3. Raster Algorithms

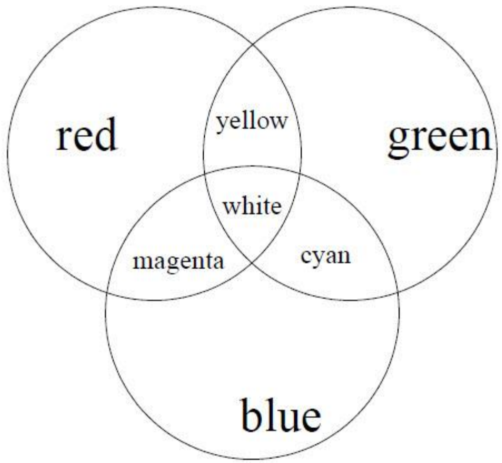
3.1 RGB Color

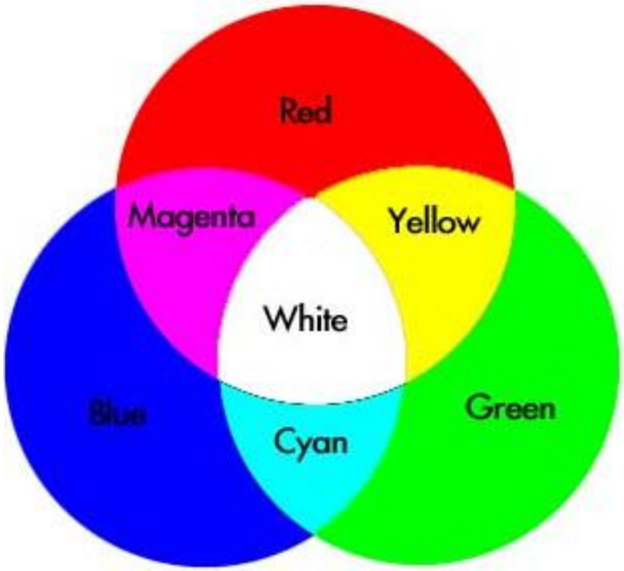
Starts with darkness

- color (light) is displayed by three primary lights: **red, green, blue, in an *additive manner***

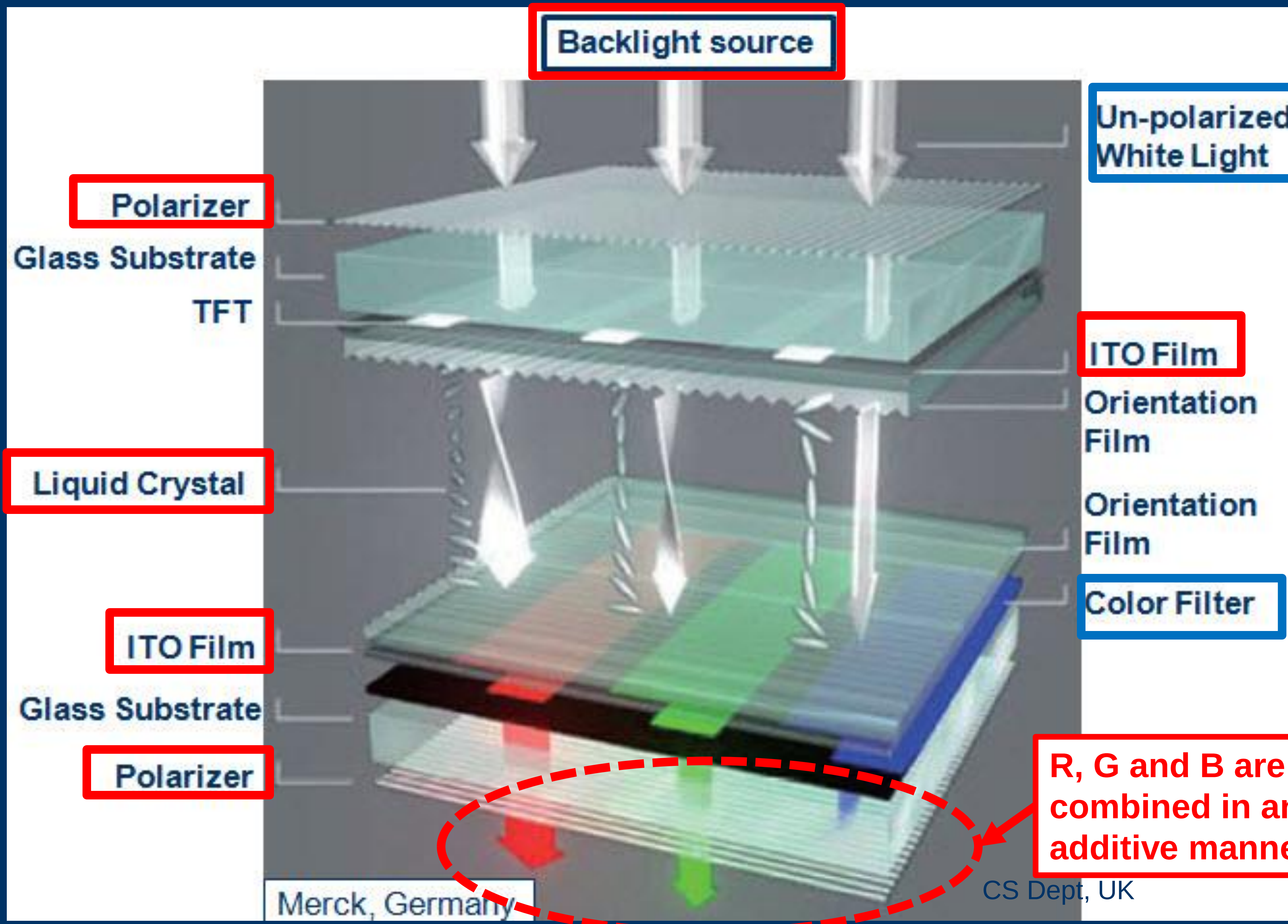
Red + green = yellow

….





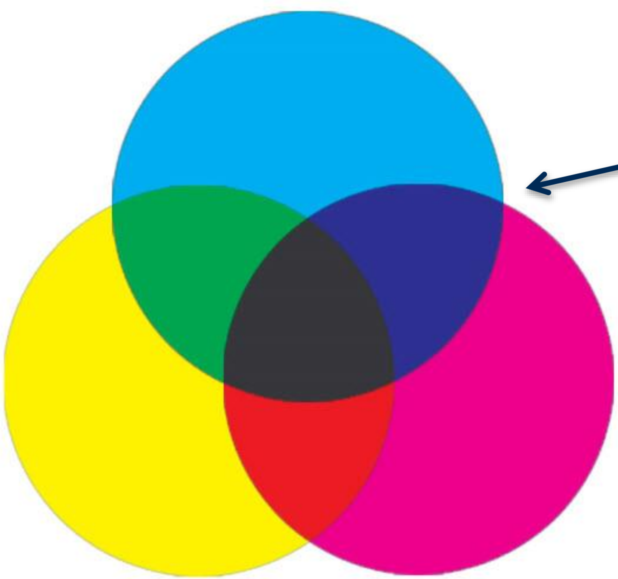
Color Transmissive LCD (single pixel w/ rgb sub)



R, G and B are combined in an additive manner

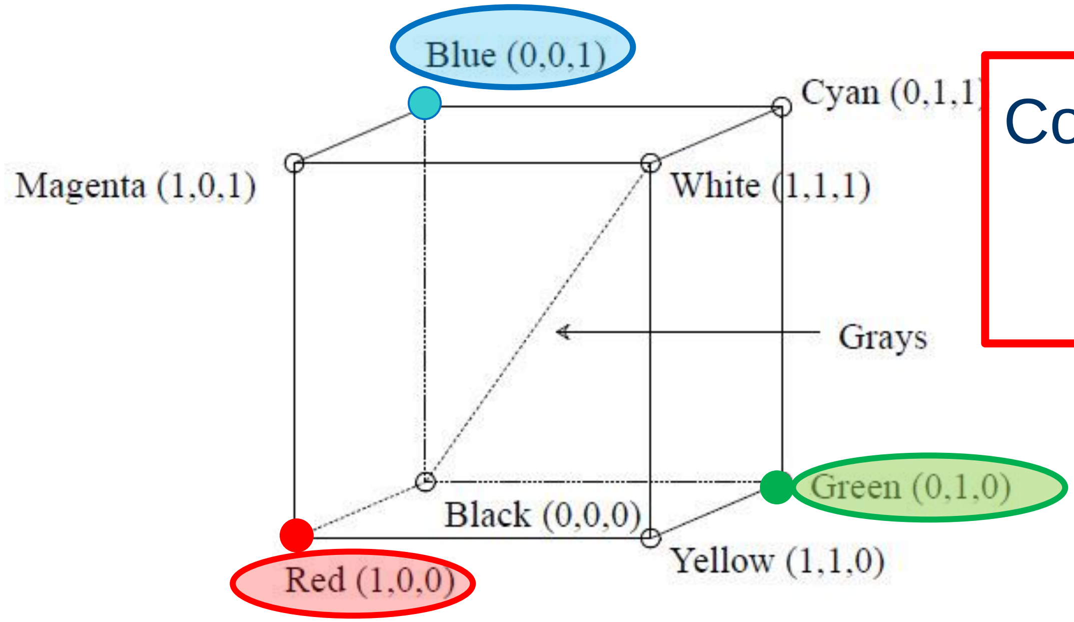
3.1 RGB Color

- on the other hand, paints and crayons are generated using *subtractive color mixing, with primaries:* ***red,*** **yellow, blue**



Starts with light

3.1 RGB Color



Colors are specified as:
Color = (r, g, b)

$$0 \le r, g, b \le 1$$

- Individual contributions of each primary are added together to yield the result
- Most popular for CRT monitors

3.2 The Alpha Channel

- How to partially overwrite the contents of a pixel, such as in *compositing*

- Compositing is the process of combining separate image layers into a single picture

- a multi-bit gray-scale mask (called *alpha*) *is* maintained as a fourth *alpha channel in addition* to (r, g, b) color channels

3.2 The Alpha Channel

- *alpha channel is used to hold an* ***opacity factor*** (or, the fraction of the pixel covered by an opaque surface) for each pixel

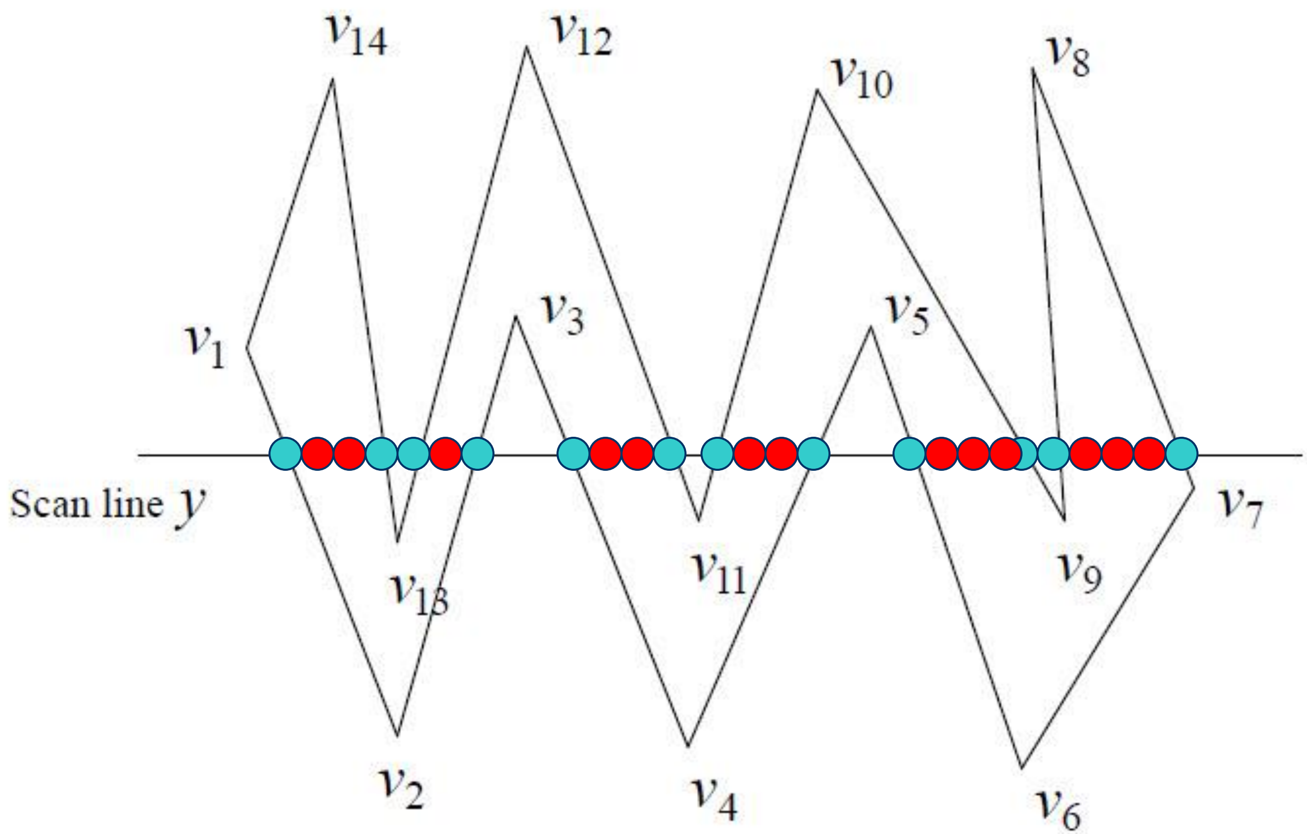
r	g	b	alpha

To composite a foreground color c_f *over background* color c_b *, and the fraction of the pixel* covered by foreground is α , use the formula

$$\mathbf{c} = \alpha \mathbf{c}_f + (1 - \alpha)\mathbf{c}_b$$

Boundary tracking usually is done scanline by scanline.

3.3 Scan Converting Polygons (triangles are special cases)



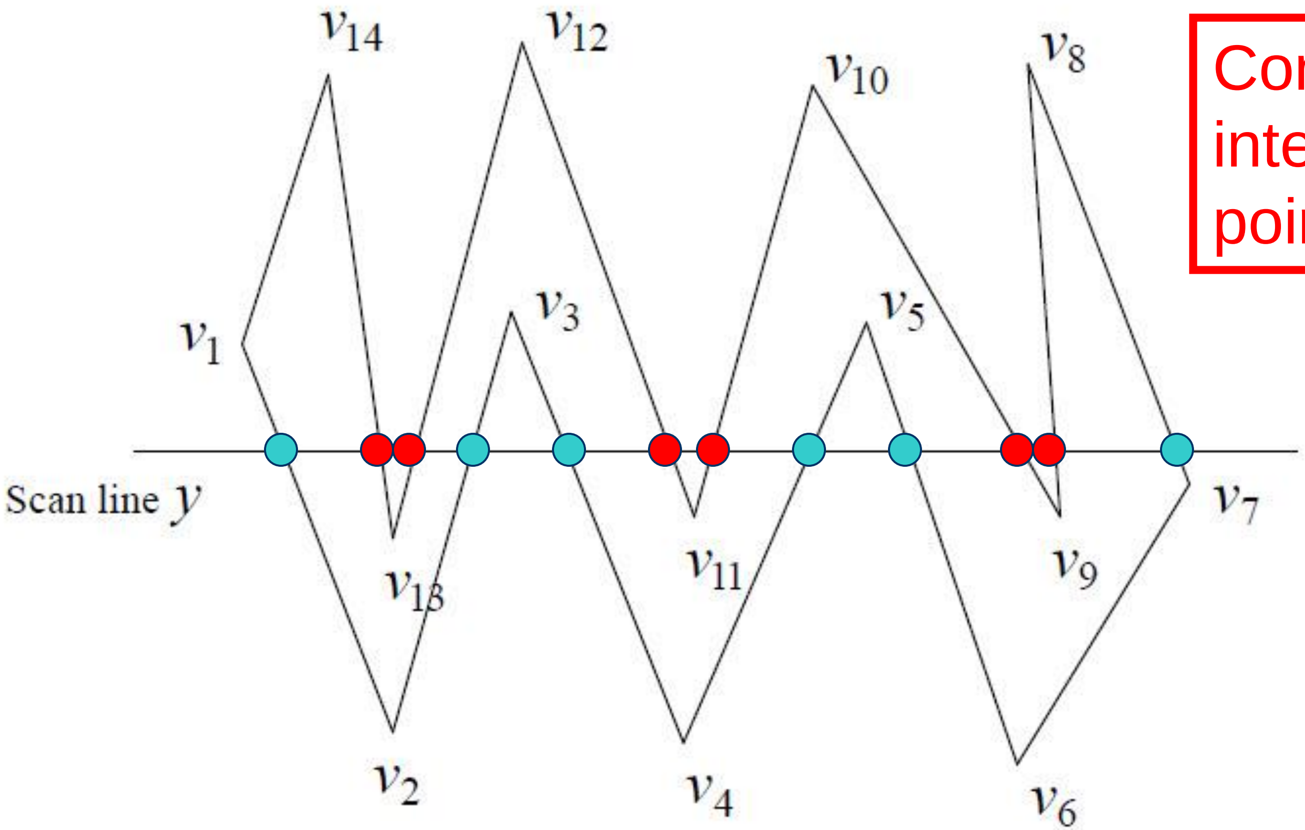
Costly operations; Would like to avoid these opertions. How?

3.3 Scan Converting Polygons

Basic idea:

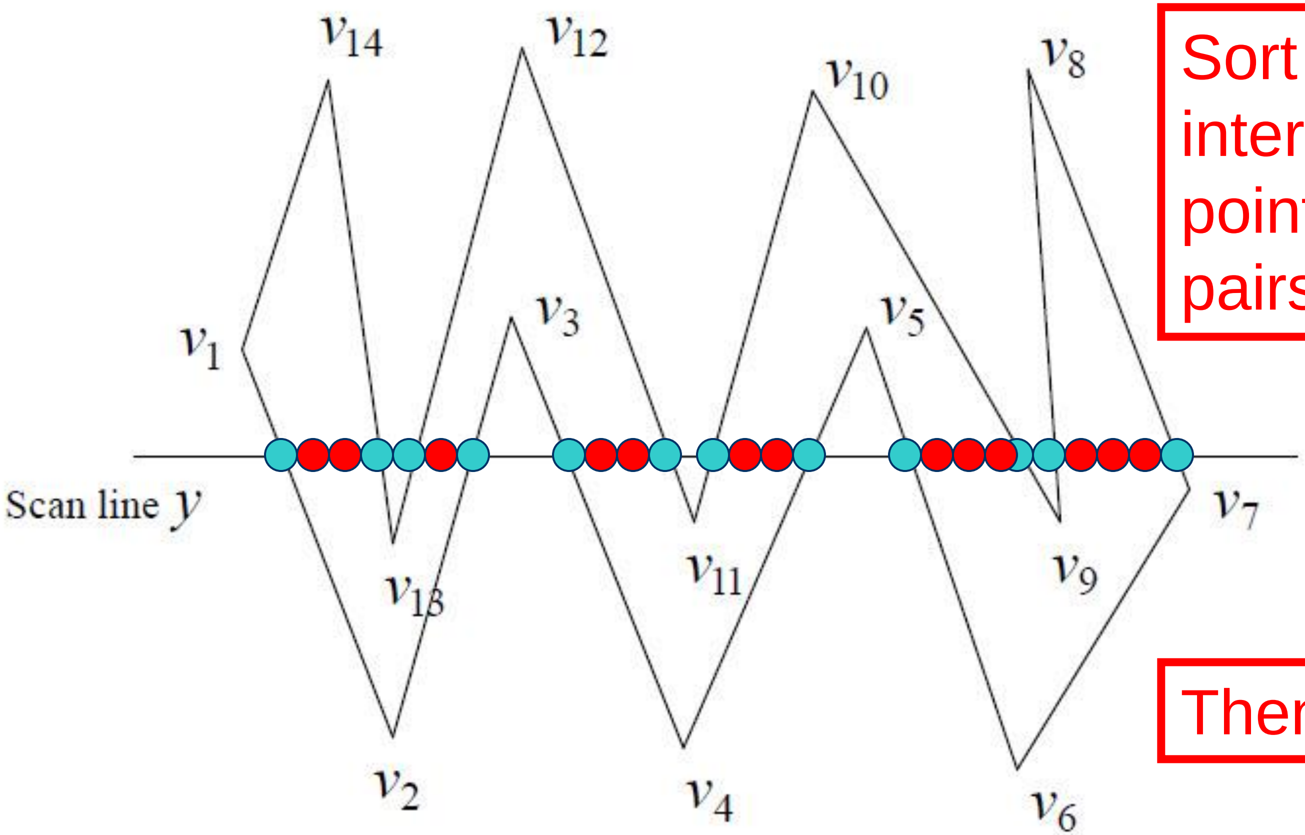
1. Compute *x coordinates of intersection points* of current scan line with all edges
2. Sort intersection points by increasing *x values*
3. Group intersection points by pairs
4. Fill in the pixels between each pair of intersection points on the current scan line

3.3 Scan Converting Polygons



Compute intersection points

3.3 Scan Converting Polygons



Sort intersection points into pairs

Then do fill-in

The goal is to lower the cost of intersection point computation and to avoid the cost of sorting.

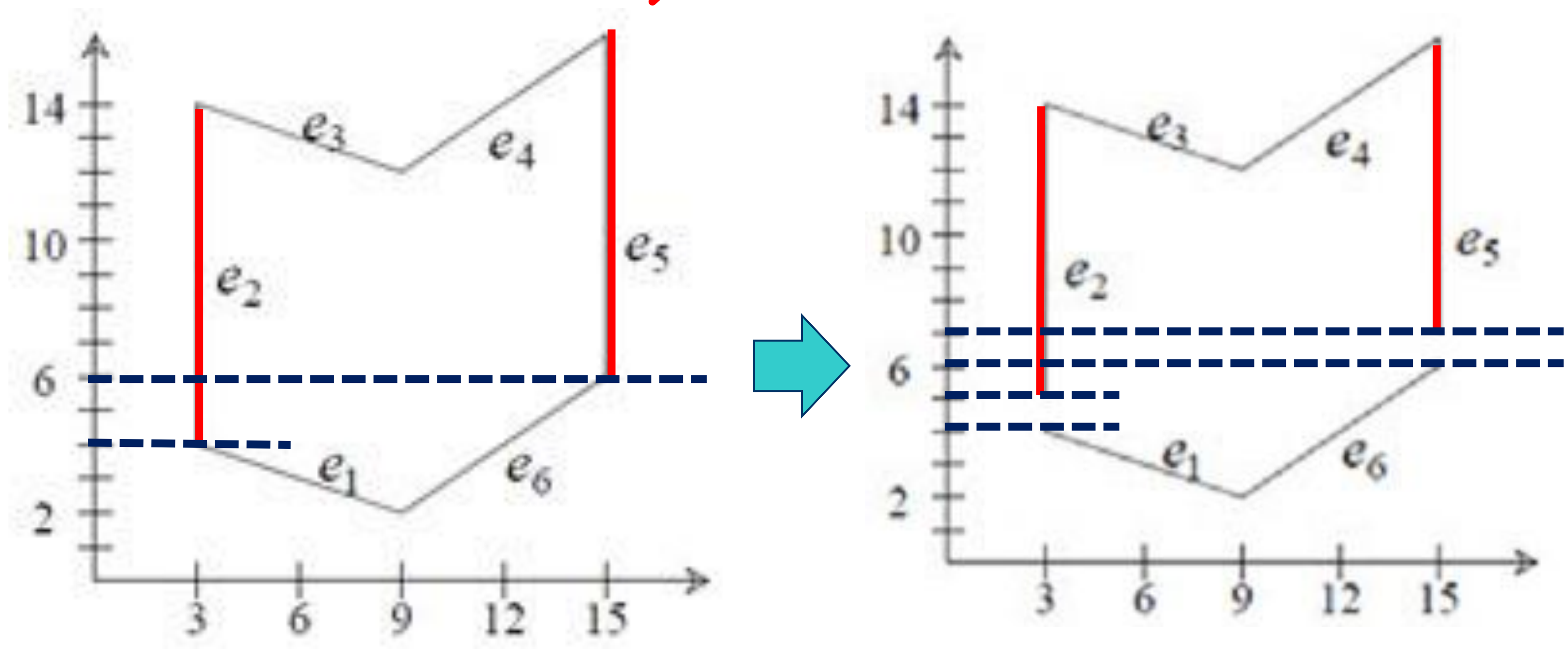
3.3 Scan Converting Polygons

Need to create a **bucket-sorted edge table (ET)** first:

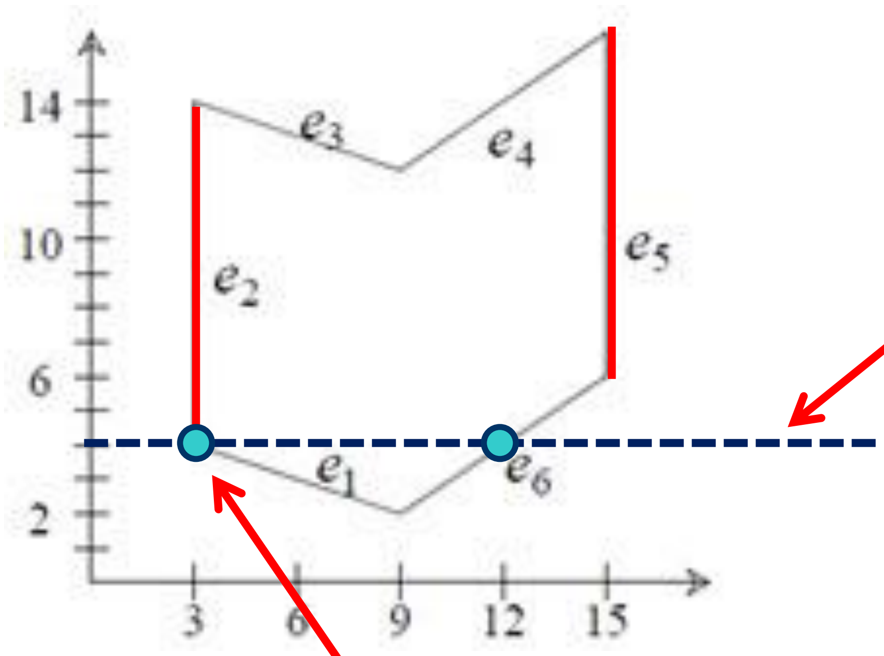
- To determine which edges intersect current scan line
- To efficiently compute intersection points of these edges with the current scan line
- Edges whose lower vertices are not a local minimum need to be shortened by 1 in y-direction

3.3 Scan Convert... Polygons

Edges whose lower vertices are not local minima nor local maxima have to be shortened by one unit in y direction



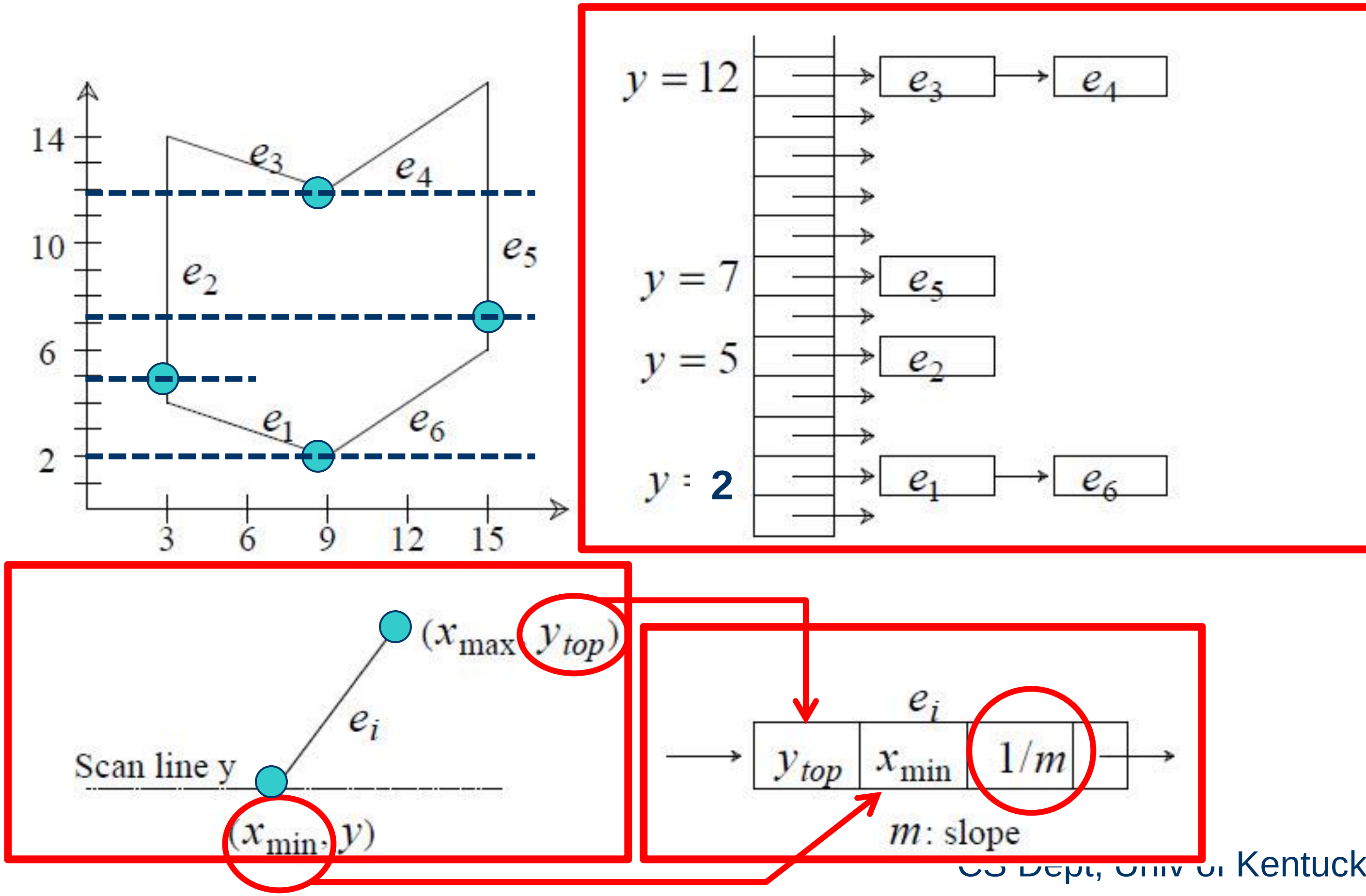
3.3 Scan Converting Polygons



Why?
For this scan line, three intersection points will be reported, not an even number.

This vertex will be reported twice

3.3 Scan Converting Polygons



In sorted order

3.3 Scan Converting Polygons

Also need to maintain an **active-edge list (AEL)**

Purpose: keep track of the edges the current scan line intersects

How: when we move to a new scan line (bottom to top), new edges intersecting the new scan line are added into the AEL, edges in AEL that are no longer active (not intersected by the new scan line) are deleted

3.3 Scan Converting Polygons

Example:

In the previous example, when $y = 2$,

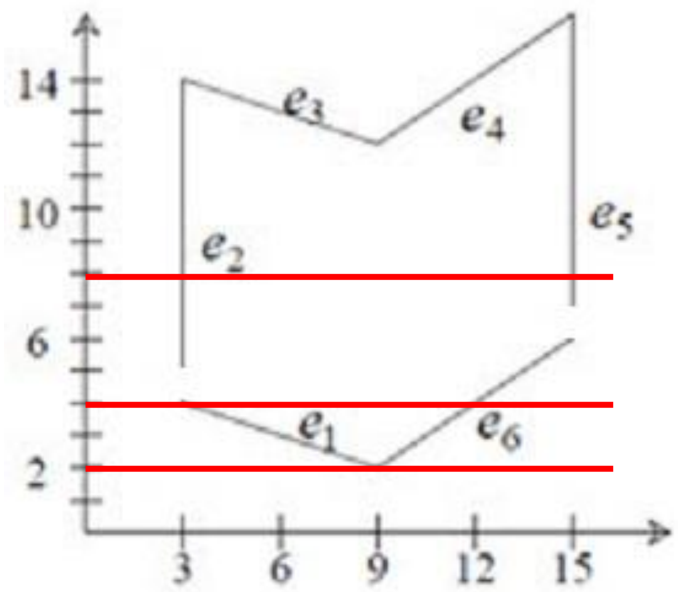
AEL ⟶ $e_1 \rightarrow e_6$

when $y = 4$,

AEL ⟶ $e_1 \rightarrow e_6$

when $y = 8$,

AEL ⟶ $e_2 \rightarrow e_5$



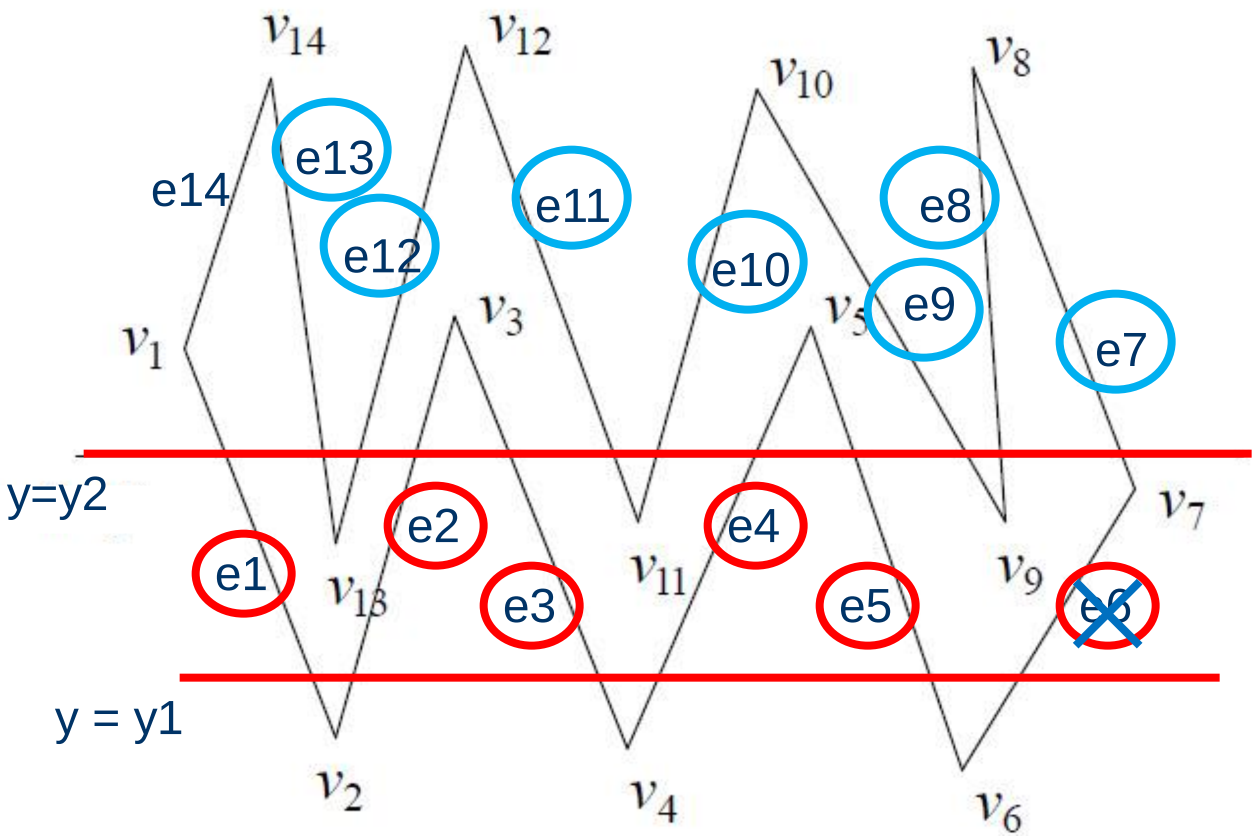
3.3 Scan Converting Polygons

When y=y1

AEL →

When y=y2

AEL →



3.3 Scan Converting Polygons

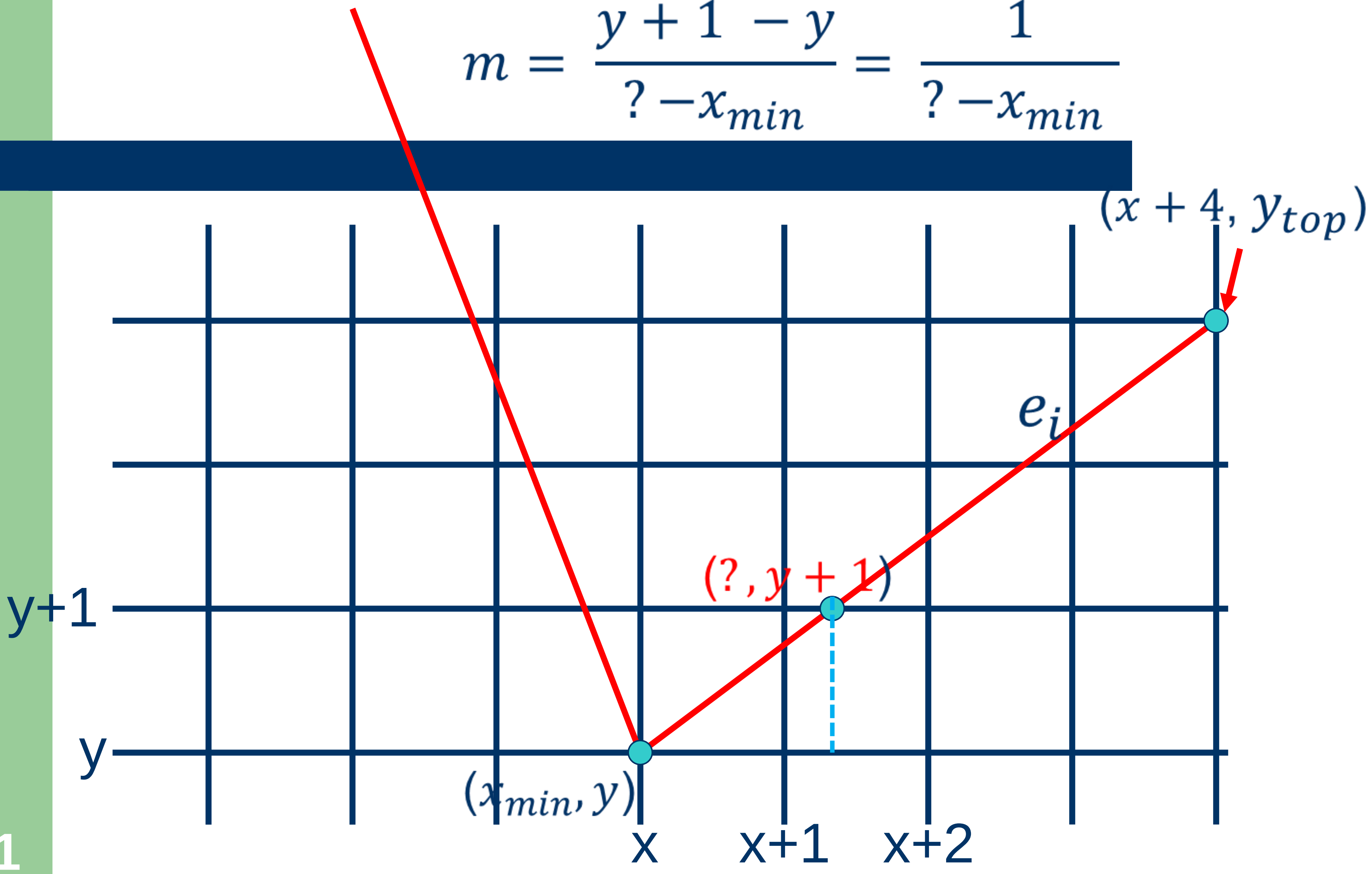
Algorithms:

- Set *y to the y-coordinate of the first nonempty* bucket
- Set AEL to empty
- **Repeat until the AEL and ET are both empty**
  - **Merge** edges from ET bucket *y with* edges in AEL, maintaining AEL sort order on *x and on 1/m*
  - **Fill in** pixels on scan line *y bounded by* pairs of *x-coordinates from edges in* AEL
  - **Remove** from AEL those edges for which $y = y_{top}$
  - For each edge remaining in AEL, **replace** *x* with *x + 1/m*
  - **Increment** *y* by *1, to the coordinate of* the next scan line

The End

$$m = \text{Slope of } e_i = \frac{y_{top} - y}{x + 4 - x_{min}}$$

$$m = \frac{y + 1 - y}{? - x_{min}} = \frac{1}{? - x_{min}}$$



$$m = \frac{y + 1 - y}{? - x_{min}} = \frac{1}{? - x_{min}}$$

$$? = x_{min} + \frac{1}{m}$$