CS 112 - Polygon Scan Conversion

Polygon Classification

- Convex
  - All interior angles are less than 180 degrees
- Concave
  - Interior angles can be greater than 180 degrees
- Degenerate polygons
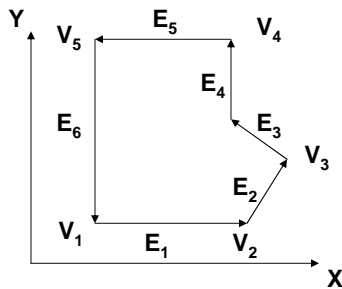 - If all vertices are collinear

Identifying Concave Polygons

- Using direction of cross products of adjacent edges
  - If all same sign, then convex, else concave

$E_1 \times E_2 > 0$

$E_2 \times E_3 > 0$

$E_3 \times E_4 < 0$

$E_4 \times E_5 > 0$

$E_5 \times E_6 > 0$

Aditi Majumder, CS 112
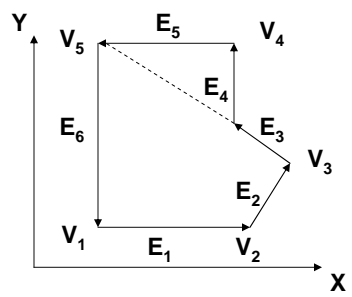
Slide 3

Splitting Concave Polygons

- Assume that the polygon is on XY plane
- Find the cross products of adjacent edges
- If it has a negative z component
  - Split the polygon along the line of the first edge vector of the cross product

Aditi Majumder, CS 112

Slide 4

Identifying Concave Polygons

- Using direction of cross products of adjacent edges
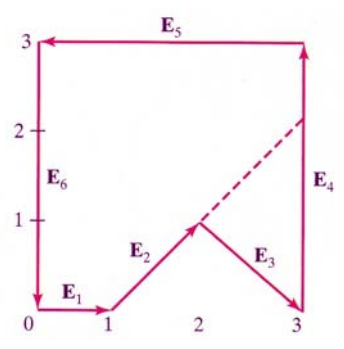 - If all same sign, then convex, else concave

$E_1 \times E_2 > 0$

$E_2 \times E_3 > 0$

$E_3 \times E_4 < 0$

$E_4 \times E_5 > 0$

$E_5 \times E_6 > 0$

Example

$E_1 = (1, 0, 0)$ $E_2 = (1, 1, 0)$

$E_3 = (1, -1, 0)$ $E_4 = (0, 2, 0)$

$E_5 = (-3, 0, 0)$ $E_6 = (0, -2, 0)$

$E_1 \times E_2 = (0, 0, 1)$ $E_2 \times E_3 = (0, 0, -2)$

$E_3 \times E_4 = (0, 0, 2)$ $E_4 \times E_5 = (0, 0, 6)$

$E_5 \times E_6 = (0, 0, 6)$ $E_6 \times E_1 = (0, 0, 2)$
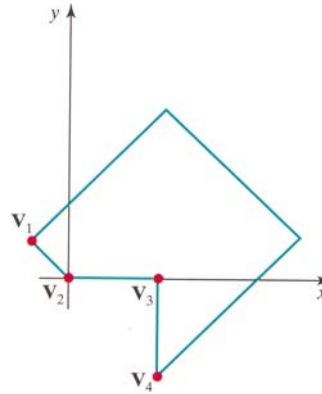
Rotational Method

- For each vertex V_k
  - Translate so that V_k coincides with origin
  - Rotate clockwise so that V_{k+1} lies on the x-axis
  - If V_{k+2} is below the x-axis, then concave
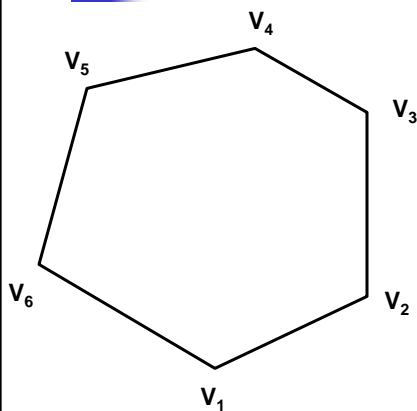 - Split along x-axis to get two polygons
  - Apply inverse transformations
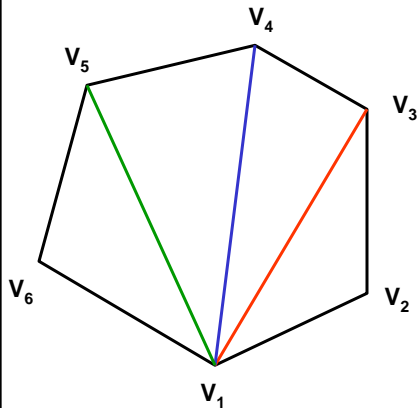
Triangulating a Convex Polygon

- Take any three vertices in order
- Throw away the middle vertex from the list
- Apply to the same procedure
- Until only three vertices remain

Triangulating a Convex Polygon

V_4

V_5

V_3

V_6

V_2

V_1

- $V_1, V_2, V_3, V_4, V_5, V_6$
- V_1, V_2, V_3 – tri 1
- V_1, V_3, V_4, V_5, V_6
- V_1, V_3, V_4 – tri 2
- V_1, V_4, V_5, V_6
- V_1, V_4, V_5 – tri 3
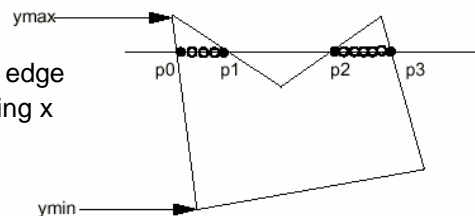- V_1, V_5, V_6 – tri 4

Aditi Majumder, CS 112

Slide 9

Polygon Rasterization

- Works for both convex and concave polygons
- Basic Idea: Intersect scanline with polygon edges and fill between pairs of intersections

For y = ymin to ymax
  1) intersect scanline y with each edge
  2) sort interesections by increasing x [p0,p1,p2,p3]
  3) fill pairwise (p0 - p1, p2 - p3)

ymax

p0   p1   p2   p3

ymin
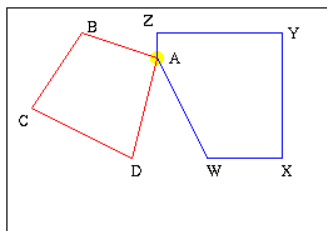
Aditi Majumder, CS 112

Slide 10

5

Filling in – Odd Parity Rule

- Sort the intersections
- Start with even parity, and traverse scanline from left to right
- Whenever an intersection is encountered, flip parity
- If parity is odd, draw the pixel
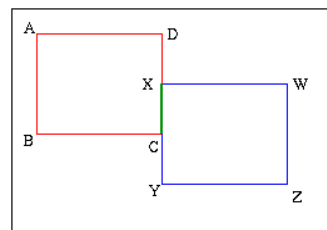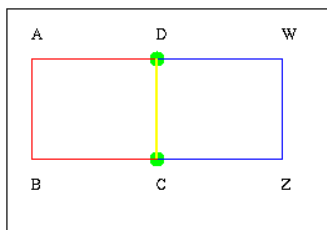- Continue till the end of the scanline

Problem: Shared Boundaries

Case 1: Shared vertices

Case 2: Shared edges

Case 3: Shared vertices and edges

How to solve this?

- Let the pixel be colored in the order the polygons are scan converted
- Rightmost polygon is the dominant one
- Problems
  - Coloring same pixel multiple times
  - Can cause coloring problems depending on what exactly the coloring function does
    - Can create an amalgam of colors

How to solve this?

- Define interior: For a given pair of intersection points (X_i, Y), (X_j, Y)
- Fill ceiling(X_i) to floor(X_j)
  - Will resolve shared boundary problems
- Intersection has an integer X coordinate
  - if X_i is integer, we define it to be interior (fill)
  - if X_j is integer, we define it to be exterior (don't fill)

Problem: Edge endpoint

- Intersection is an edge end point, say: (p0, p1, p2) ??
- (p0,p1,p1,p2), so we can still fill pairwise
- In fact, if we compute the intersection of the scanline with edge e1 and e2 separately, we will get the intersection point p1 twice. Keep both of the p1.
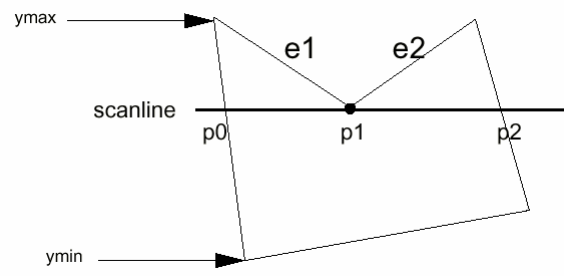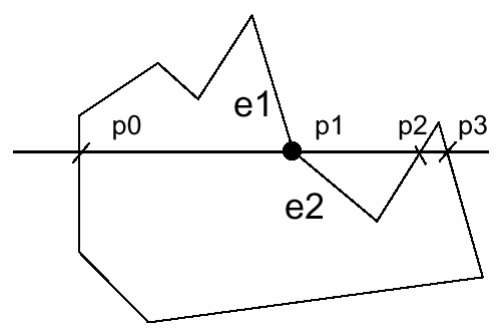

Aditi Majumder

Slide 15

Problem: Edge endpoint

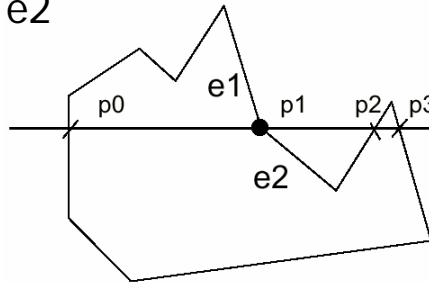- **But what about this case: still** (p0,p1,p1,p2)

Aditi Majumder, CS 112

Slide 16

8

Rule

- Rule:
  - If the intersection is the ymin of the edge's endpoint, count it. Otherwise, don't.
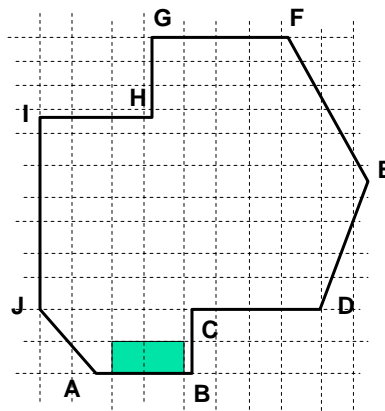- Don't count p1 for e2

Horizontal edges

- Need not be considered
- AB
  - A – intersection of JA
    - Parity becomes odd at A
  - B – intersection of BC
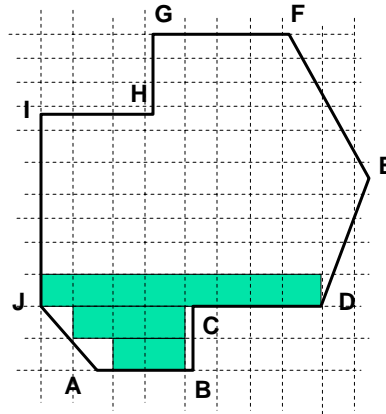   - Parity becomes even at B
  - Hence AB is drawn

Horizontal edges

- **CD**
  - **J – intersection of IJ**
    - Not of JA
    - Parity becomes odd at J
  - **C – no change seen**
    - Parity remains odd
  - **D – intersection of ED**
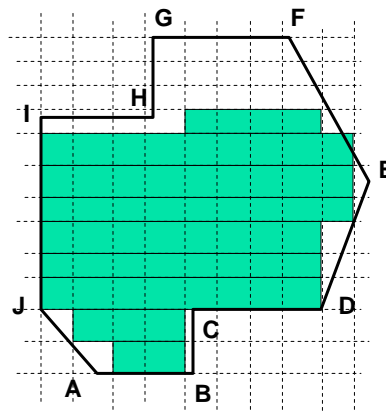   - Parity changes to even
  - **CD is drawn**

Aditi Majumder, CS 112   Slide 19

Horizontal edges

- **IH**
  - **I – no intersection of IJ**
    - Parity remains even
  - **H – intersection of GH**
    - Parity becomes odd
  - **IH is not drawn**
  - **But right span of IH is drawn**

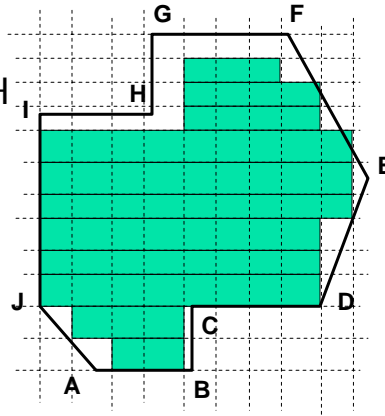Aditi Majumder, CS 112   Slide 20

10

Horizontal edges

- GF
  - G – no intersection of GH
    - Parity remains even
  - F – no intersection of FE
    - Parity becomes even
  - GF is not drawn

Performance Improvement

- Brute force: intersect all the edges with all scanline
- Goal:compute the intersections more efficiently
  - find the ymin and ymax of each edge and intersect the edge only when it crosses the scanline
  - only calculate the intersection of the edge with the first scan line it intersects
  - calculate dx/dy
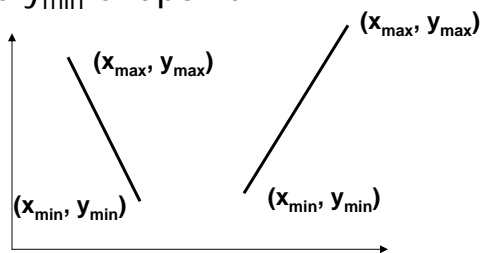 - for each additional scanline, calculate the new intersection as x = x + dx/dy

Data Structure

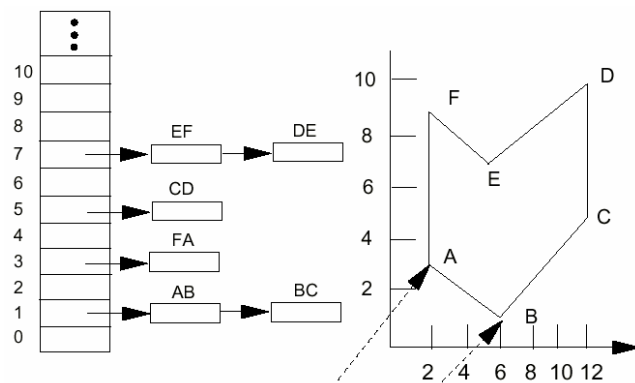- Edge table: Bucket Sort
  - A separate bucket for each scanline
  - Each edge goes to the bucket of its y_{min} scanline
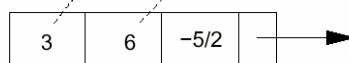 - Within each bucket, edges are sorted by increasing x of the y_{min} endpoint

(x_{max}, y_{max})

(x_{max}, y_{max})

(x_{min}, y_{min})

(x_{min}, y_{min})

Aditi Majumder, CS 112

Slide 23

Edge Table

10		
9		
8	EF	DE
7		
6	CD	
5		
4	FA	
3		
2	AB	BC
1		
0		

10
F
D

8

6
E

4
A
C

2
B

2 4 6 8 10 12

− Edge structure: ymax, xmin, dx/dy, next
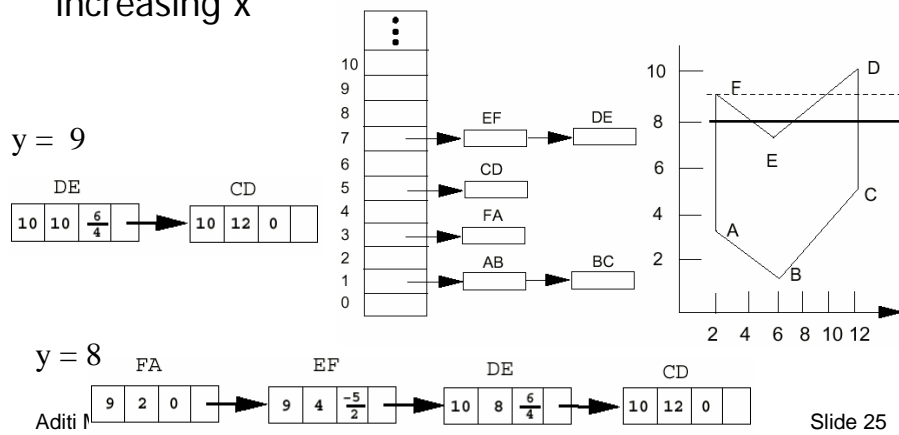
AB:

3	6	−5/2	

Aditi Majum

Slide 24

12

Active Edge Table (AET)

- A list of edges active for current scanline, sorted in increasing x

y = 9

y = 8

Aditi M

Algorithm

Construct the Edge Table (ET);
Active Edge Table (AET) = null;
for y = Ymin to Ymax
    Merge-sort ET[y] into AET by x value
    for each edge in AET
        if edge.ymax = y
           remove edge from AET
    Fill between pairs of x in AET
    for each edge in AET
      edge.x = edge.x + dx/dy
    sort AET by x value
end scan_fill