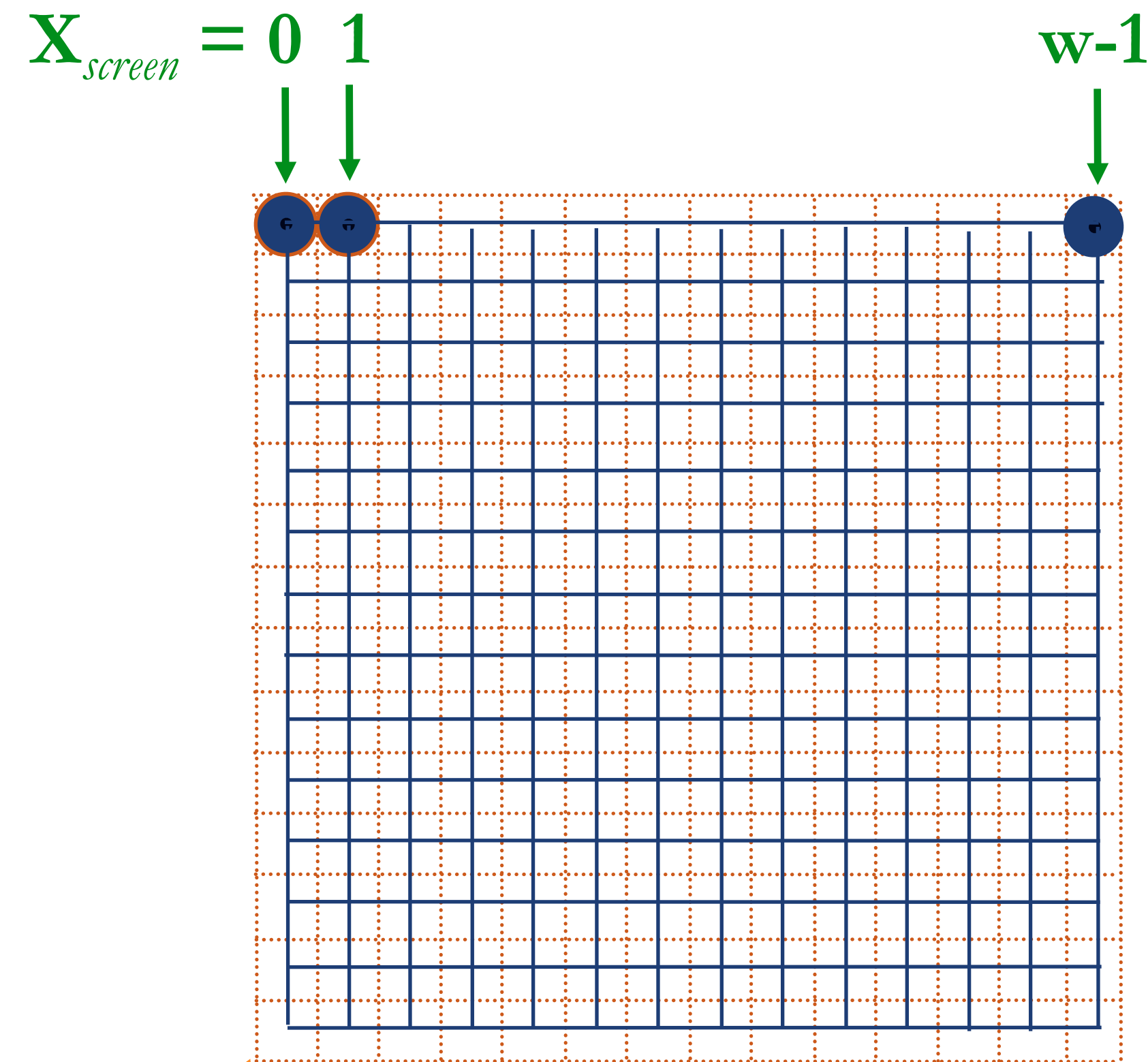


Computer Graphics

Rasterization

Subodh Kumar, Dept of Computer Sc. & Engg., IIT Delhi

Pixel/Screen Coordinates



Caution:

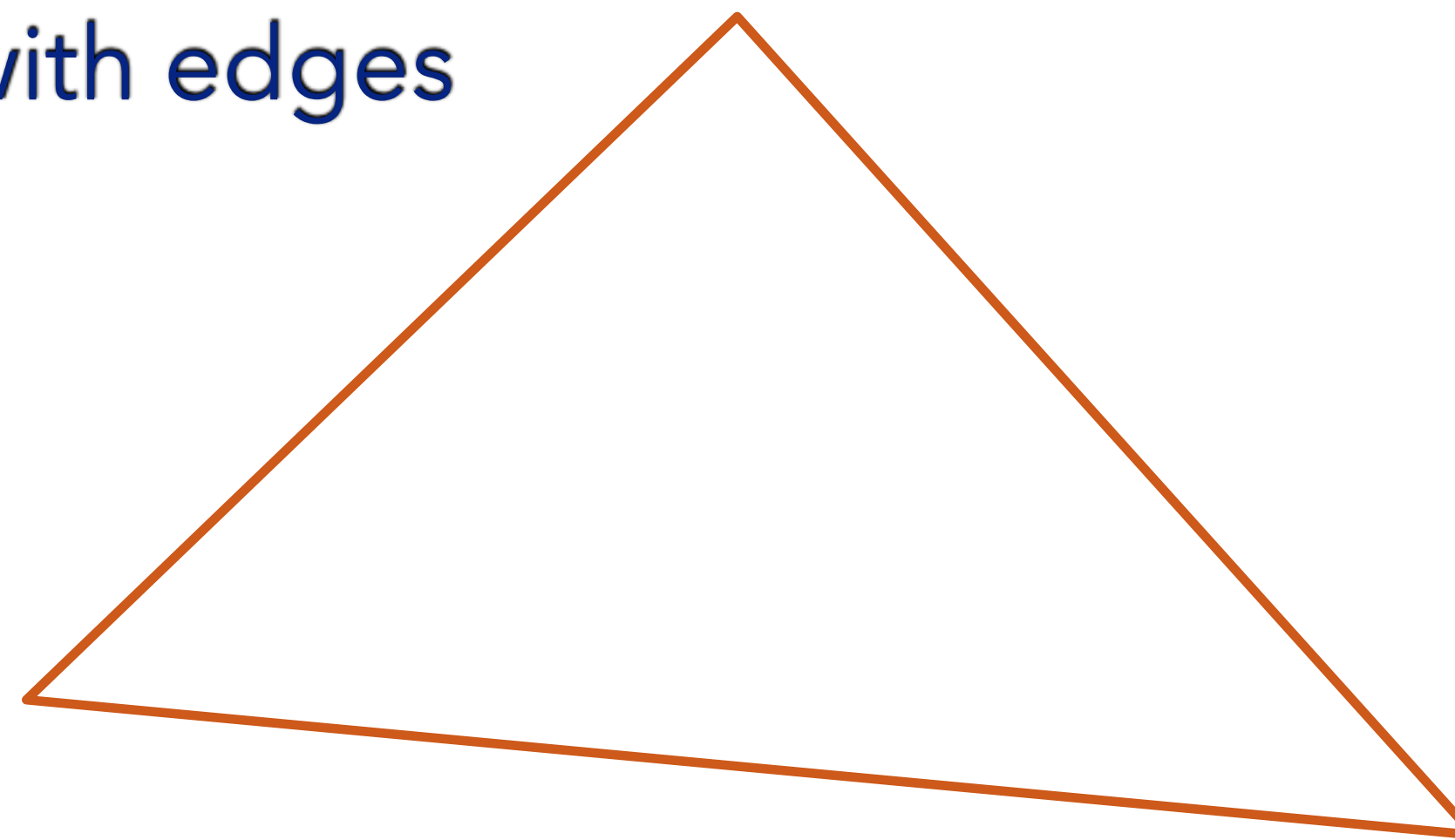
In OpenGL this is $X = -1$, pixel centers are at $\text{int} + .5$ pixels



Polygon Rasterization

Sample

- ~~Draw pixels~~ "intersected" by polygon
 - ➡ But one color per pixel.. Can have multiple samples per pixel, but both names are commonly used in drawing
 - ➡ And 3D polygon?
- Can iterate over rows (scan lines)
 - ➡ Find intersection of scan-lines with edges
 - ▶ All lines and edges?

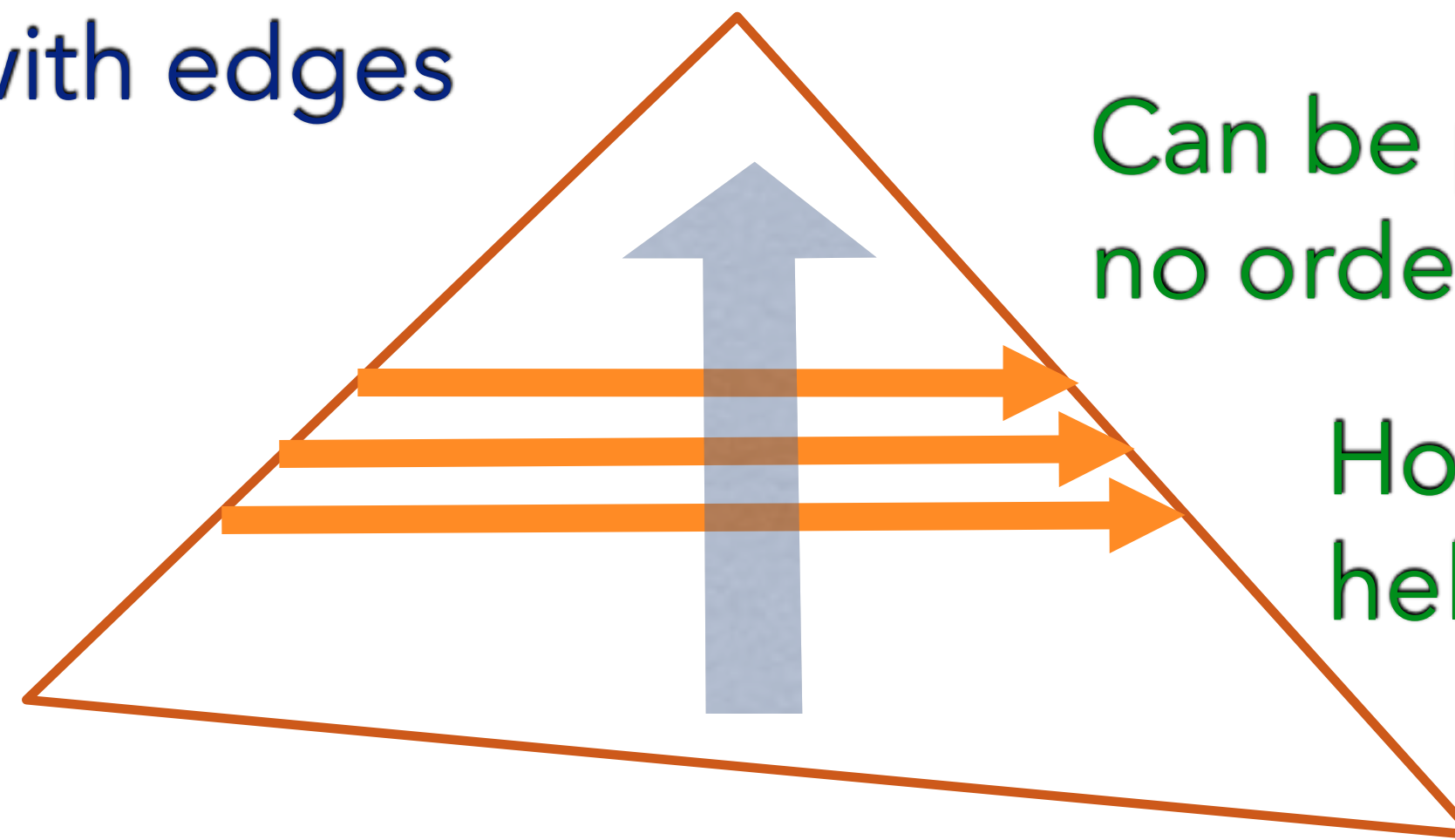




Polygon Rasterization

Sample

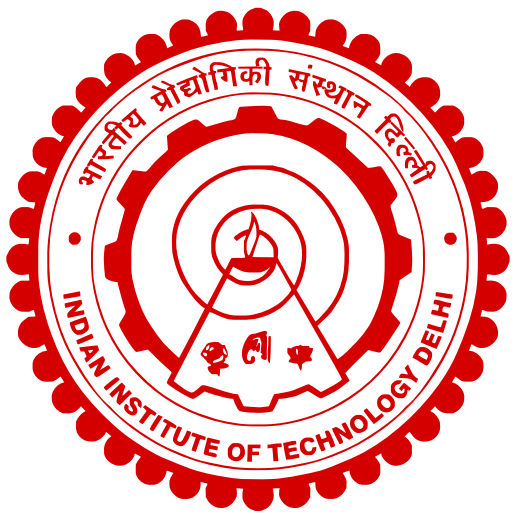
- Draw pixels "intersected" by polygon
 - ➡ But one color per pixel.. Can have multiple samples per pixel, but both names are commonly used in drawing
 - ➡ And 3D polygon?
- Can iterate over rows (scan lines)
 - ➡ Find intersection of scan-lines with edges
 - All lines and edges?



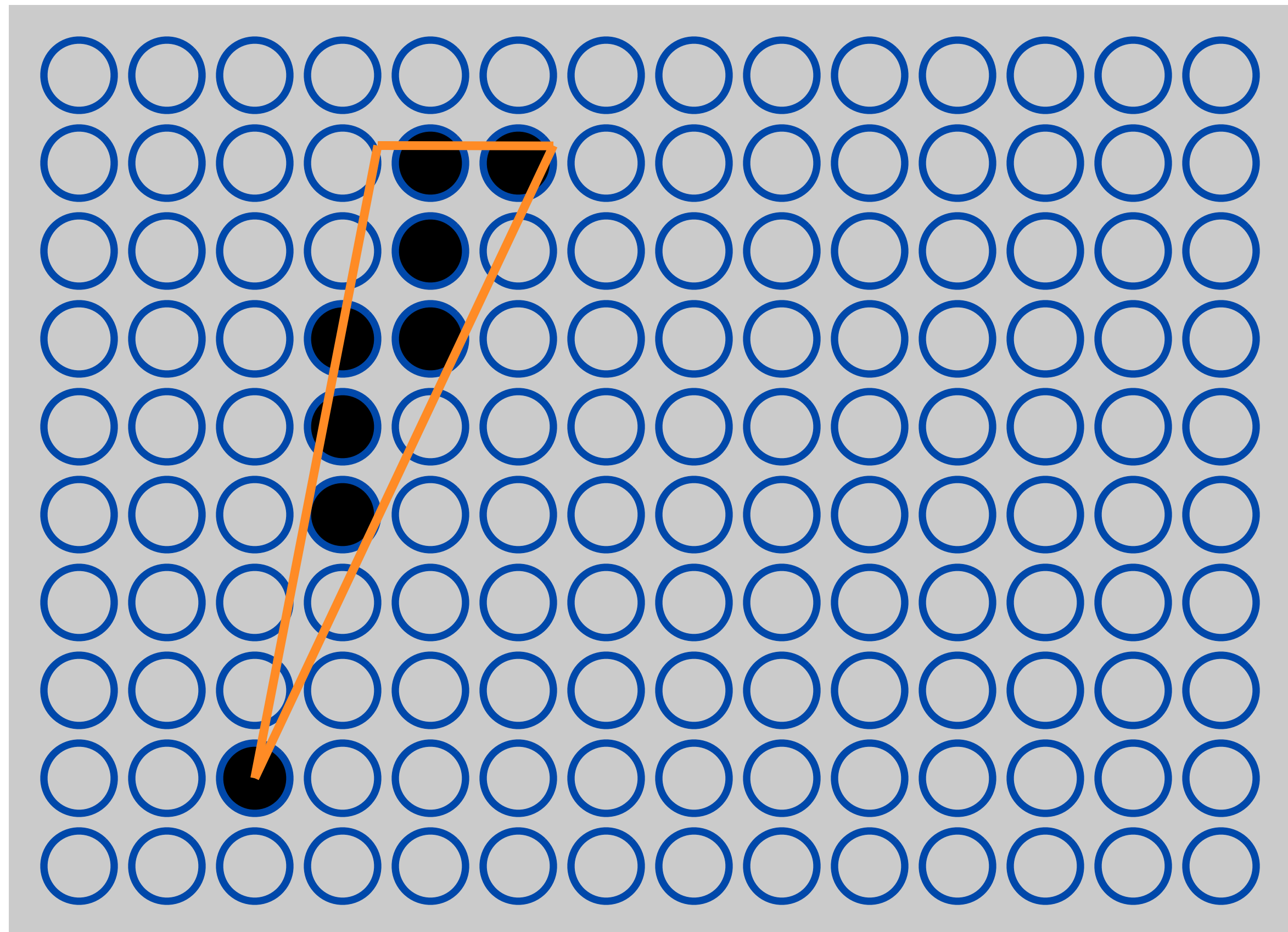
Can be parallelized as
no ordering is required

However, some ordering
helps with optimization

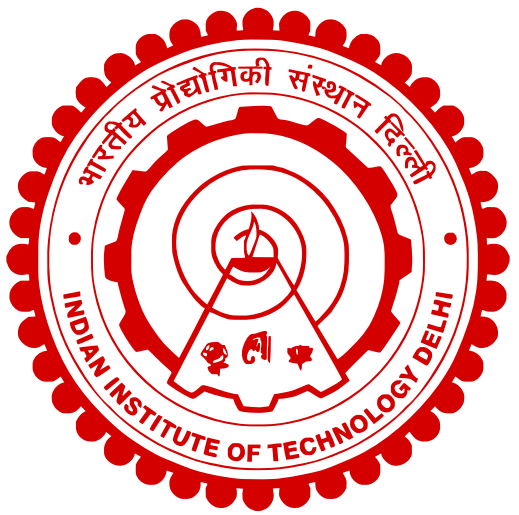
Triangle Rasterization Issues



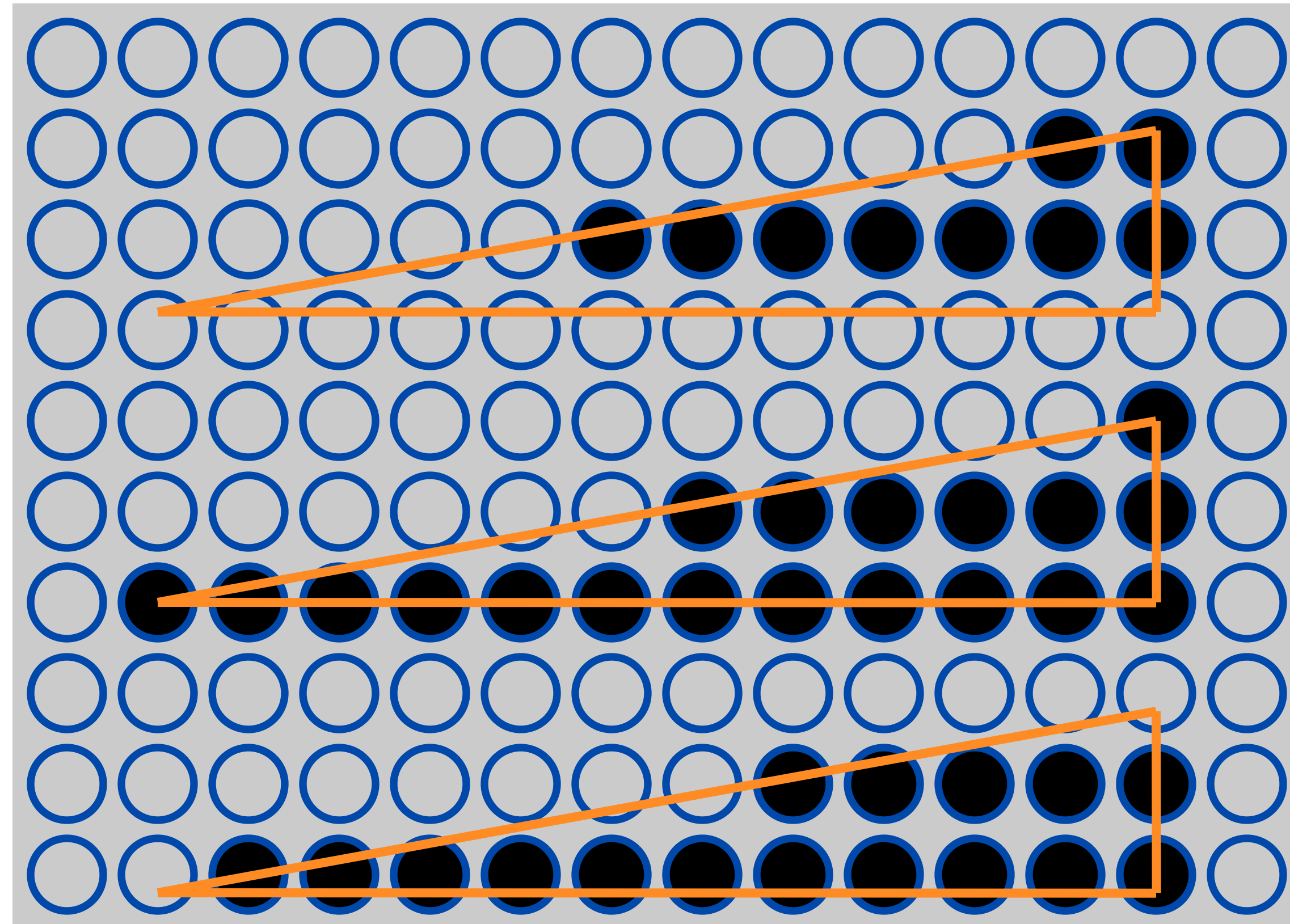
- Sliver problem



Triangle Rasterization Issues



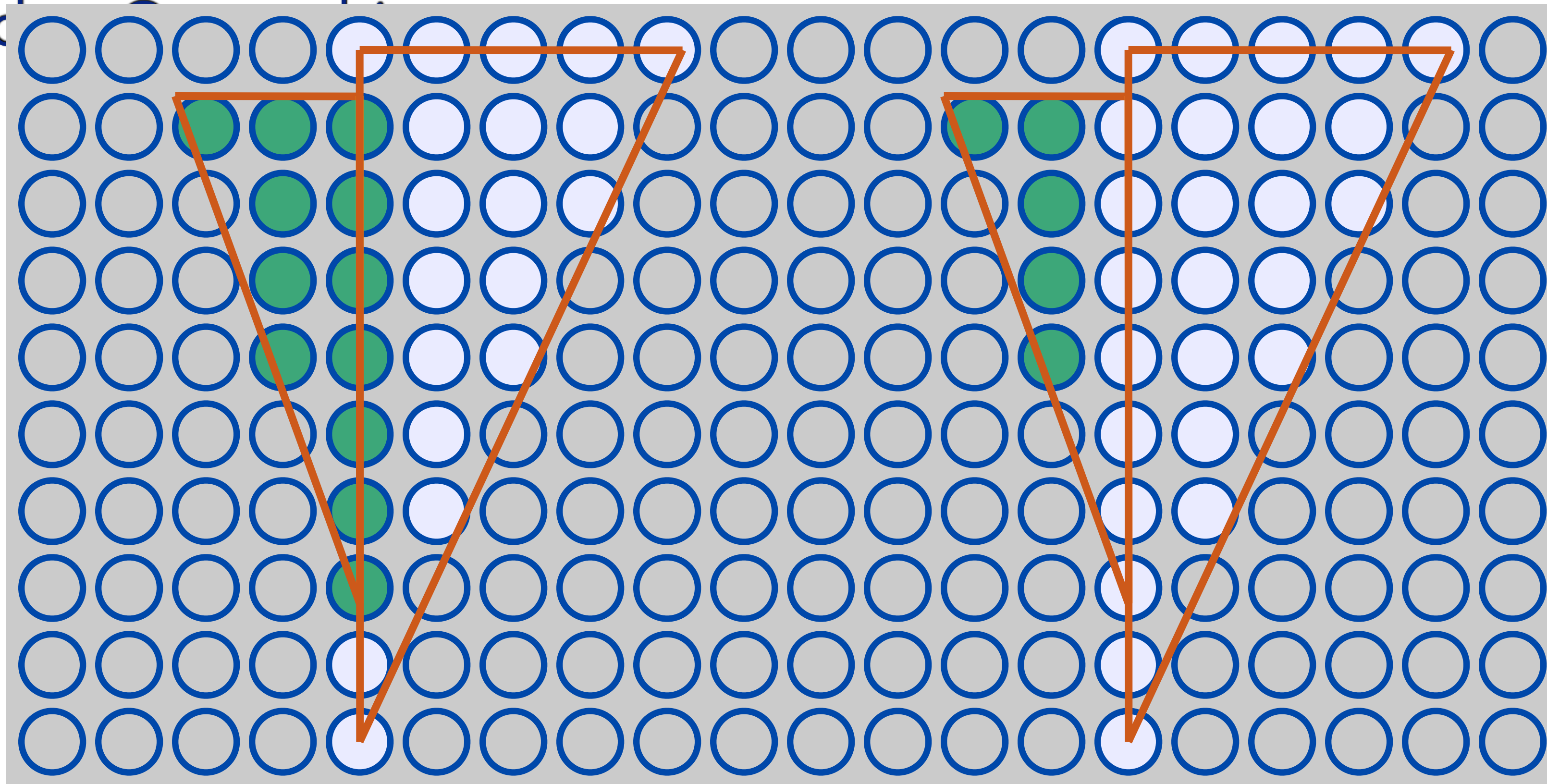
- Animating Slivers



Triangle Rasterization Issues



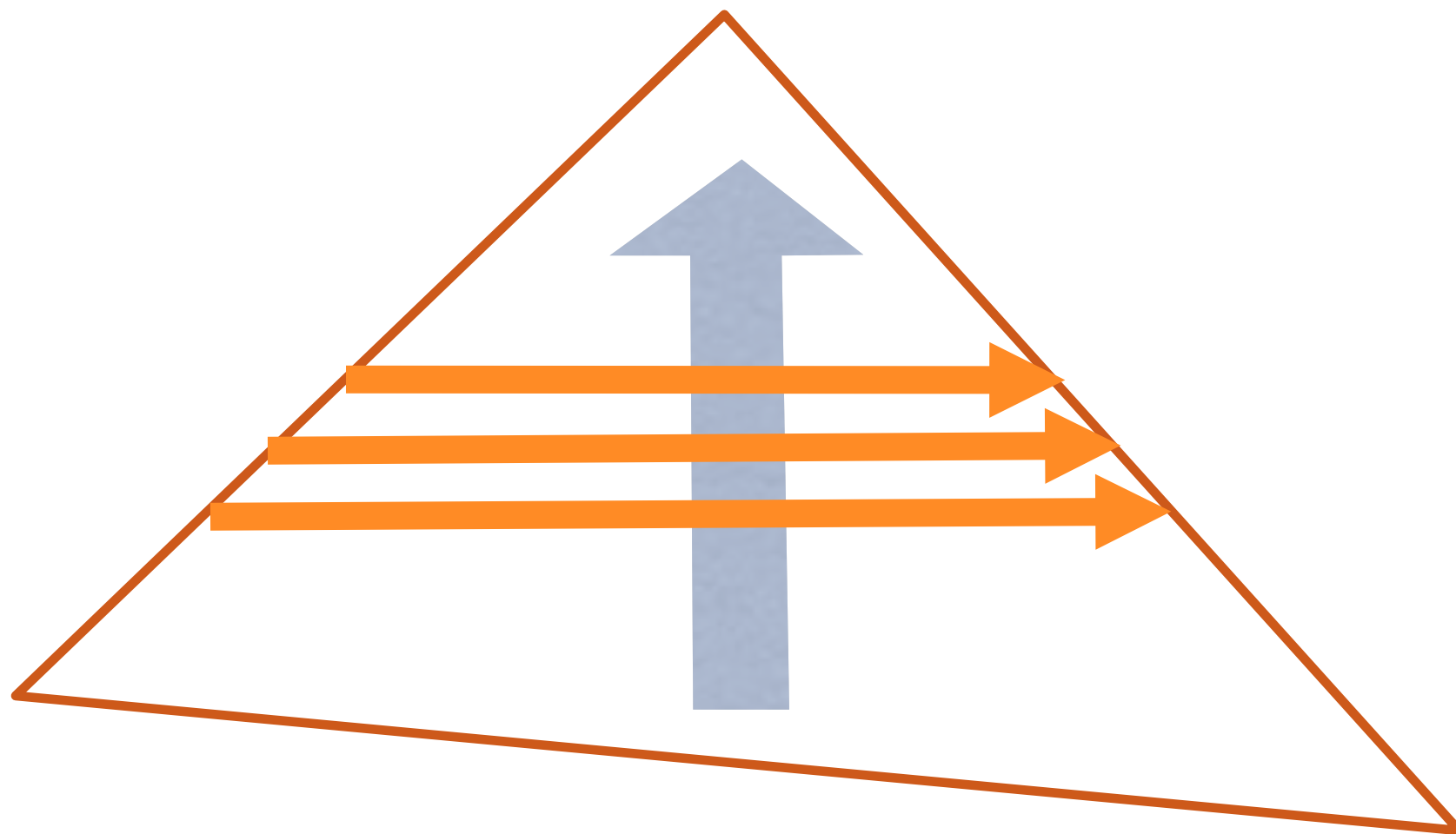
- Shared Ed





Polygon Rasterization Rules

- Draw pixel, if center inside polygon
 - ➡ What if it is at an edge?
 - ➡ Draw on left edge, Skip on right edge
 - ➡ Unless Horizontal
 - ▶ Then, draw a bottom edge, skip a top edge



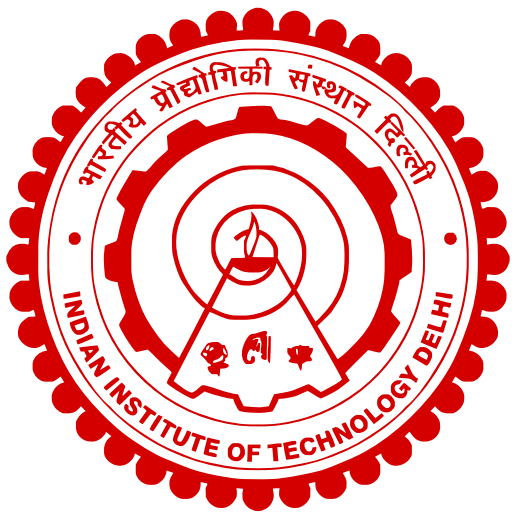
Find lo , hi y (scanline)

For $y=lo$ to hi :

Find *left* extreme (of span)

Find *right* extreme (of span)

For *left* to *right*, draw pixel



Triangle Rasterization

- Extrema for a scan line are on two edges (sometimes vertices)

- ➔ Which two edges?

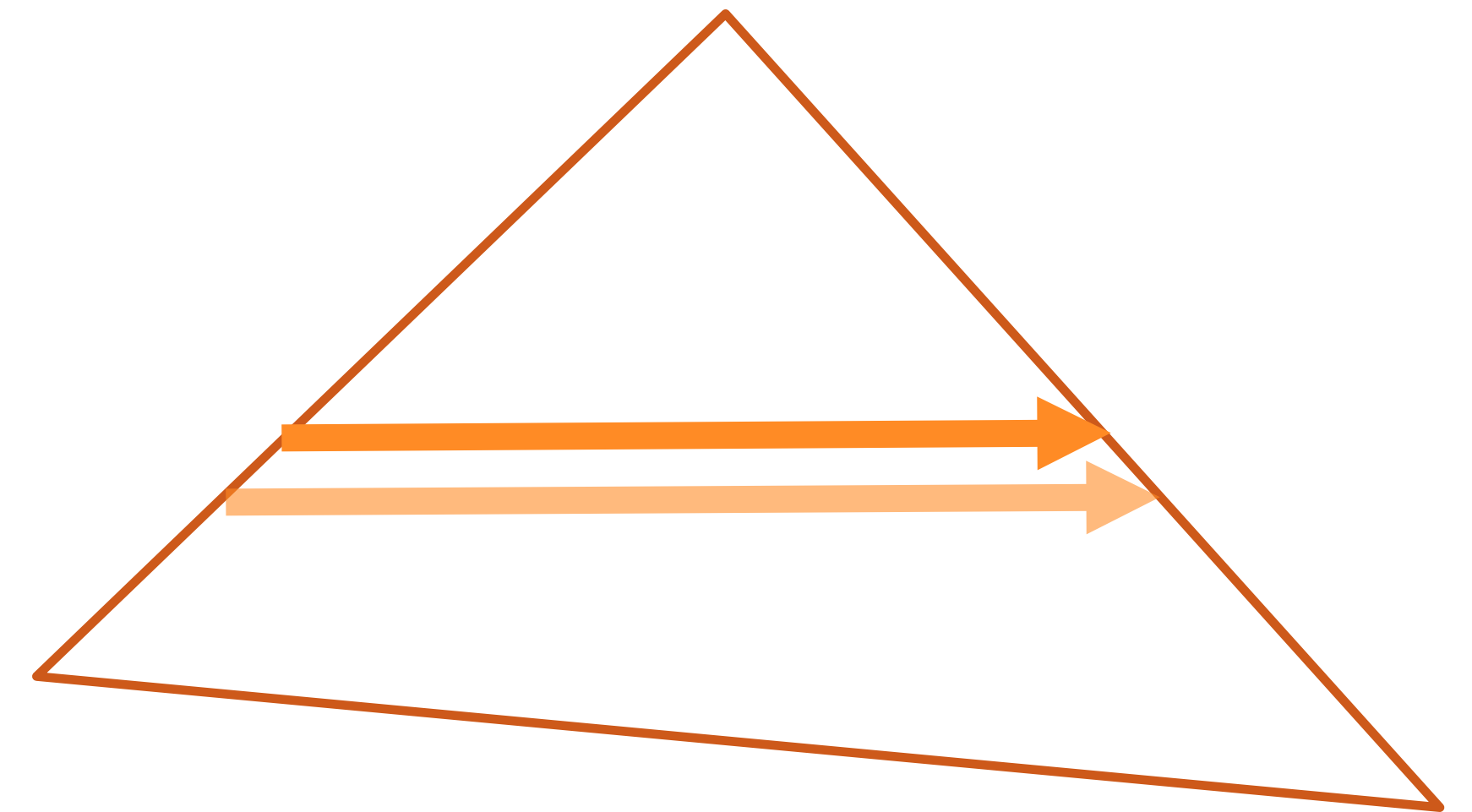
- ▶ Start with two at the lowest vertex (What if there are two lowest vertices?)
- ▶ When an edge ends (at a vertex), replace edge (What if both edges end?)

- For a scan line: find Left extreme

- ➔ Remember from previous line:

- ▶ $\text{StartX} = \text{StartX}_{\text{previous}} + \frac{1}{m}$ one pixel up

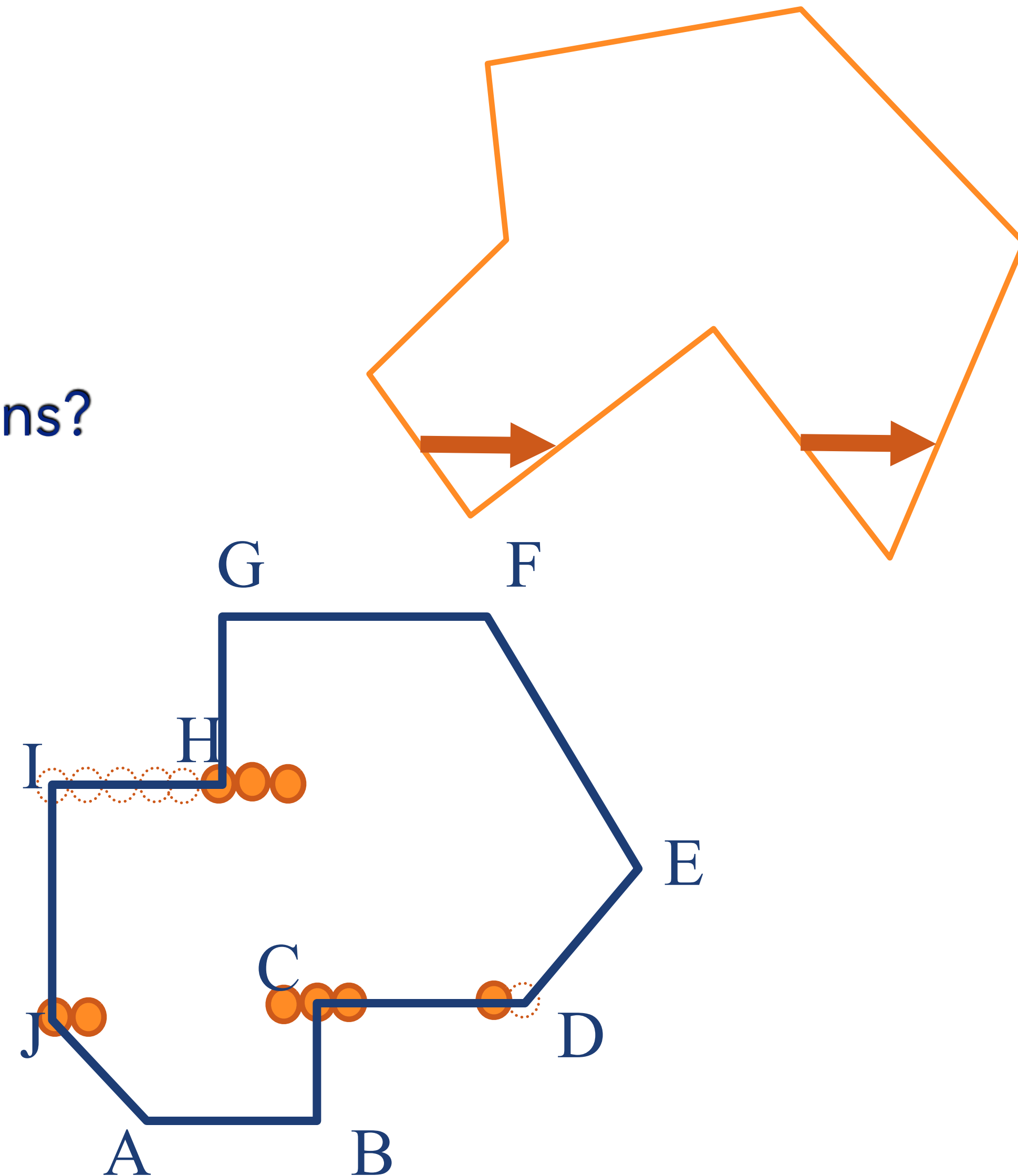
- Similarly find Right extreme

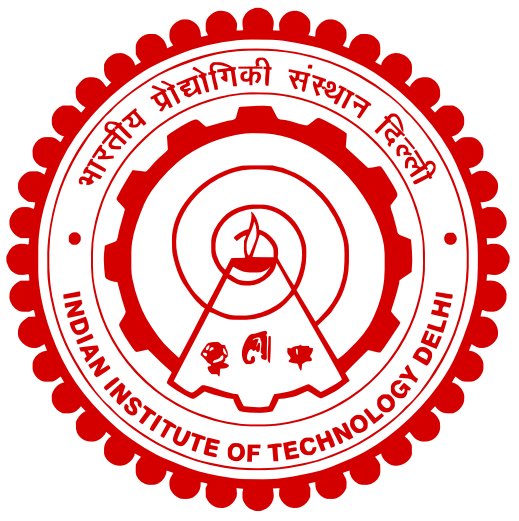


General Polygon Rasterization



- Multiple spans
- Count parity
 - ➔ How do you count vertex intersections?
- Divide into triangles





Triangle Rasterization

- Setup:

- ➡ Find inverse of edge slopes
- ➡ Find bounding box
- ➡ Initialize left, right intersection

- Rasterize:

- ➡ From Bounding Box bottom to Top:
 - Update left, right intersection (Round up on left, down on right)
 - Fill span => Generate fragment

Compute a, b, c

- Interpolate Z , Color?

- ➡ Defer to (a hidden part) of fragment shader: $ax+by+c$ on attribute fetch)

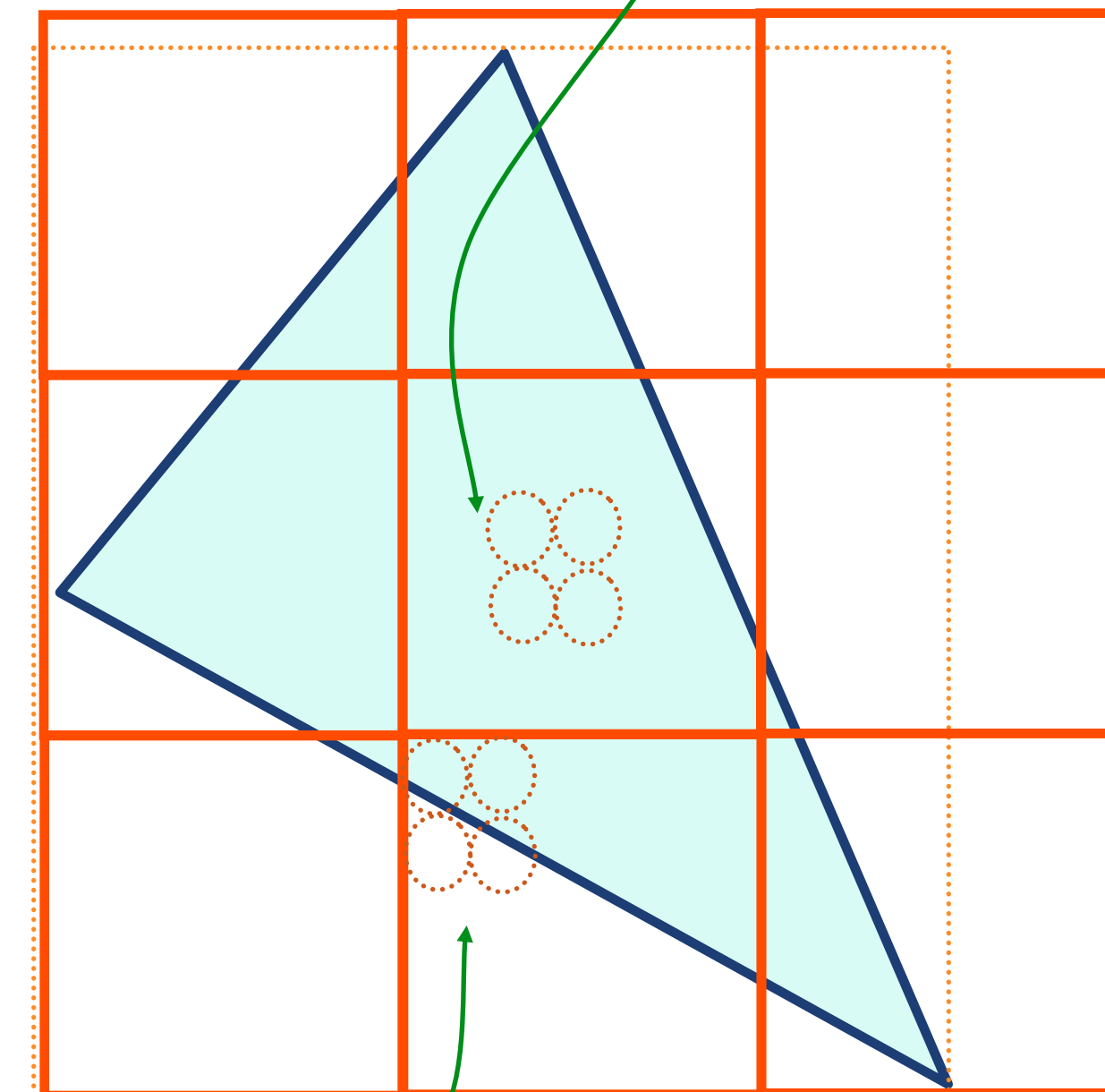


Triangle Rasterization, Really

- Multiple rasterizers?
 - ➔ Blocked or Strided
- Create tiles of fragments
 - ➔ In parallel, divide tiles into sub-tiles
 - ➔ Each sub-tile shaded together
 - Mask to turn off uncovered tiles

Partially covered sub-tile

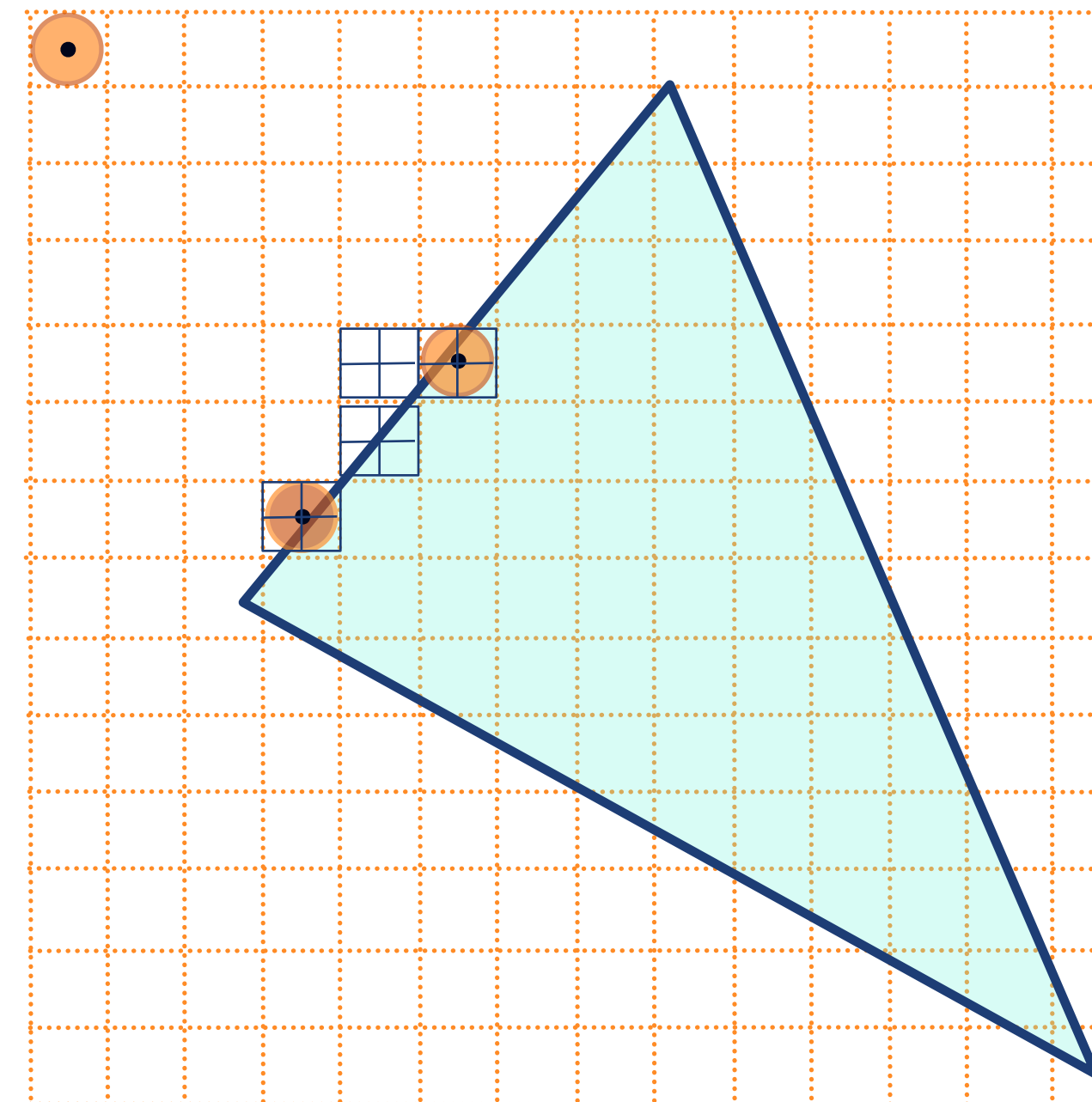
Fully covered sub-tile





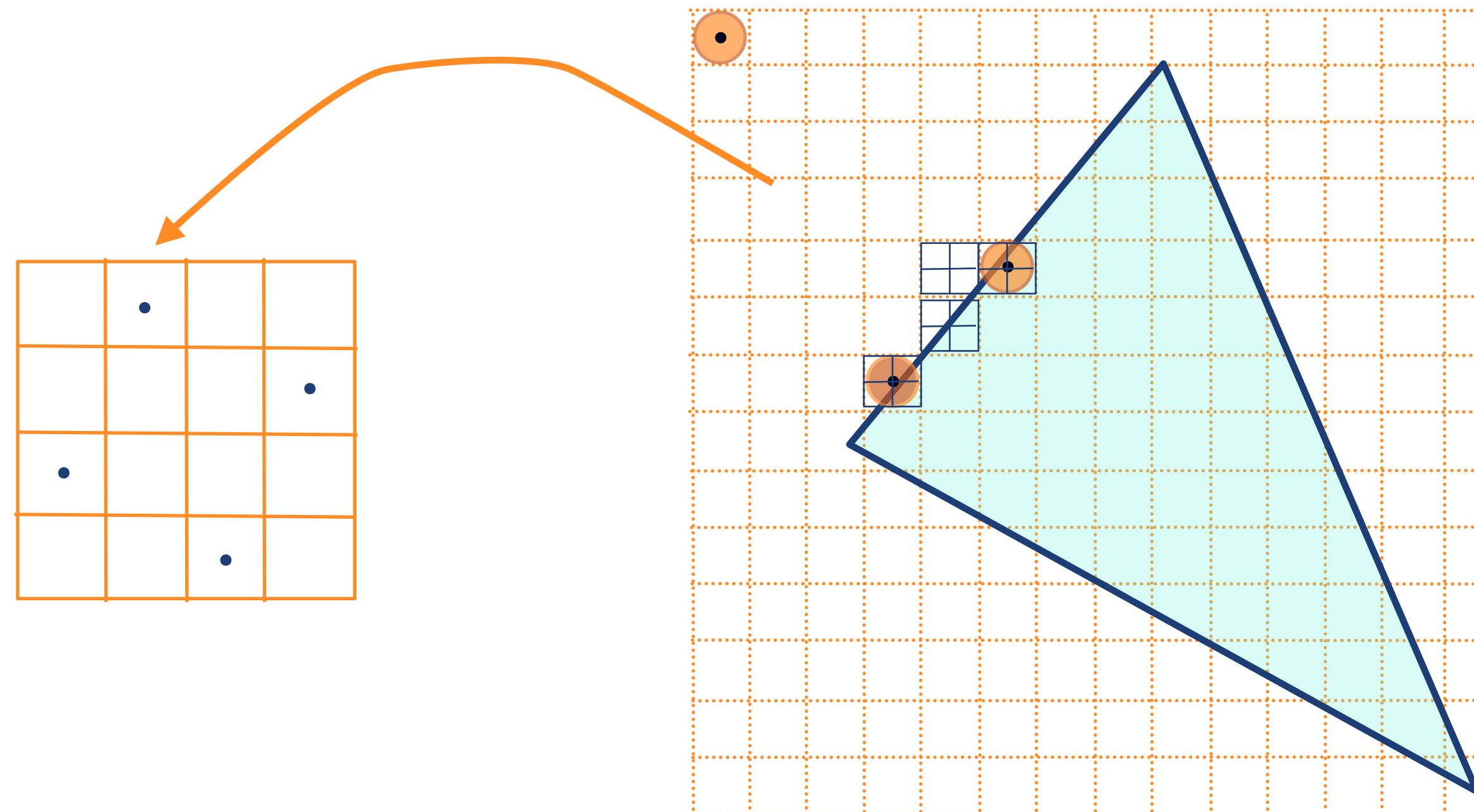
Multi-sampling

- An attempt to increase resolution
 - ➡ Anti-aliasing: more on this later

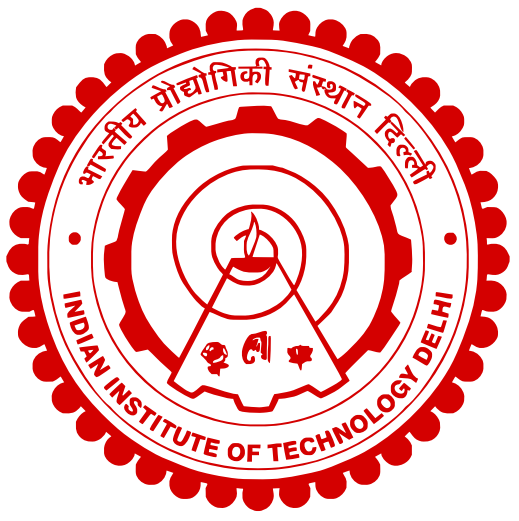


Multi-sampling

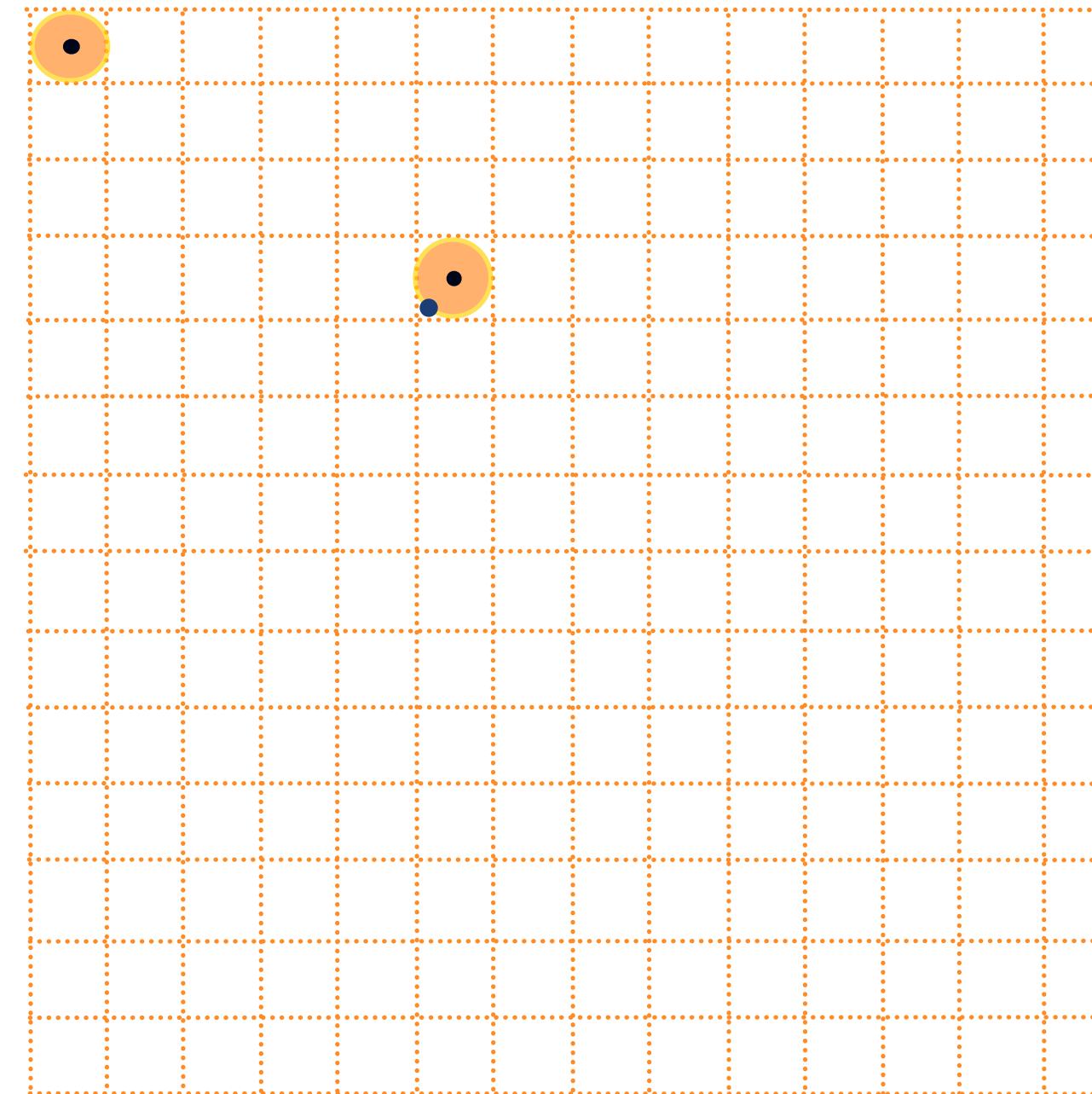
- An attempt to increase resolution
 - ➡ Anti-aliasing: more on this later



Point Drawing



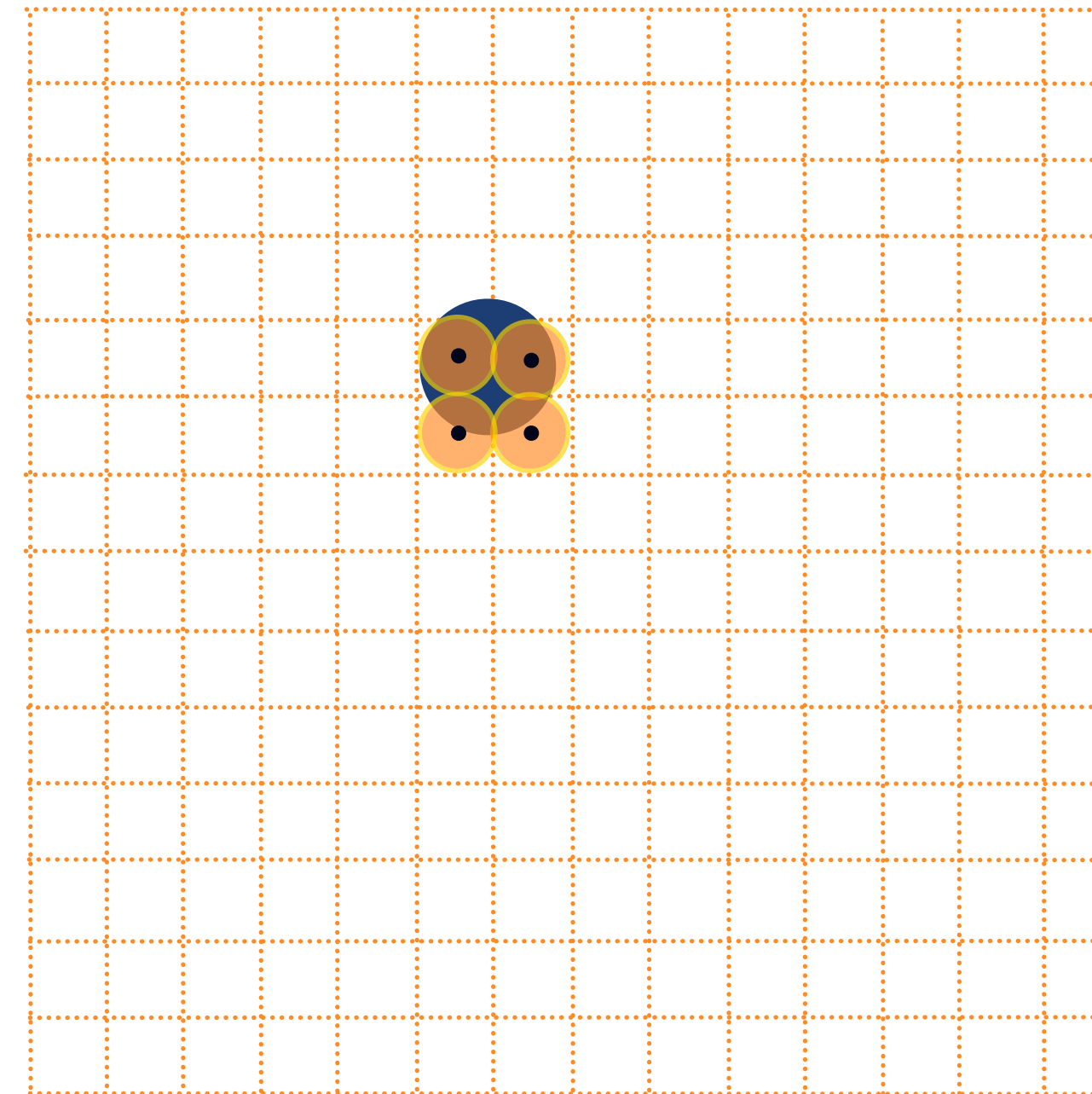
- Draw $[X_0, Y_0]$



Point Drawing



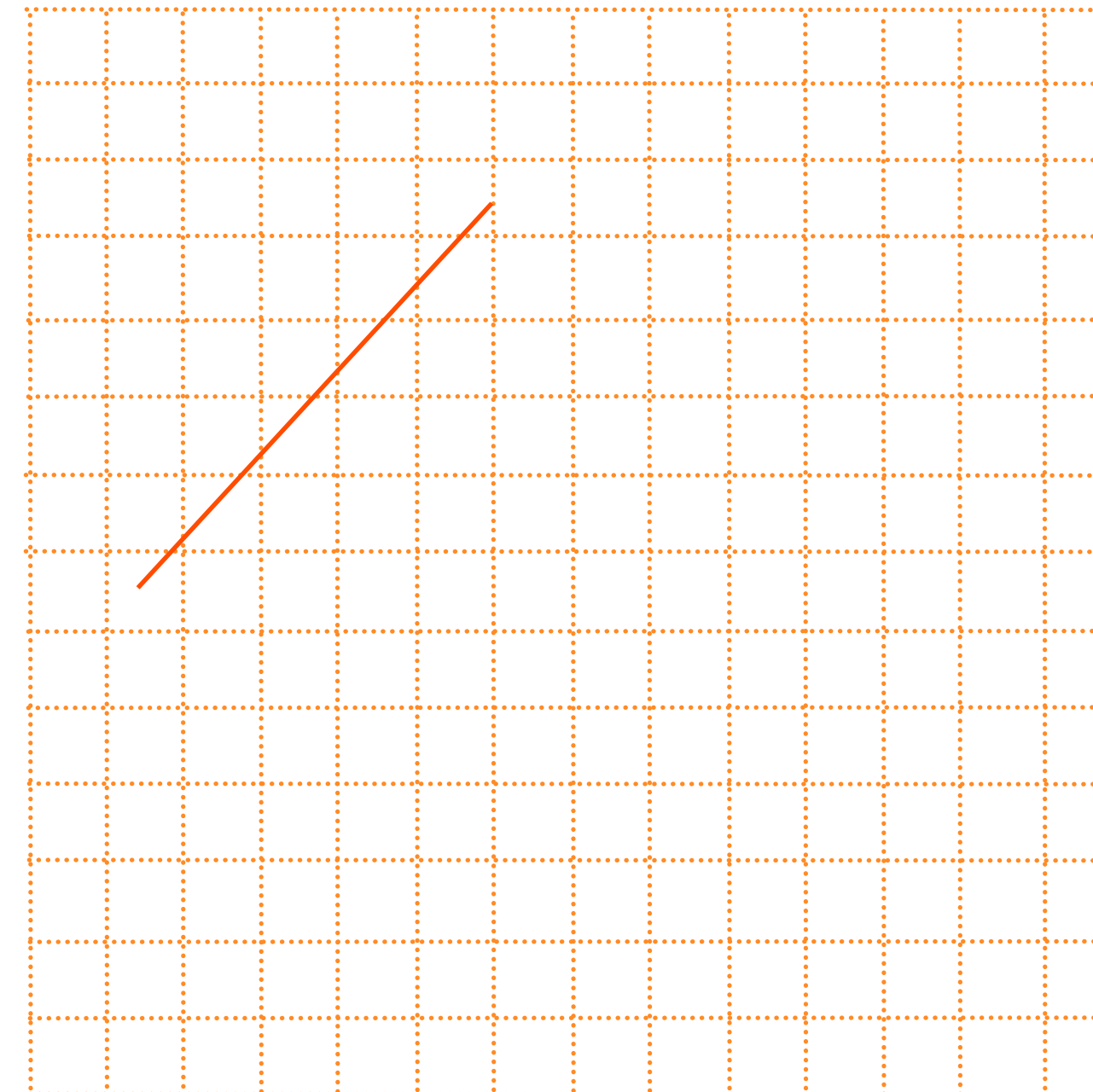
- Draw $[X_0, Y_0]$ of a certain size





Line Drawing

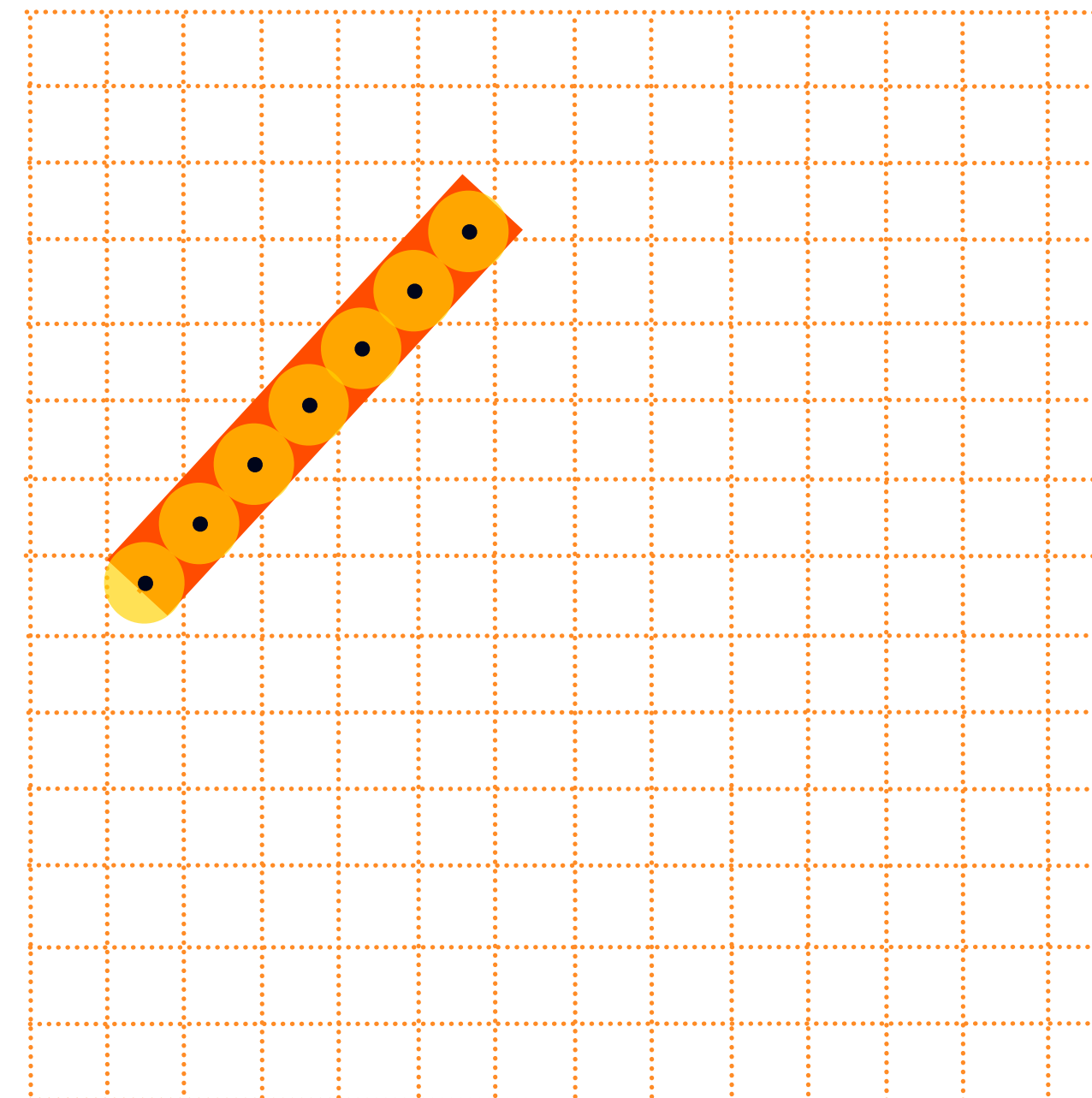
- Draw a line
 - ➔ Start at $[X_0, Y_0]$
 - ➔ End at $[X_1, Y_1]$





Line Drawing

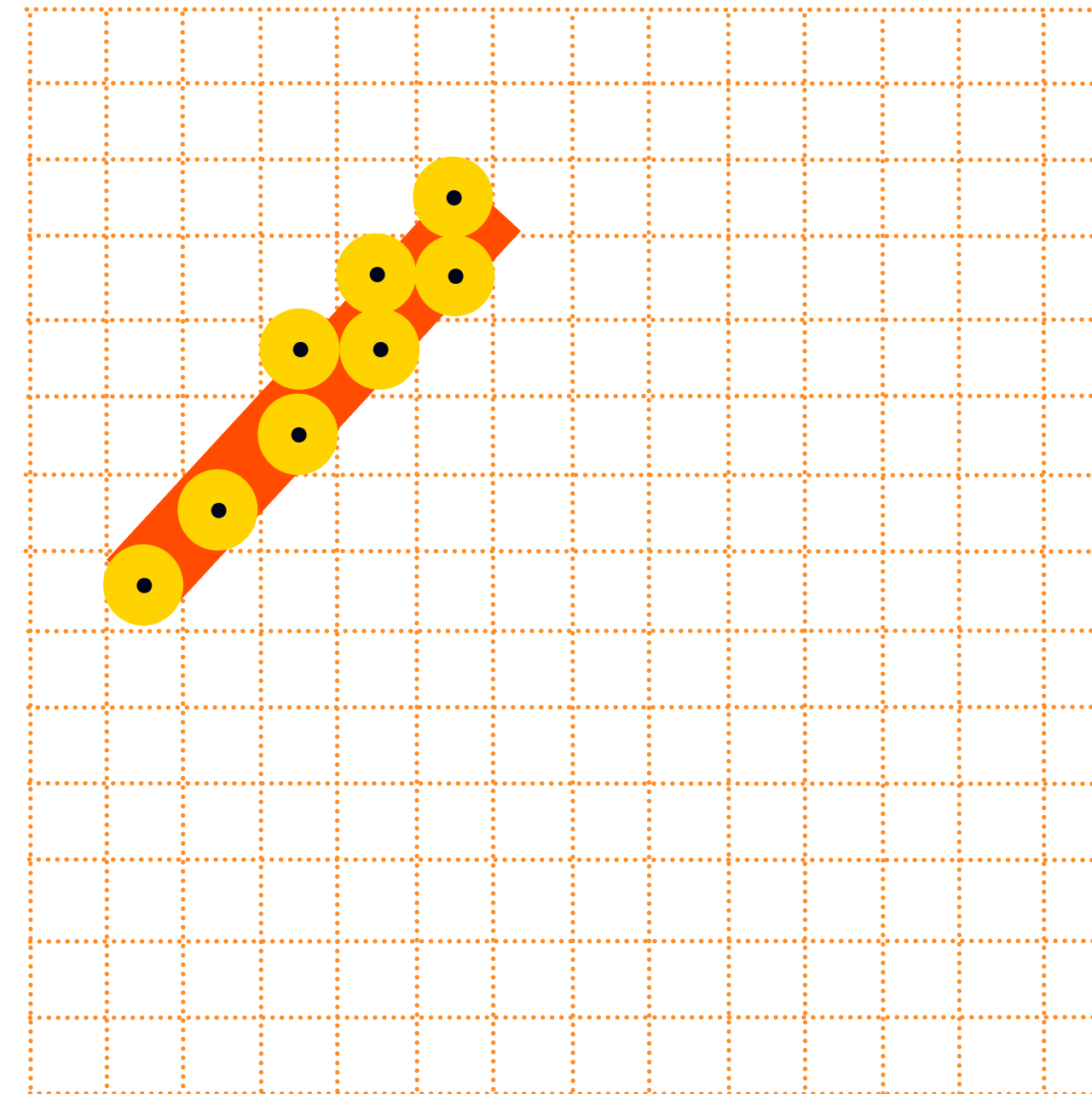
- Draw a line
 - ➔ Start at $[X_0, Y_0]$
 - ➔ End at $[X_1, Y_1]$



Drawing



- Draw a line
 - ➔ Start at $[X_0, Y_0]$
 - ➔ End at $[X_1, Y_1]$

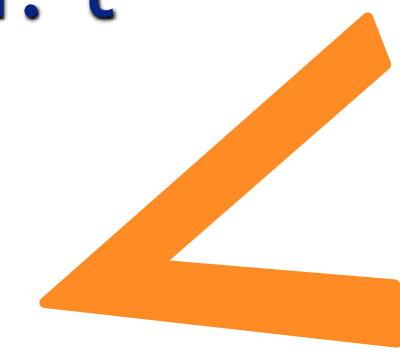




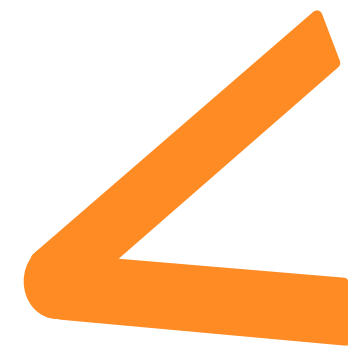
Drawing

- Draw line: $[X_0, Y_0]$ TO $[X_1, Y_1]$

➡ width: t



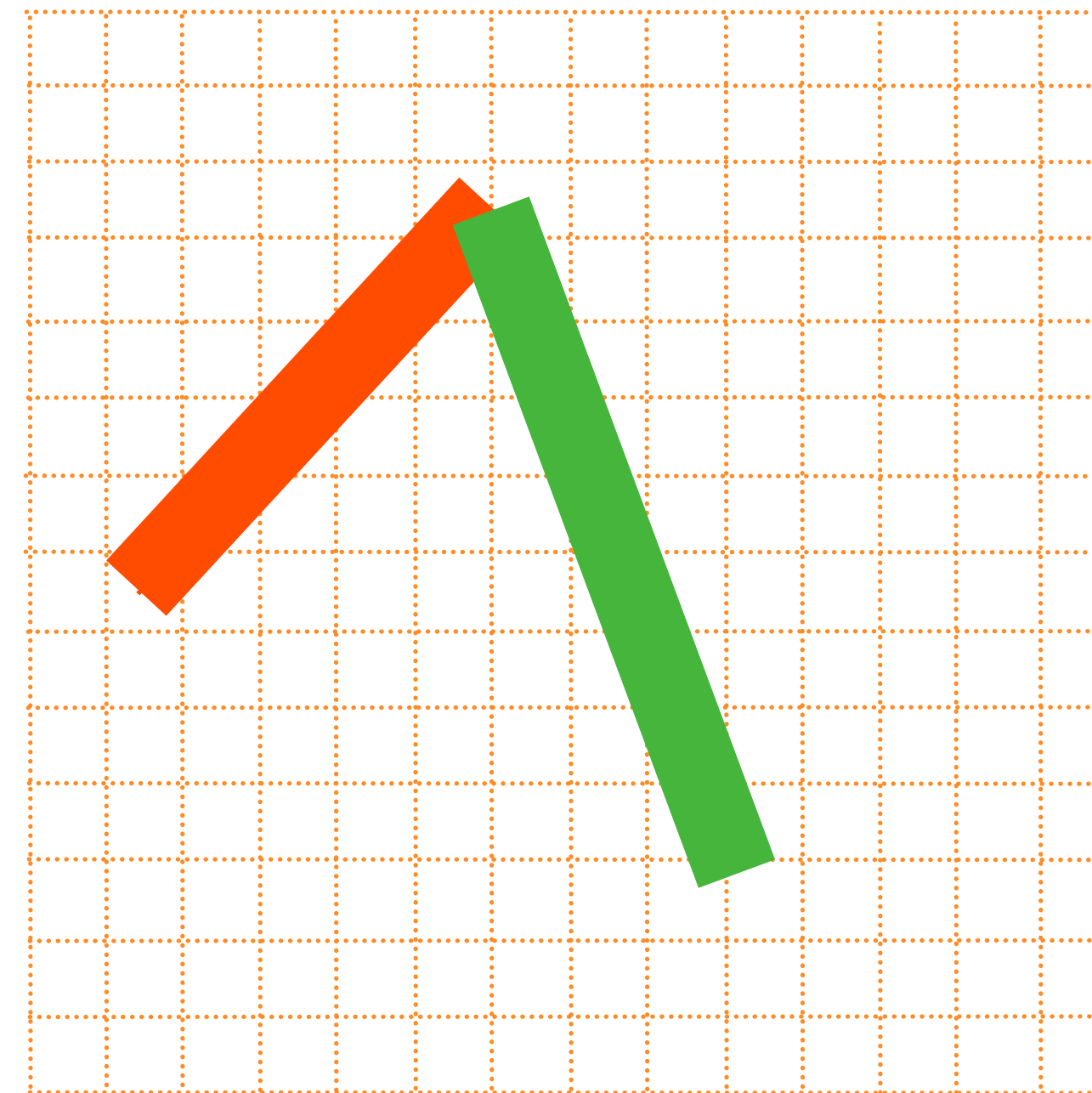
mitered



rounded



bevelled

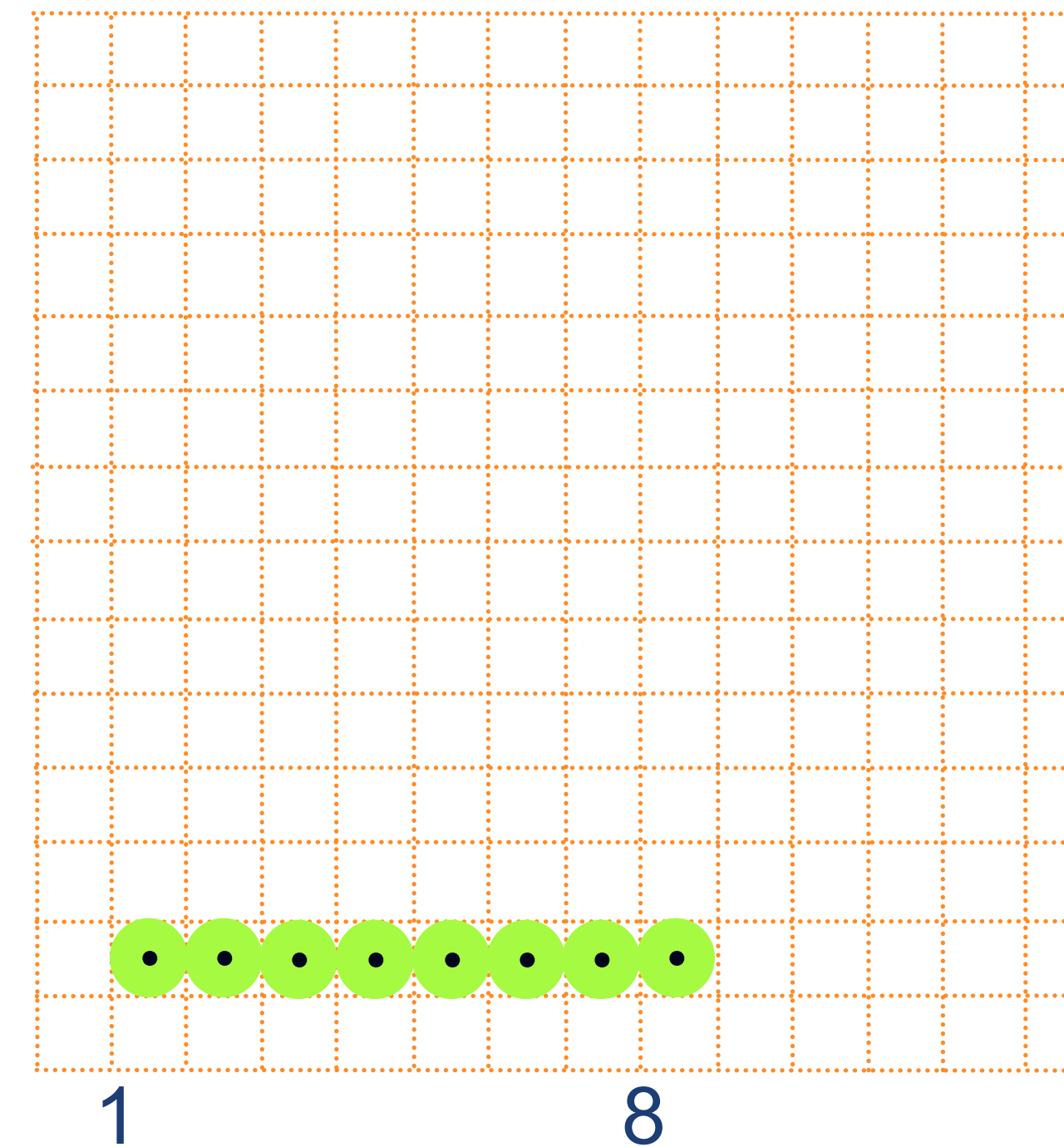


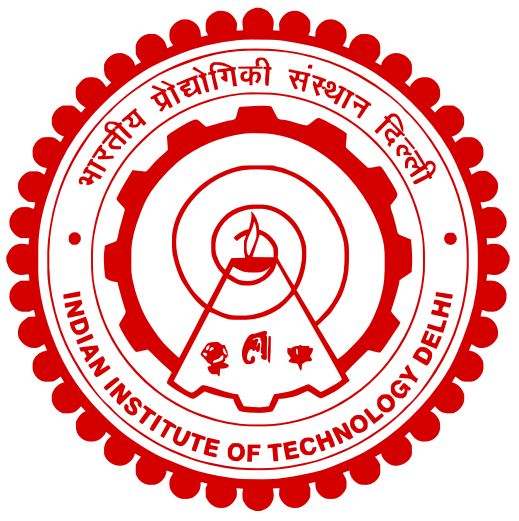
Endcap?



Drawing

- Draw line $[1, 1]$ TO $[8, 1]$
 - ➔ 7 pixels long
 - ➔ eliminate last pixel





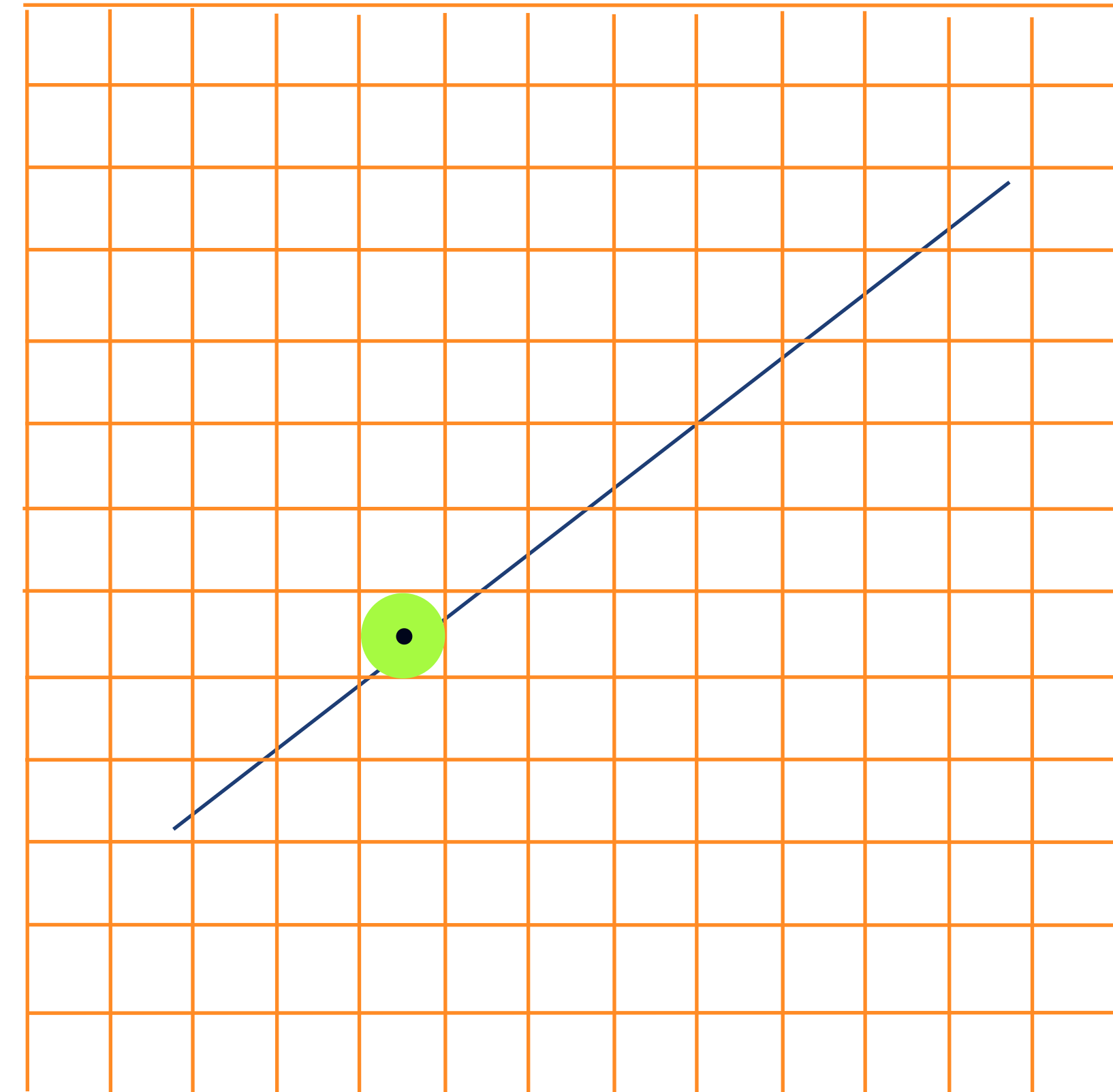
Line Drawing

- Which pixels to draw
 - ➔ How thick is the line
 - ➔ How long is the line
 - ➔ How are the ends drawn
 - ➔ Do you allow overdraw
 - ➔ Which direction is the line drawn in
- Find color and Z



Line Drawing Simplified

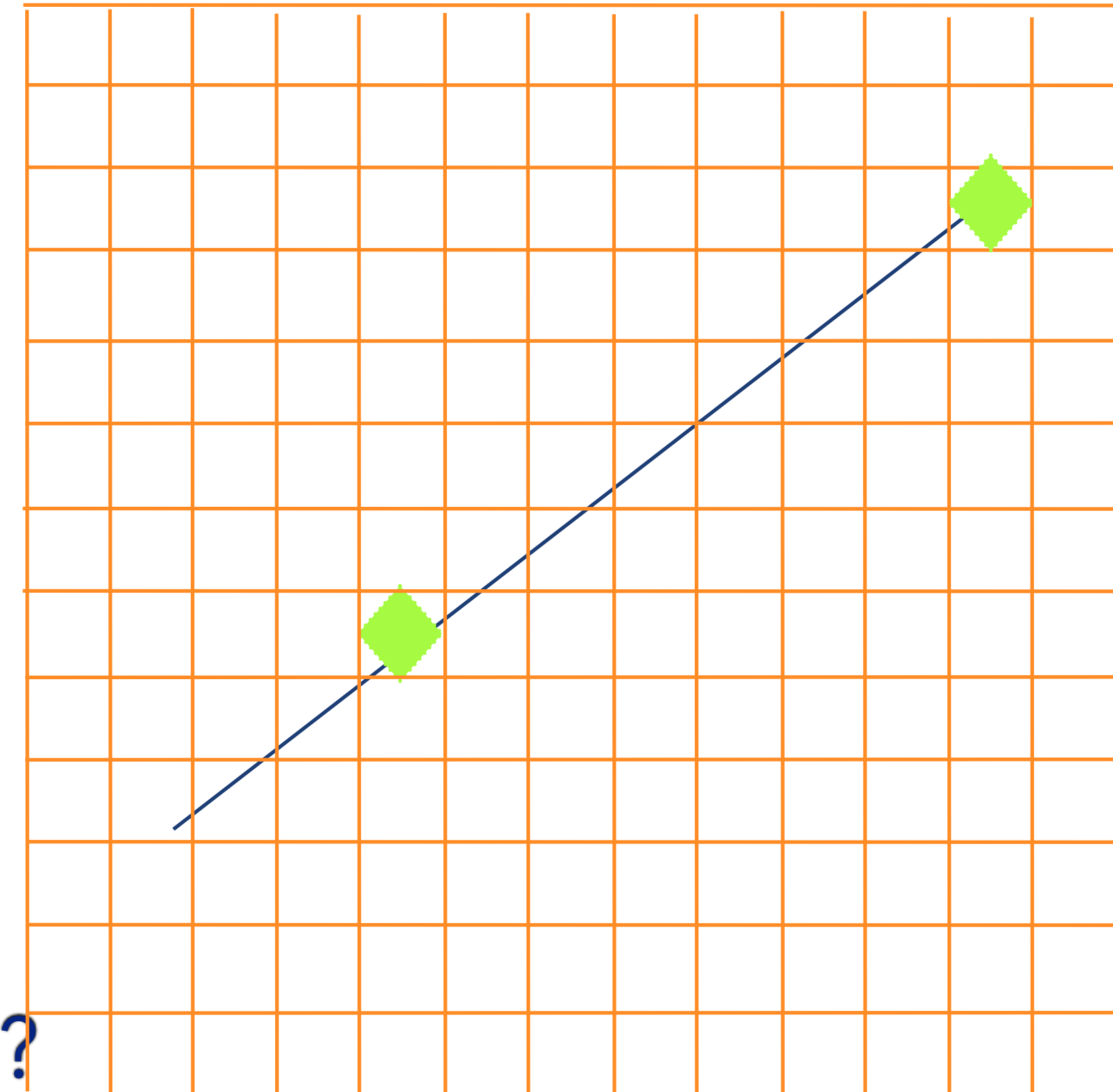
- Draw pixel if
 - ➔ Line is close to it
 - ➔ ie, lies in the circle
 - ➔ Or, some "region of influence"

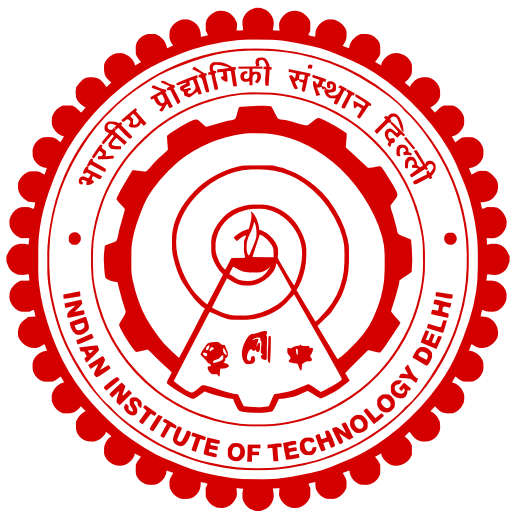




Line Drawing Simplified

- Draw pixel if
 - ➔ Line is close to it
 - ➔ ie, lies in the circle
 - ➔ Or, some "region of influence"
- Draw pixel if:
 - ➔ Line exits diamond
 - ➔ What if line does not exit the diamond?
 - ➔ What if line straddles or ends at a diamond edge





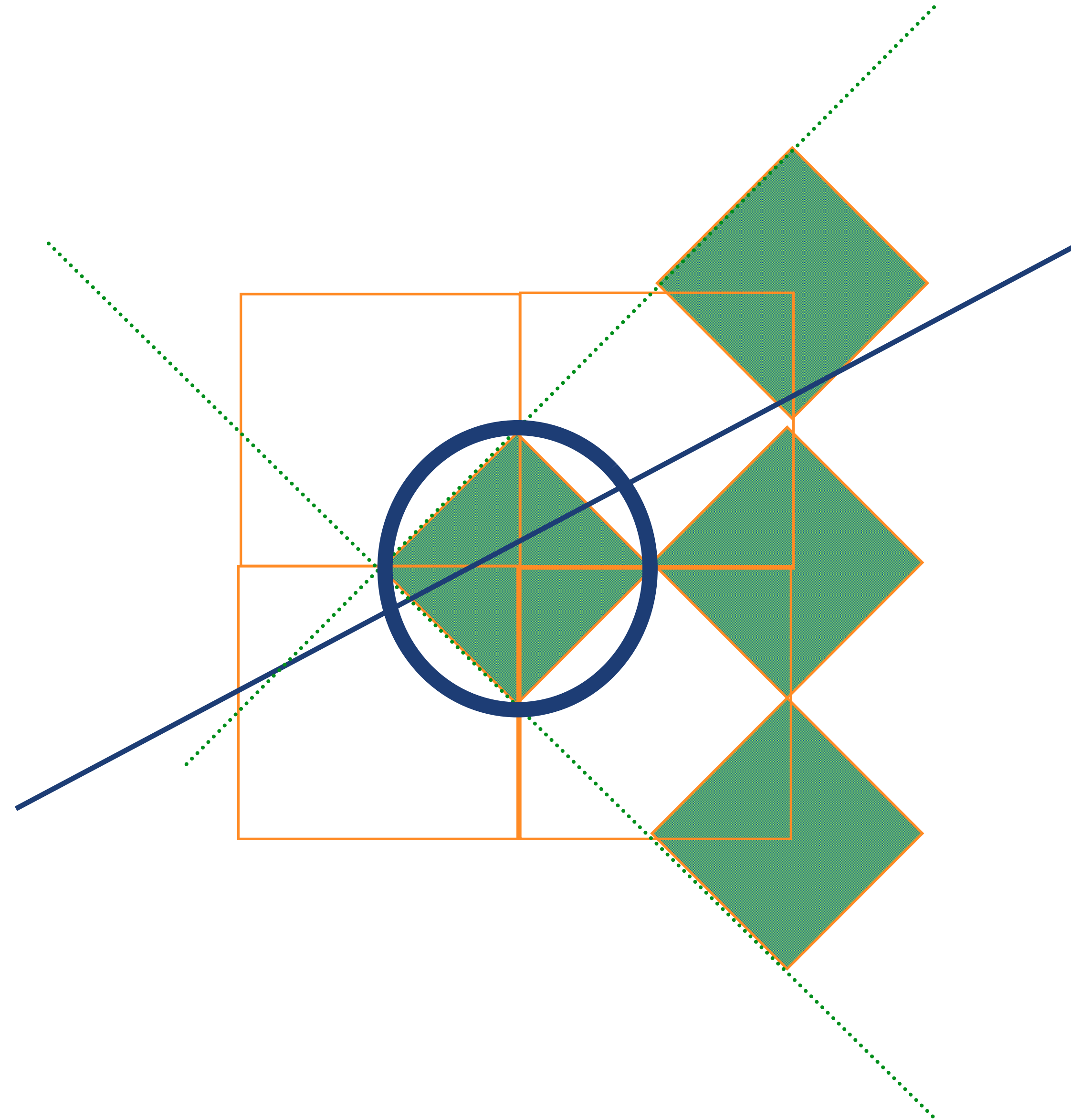
Efficient Line Drawing

- Proceed in the direction of smaller slope
- DDA: Digital Difference Analyzer
 - ➔ $Y = mX + i$
 - ➔ $m(x+1) + i, m(x+2) + i \dots$
 - ➔ Successively add m to y and round it
- Bresenham's midpoint algorithm
 - ➔ Which next pixels are the closest to the line
 - ➔ $aX + bY + c = 0$
 - ➔ Useful for curves as well



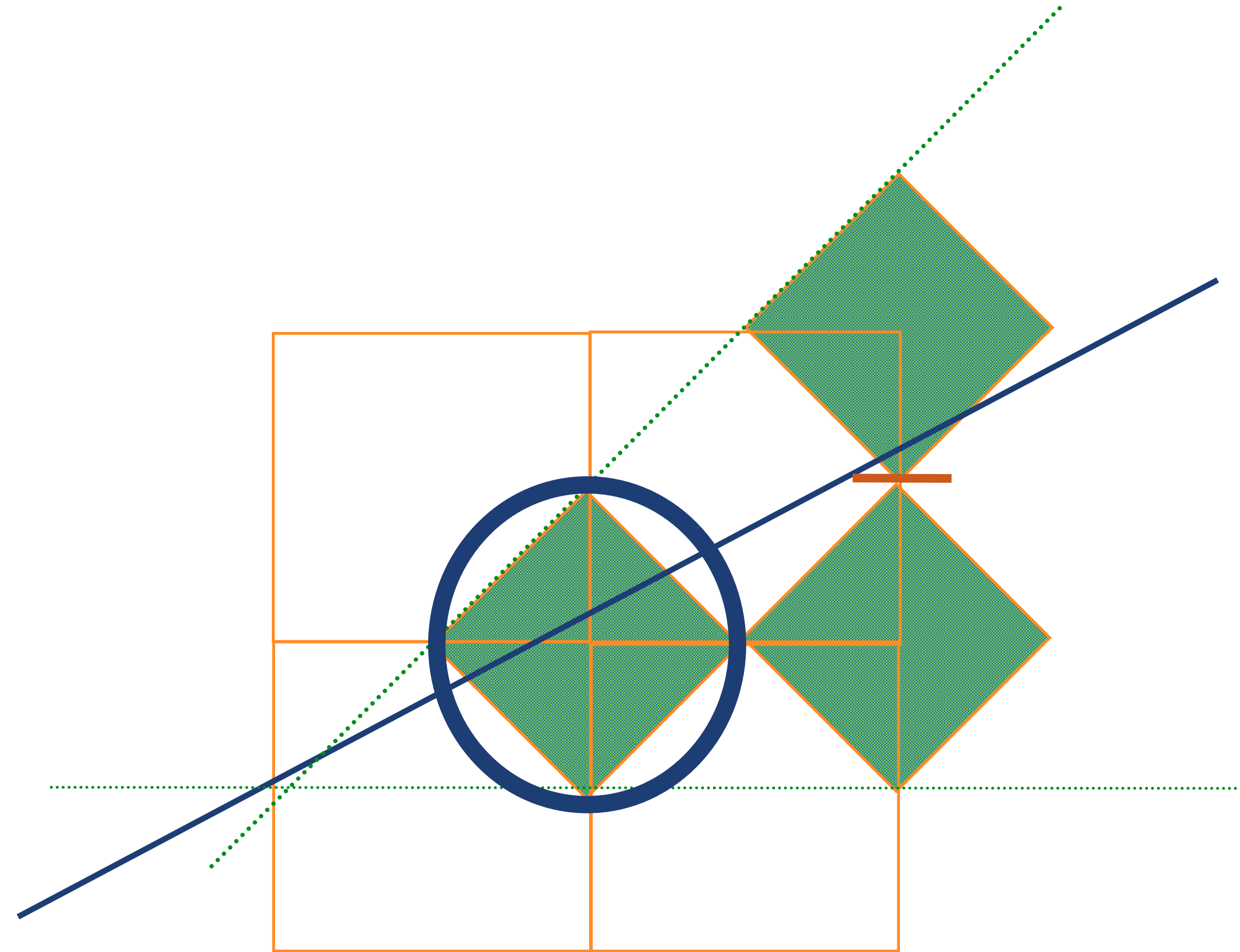
Bresenham's Algorithm

- If slope in $[-1:1]$
➔ 3 possibilities



Bresenham's Algorithm

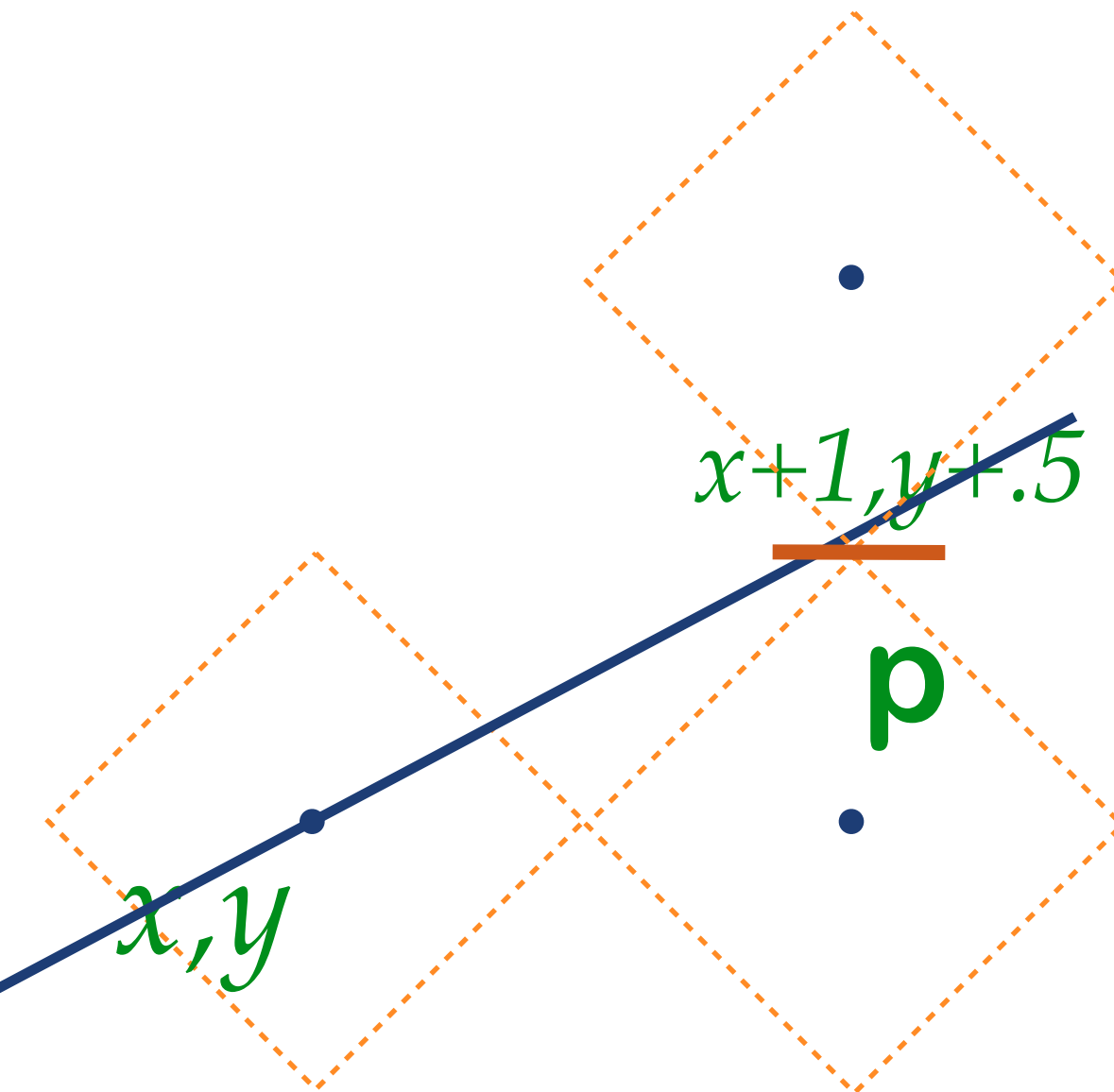
- If slope in $[-1:1]$
 - ➔ 3 possibilities
- If slope in $[0:1]$?
- On which side of the next mid-point does the line lie?





Locate Mid-point

- $Y = mX + i$
- $Y^+ = m(X+1) + i$
 - ⇒ $> y+.5$?
- Consider:
 - ⇒ $F(x,y) = 0$
 - ⇒ ie, $aX+bY+c = 0$



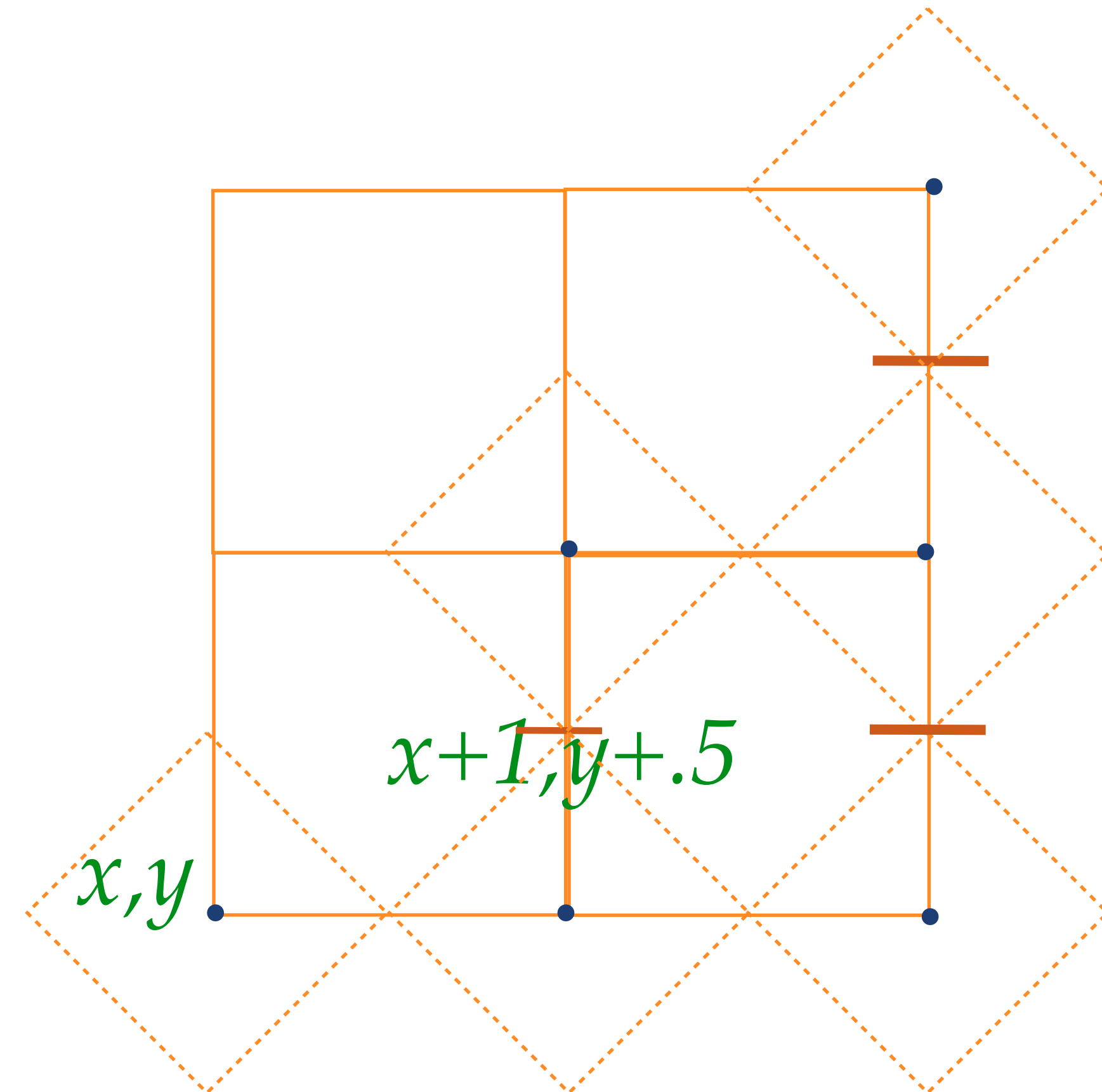
$F(p) = 0 \Rightarrow$ Curve intersects p
 $F(p) < 0 \Rightarrow$ Curve passes below p
 $F(p) > 0 \Rightarrow$ Curve passes above p

If $a > 0$



Locate Mid-point

- $d = a(x+1) + b(y+.5) + c$
- $d_{\text{next}} =$
 - ⇒ $a(x+2) + b(y+.5) + c$
 - ⇒ Or, $a(x+2) + b(y+1.5) + c$
- ie,
 - ⇒ $d + a$
 - ⇒ Or, $d + a + b$
- How do you initialize?





Initialize d

- What if initial vertex coordinates are at pixel centers?

$$\Rightarrow d_{\text{initial}} = a(x_0 + 1) + b(y_0 + .5) + c$$

$$\Rightarrow = ax_0 + by_0 + c + a + b/2$$

$$\Rightarrow = a + b/2$$

⇒ Don't really need the division: Use $2a$, b

- We snap to a fixed-point resolution

