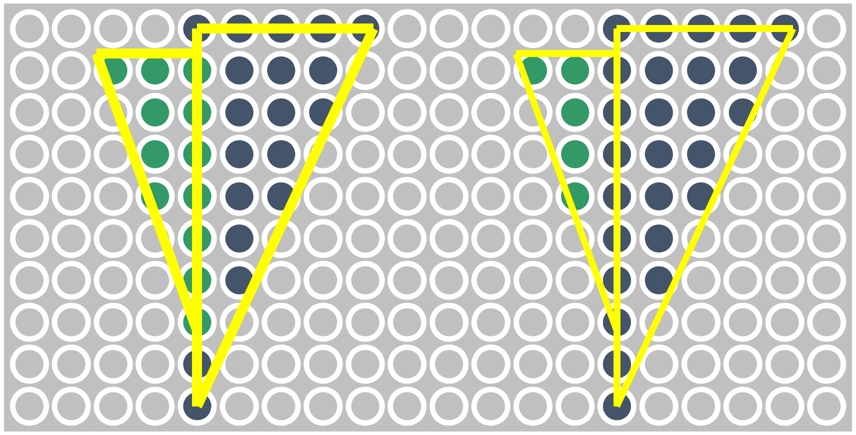
SCANLINE: OPTIMIZED CODE

```
// more efficient inner loop
for (int y = yMin; y <= yMax; y++) {
 float e0 = a0*xMin + b0*y + c0;
 float e1 = a1*xMin + b1*y + c1;
 float e2 = a2*xMin + b2*y + c2;
 for (int x = xMin; x <= xMax; x++) {
  if (e0 > 0 && e1 > 0 && e2 > 0)
    Image[x][y] = TriangleColor;

  e0 += a0;   e1+= a1;    e2 += a2;
 }
}
```

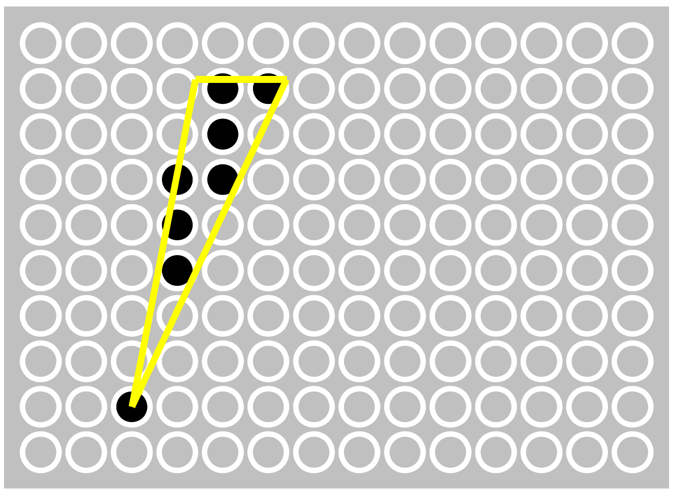
TRIANGLE RASTERIZATION ISSUES

- Exactly which pixels should be lit?
- A: Those pixels inside the triangle edges
- What about pixels exactly on the edge?

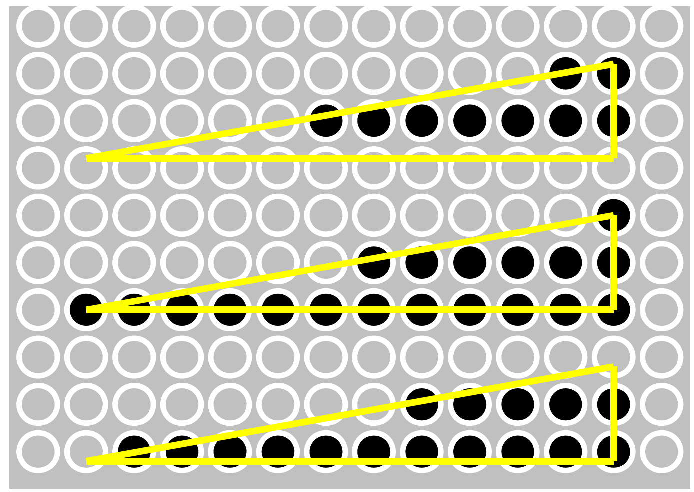


TRIANGLE RASTERIZATION ISSUES

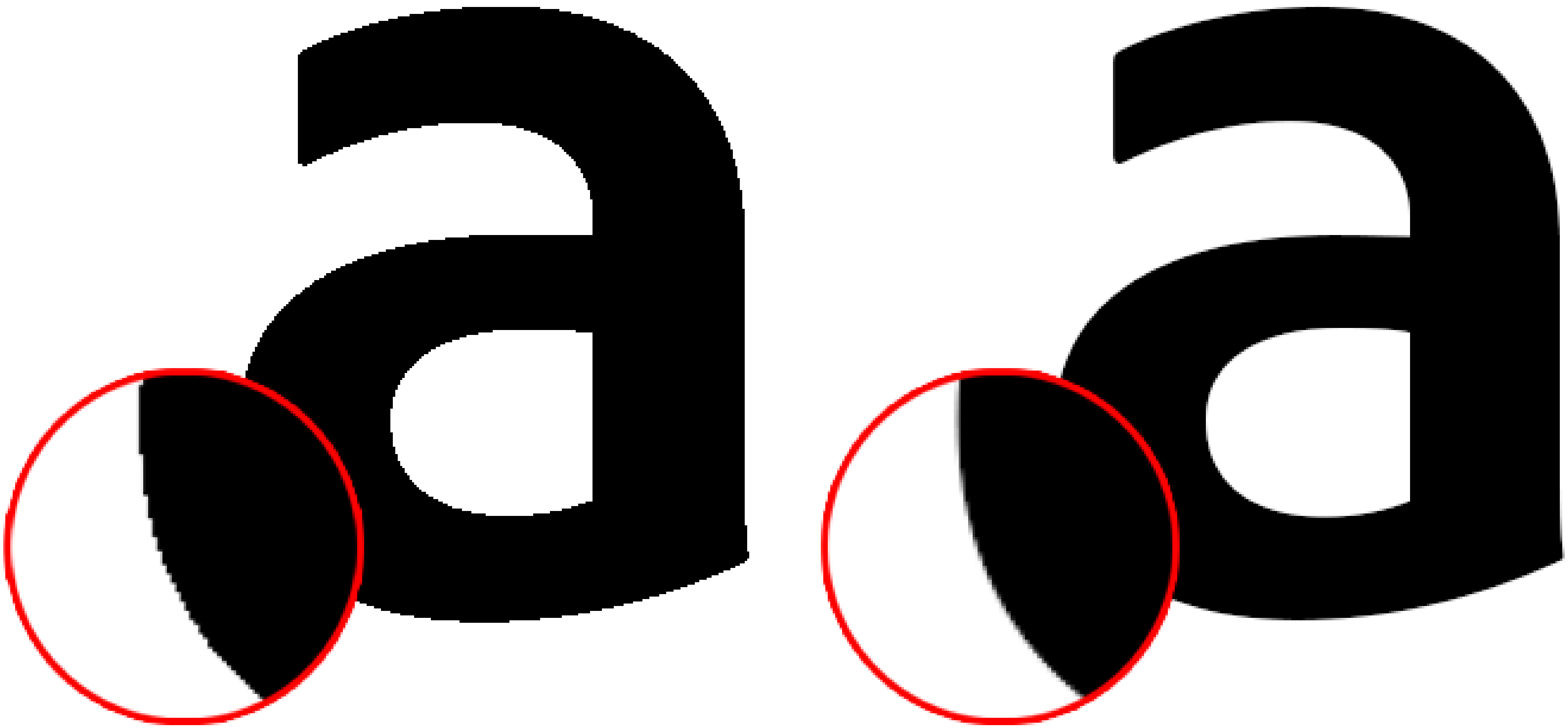
Sliver



- Moving Slivers

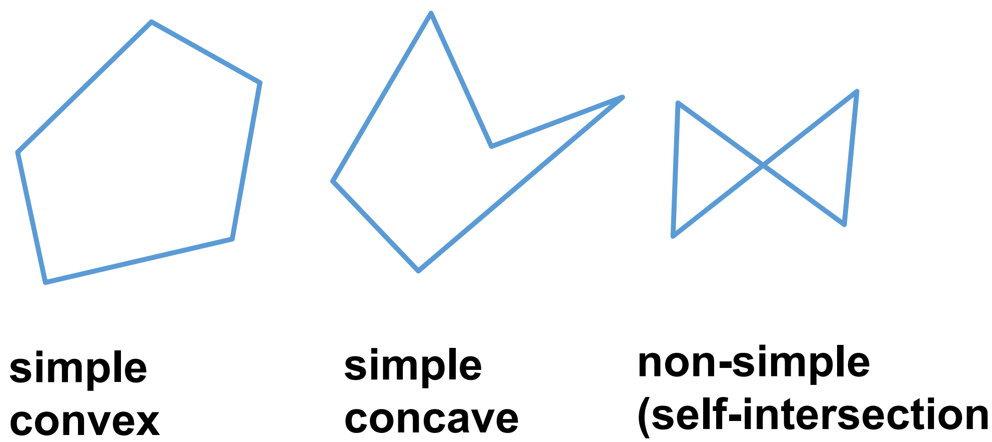


ALIASING & ANTI-ALIASING



Q: HOW TO TEST IF A POINT IS IN A POLYGON?

- Question: Which of these can we get from clipping?
  - A. Only triangles
  - B. Convex polygons
  - C. Simple non-convex
  - D. Non-simple

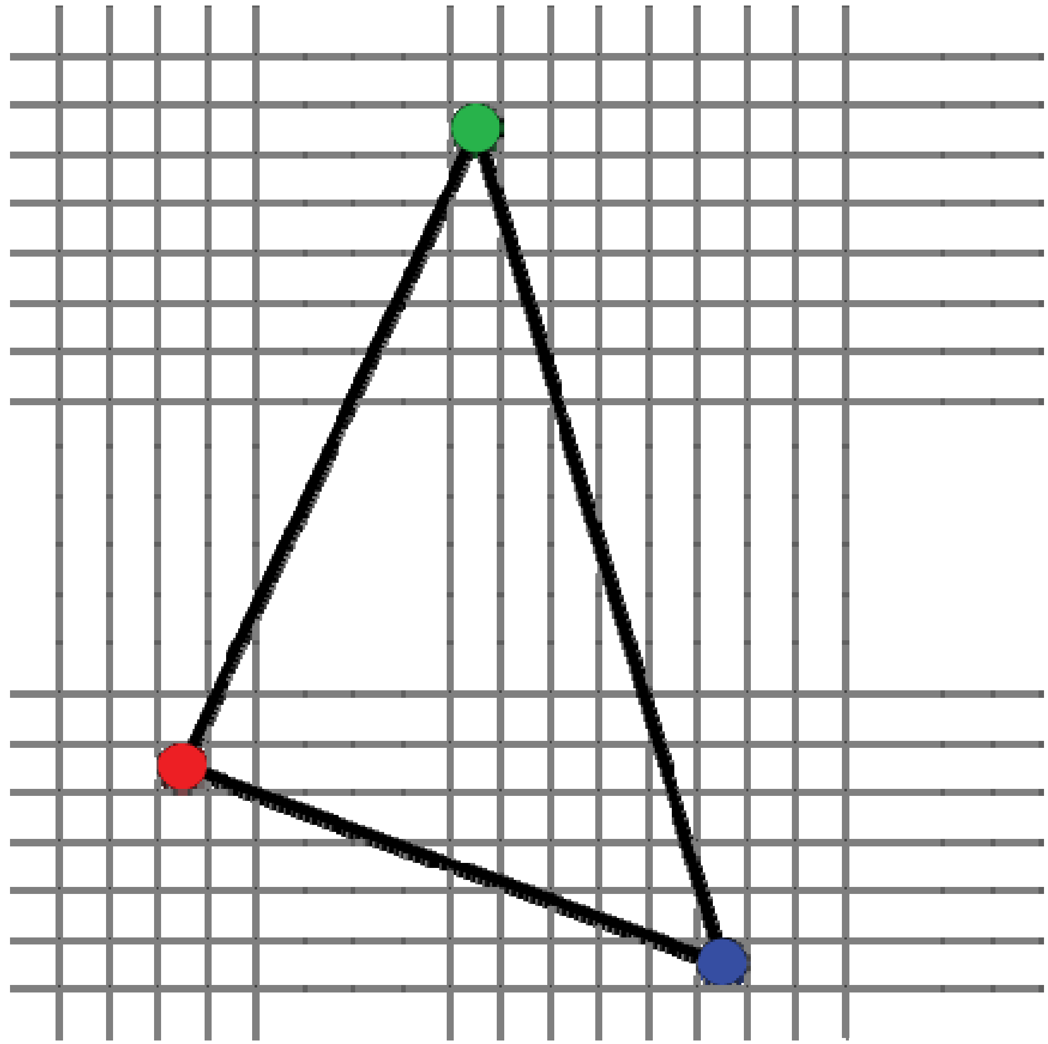


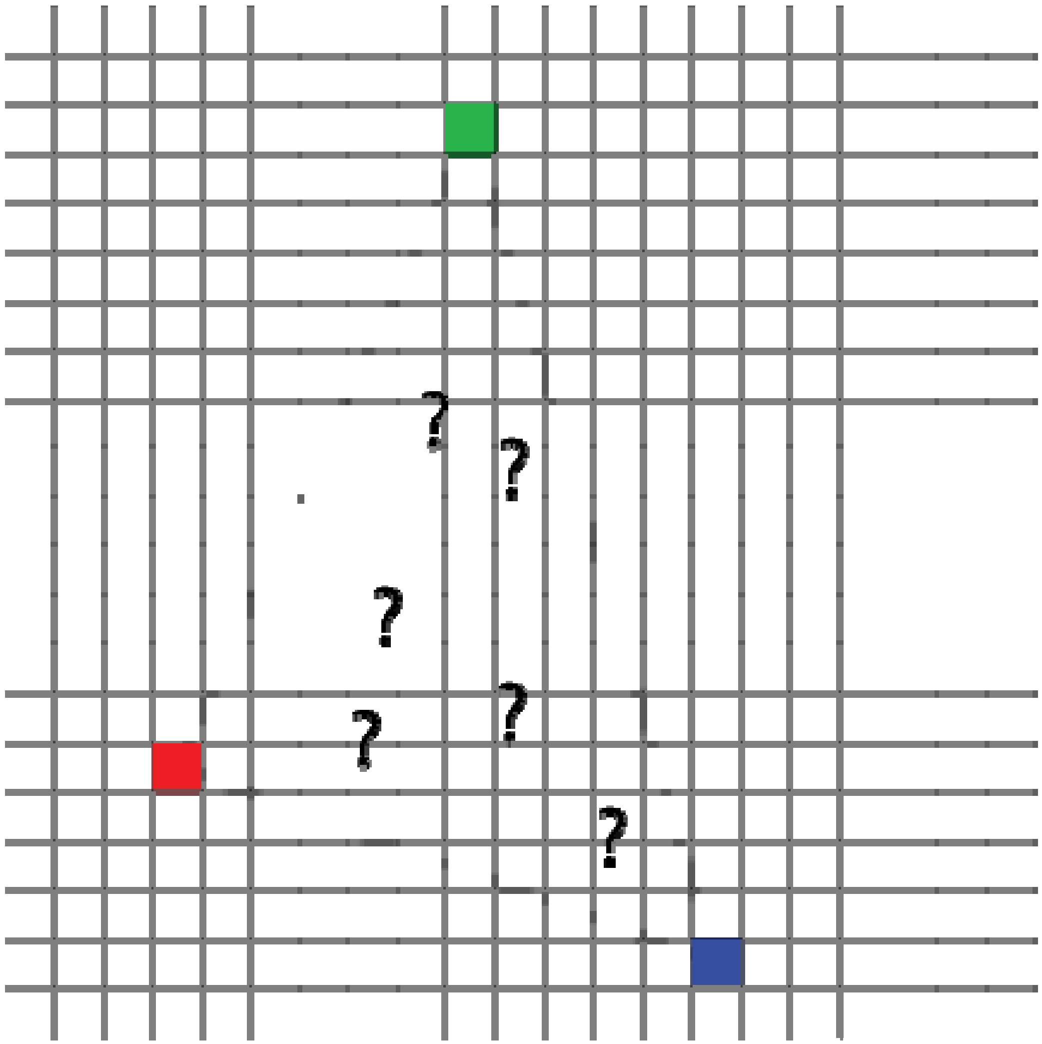
VALUES IN THE INTERIOR

Barycentric coordinates

INTERPOLATION – ACCESS TRIANGLE INTERIOR

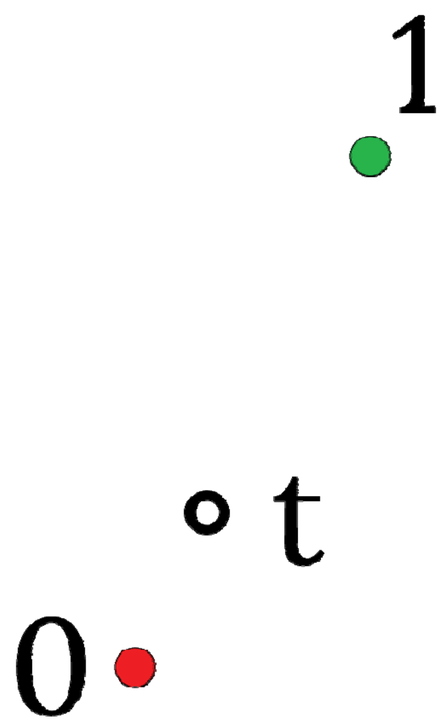
- Interpolate between vertices:
  - z
  - r,g,b - colour components
  - u,v - texture coordinates
  - N_x, N_y, N_z - surface normals
- Equivalent
  - Barycentric coordinates
  - Bilinear interpolation
  - Plane Interpolation





SIMPLER:

How to interpolate color between two points?

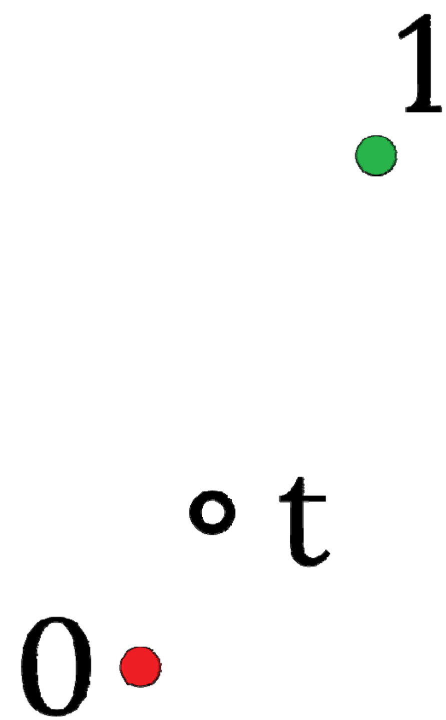


SIMPLER:

How to interpolate color between two points?

$$c(t) = c(0) \cdot (1 - t) + c(1) \cdot t$$

Linear interpolation

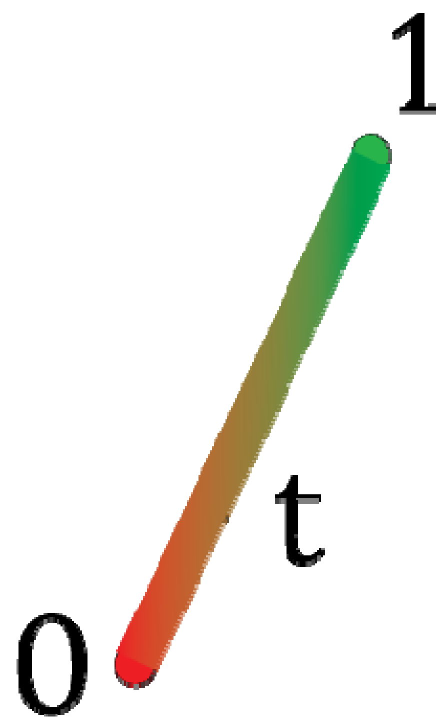


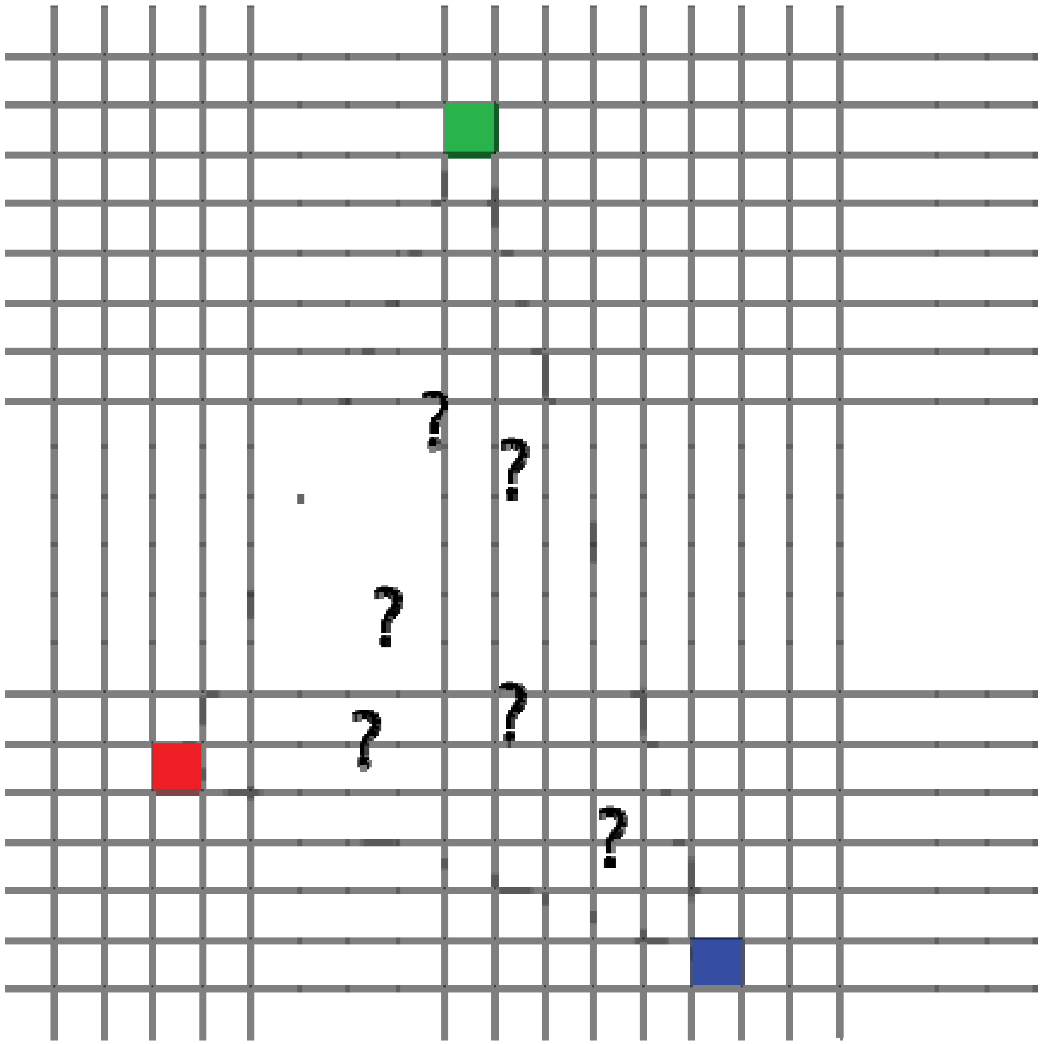
SIMPLER:

How to interpolate color between two points?

$$c(t) \approx c(0) \cdot (1 - t) + c(1) \cdot t$$

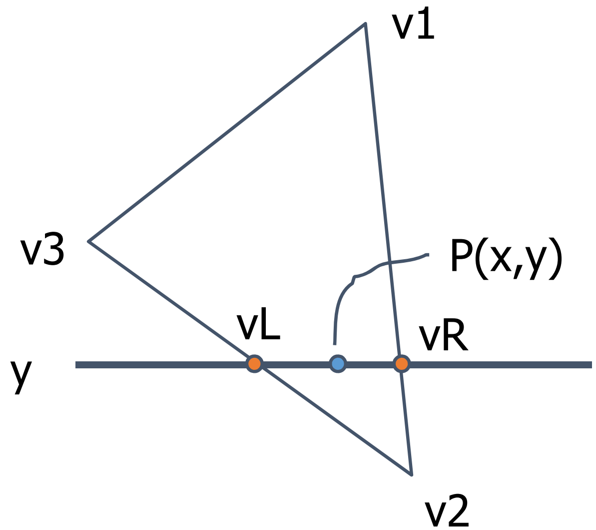
Linear interpolation



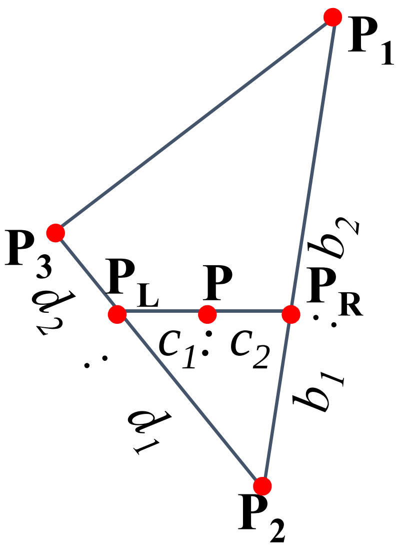


SIMPLE GENERALIZATION: BI-LINEAR INTERPOLATION

- Interpolate quantity along L and R edges
  - (as a function of y)
  - Then interpolate quantity as a function of x



BI-LINEAR INTERPOLATION



$$P = \frac{c_2}{c_1 + c_2} \cdot P_L + \frac{c_1}{c_1 + c_2} \cdot P_R$$

$$P_L = \frac{d_2}{d_1 + d_2} P_2 + \frac{d_1}{d_1 + d_2} P_3$$

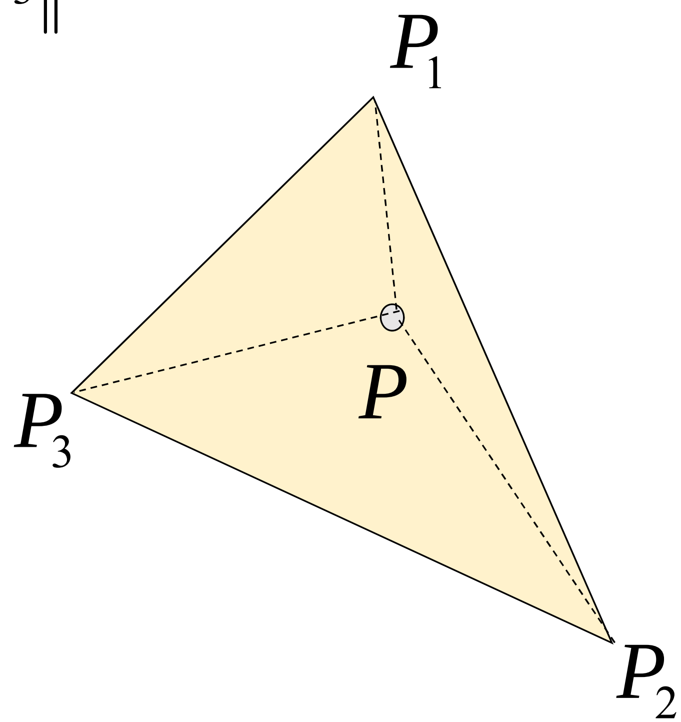
$$P_R = \frac{b_2}{b_1 + b_2} P_2 + \frac{b_1}{b_1 + b_2} P_1$$

$$P = \frac{c_2}{c_1 + c_2}\left(\frac{d_2}{d_1 + d_2} P_2 + \frac{d_1}{d_1 + d_2} P_3\right) + \frac{c_1}{c_1 + c_2}\left(\frac{b_2}{b_1 + b_2} P_2 + \frac{b_1}{b_1 + b_2} P_1\right)$$

BARYCENTRIC COORDINATES

- Area

$$A = \frac{1}{2}\left\| \overrightarrow{P_1P_2} \times \overrightarrow{P_1P_3} \right\|$$



- Barycentric coordinates

$$a_1 = A_{P_2P_3P} / A, a_2 = A_{P_3P_1P} / A,$$

$$a_3 = A_{P_1P_2P} / A,$$

$$P = a_1P_1 + a_2P_2 + a_3P_3$$

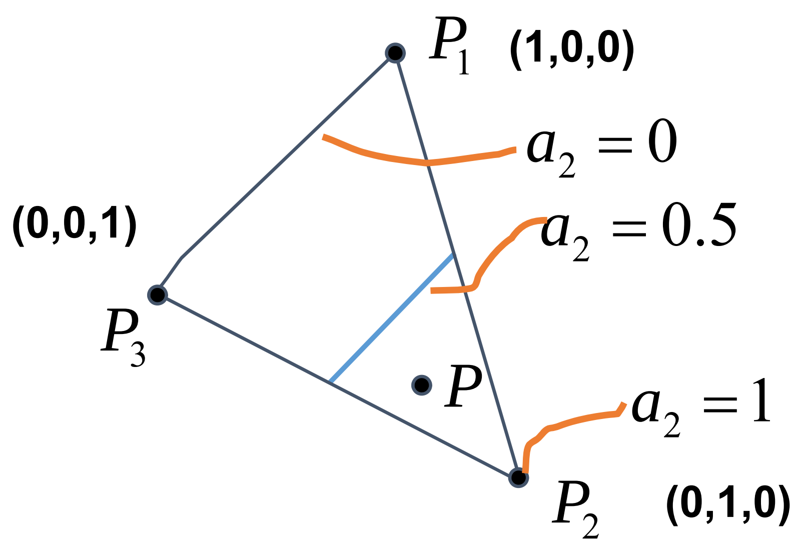
BARYCENTRIC COORDINATES

•weighted (affine) combination of vertices

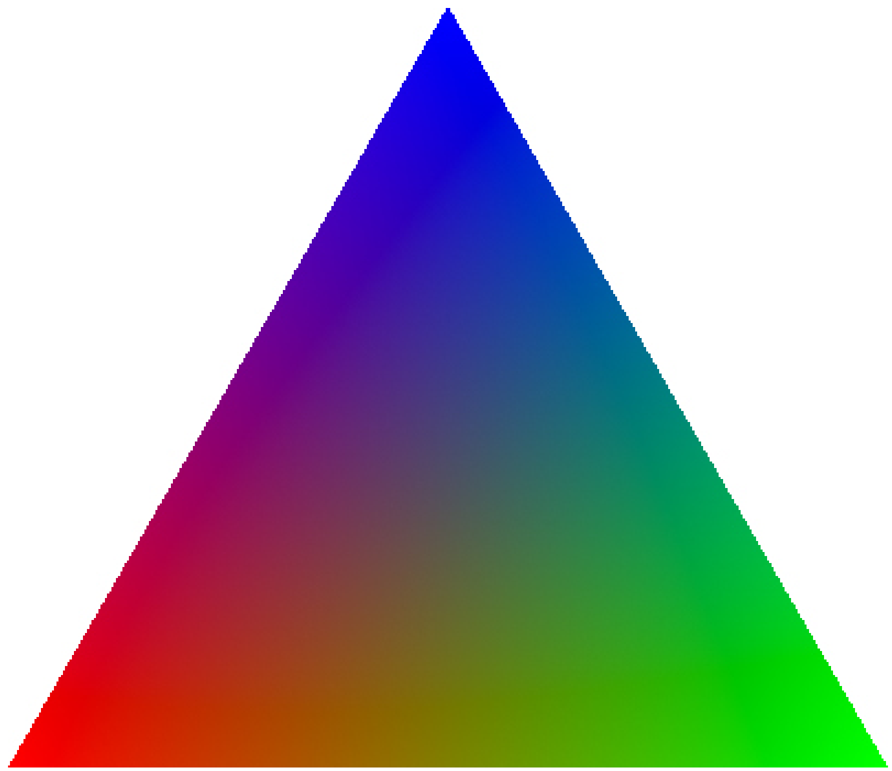
$$P = a_1 \cdot P_1 + a_2 \cdot P_2 + a_3 \cdot P_3$$

$$a_1 + a_2 + a_3 = 1$$

$$0 \le a_1, a_2, a_3 \le 1$$



BARYCENTRIC COORDINATES



NOTE:

- In reality, only two values are enough to encode a point in a triangle
- We added a 3rd one – a similar idea to homogeneous coordinates!

- Those are, however, unique because of this:

$$a_1 + a_2 + a_3 = 1$$

BARYCENTRIC COORDINATES

- Are used to interpolate
  - z
  - all varying variables
    - color
    - normals
  - Why do we interpolate z?
  - Problems when using perspective camera. We'll see later (in texture mapping)