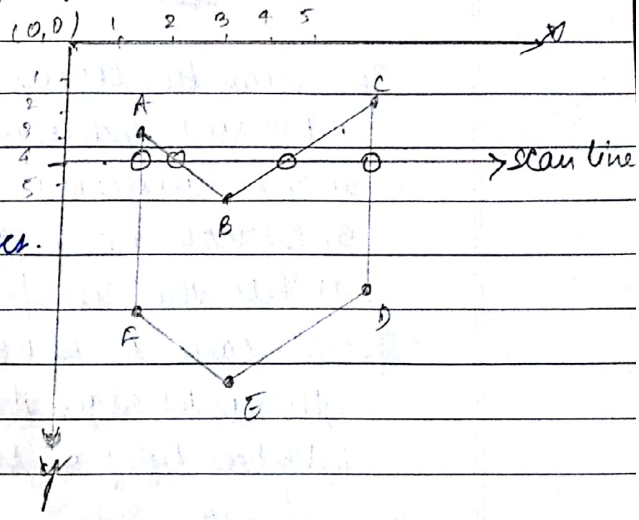
lecture-16
mau

classmate
Date 05/03/18
Page

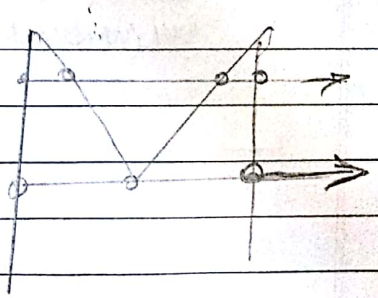
Scan line fill algorithm

- Solid filling of polygonal areas
- For each scan line crossing a polygon, the area-fill algorithm locates the intersection points of the scan line with the polygon edges.
- These intersection points are then sorted from left to right, and the corresponding frame-buffer pos^{n}s b/w each intersection pair are set to the specified fill colour.



- Scan line intersection with polygon vertices needs special handling
  - intersects two polygon edges.
- The corresponding edges either same side of the scan line or opposite side of it.

- Identify these vertices by traversing the polygon boundary in clockwise or anti-clockwise manner
  - observing the relative changes in y-direc^{n}
  - y-value increases or decreases monotonically
    → count single
  - otherwise count twice

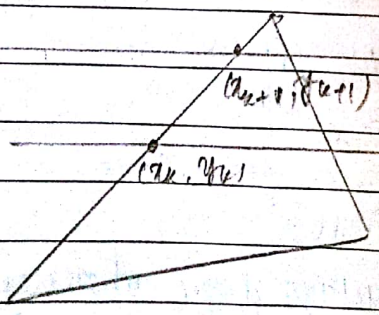


- Graphics algorithms take advantage of the coherence of a scene.
- Properties of one part of the scene is somewhat related to other part of the scene.
- Involves in incremental calc^{ns}, applied along a scan line or successive scan lines.
- To determine edge intersection, incremental coordinate calc^{n}.

- Slope of the line remains constant from one scan line to the other.
- $m = (y_{k+1} - y_k)/(x_{k+1} - x_k)$; changes in y-coordinates: $y_{k+1} - y_k = 1$
- x-coordinate can also be determined : $x_{k+1} - x_k = 1/m$

- The x_k value of k^{th} scan line of slope m $\Rightarrow$ $x_k = x_0 + \frac{k}{m}$
- The incremental x value is $1/m$, $m = \frac{\Delta y}{\Delta x}$ $\quad x_{k+1} = x_k + \frac{\Delta x}{\Delta y}$

- The scan conversion algorithm works as follows:
  1) Intersect each scan line with all edges.
  2) Sort intersections in x.
  3) Calculate parity of intersections to determine in/out.
  4) Fill the 'in' pixels.
- Special cases to be handled:
  i) Horizontal edges should be excluded.
  ii) Vertices lying on scanlines handled by shortening of edges
- Coherence b/w scan lines tells us that
  - Edges that intersect scanline y are like to intersect y+1
  - X changes predictably from scan line y to y+1 (incremental calculation possible)



Edge table

#	Edge		1/m	y_{min}	x (corresponding to y_{min})	y_{max}
0	A(2,7)	B(4,12)	2/5	7	2	12
1	B(4,12)	C(8,15)	4/3	12	4	15
2	C(8,15)	D(16,9)	-8/6	9	16	15
3	D(16,9)	E(11,5)	5/4	5	11	9
4	E(11,5)	F(8,7)	-3/2	5	11	7
5	F(8,7)	G(5,5)	3/2	5	5	7
6	G(5,5)	A(2,7)	-3/2	5	5	7

Edge number 0

#	Edge		1/m	y_{min}	x	y_{max}
0	A(2,7)	B(4,12)	2/5 = 0.4	7	2	11

Scan line	x-intersection
y = 7	2
y = 8	2 + 0.4 = 2.4 ~ 2
y = 9	2.4 + 0.4 = 2.8 ~ 3
y = 10	2.8 + 0.4 = 3.2 ~ 3
y = 11	3.2 + 0.4 = 3.6 ~ 4

Scan line fill algo: AEL (active edge list)

Process the scan lines from top to bottom to construct active Edge Table during scan conversion.

Maintain one active edge list for the current scan line.

When the current scan line reaches the lower/upper endpoint of an edge it becomes active.

When the current scan line moves above the upper/below the lower endpoint the edge becomes inactive.

Use incrative coherence calculations to obtain edge intersections quickly.

AEL is a linked list of active edges on the current scanline y