

CS-184: Computer Graphics

Lecture #2: Scan Conversion

Prof. James O'Brien
University of California, Berkeley

V2005-02-1.3

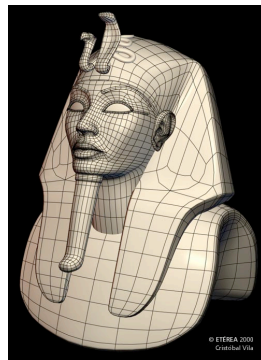
Today

- 2D Scan Conversion
 - Drawing Lines
 - Drawing Curves
 - Filled Polygons
 - Filling Algorithms

2

Drawing a Line

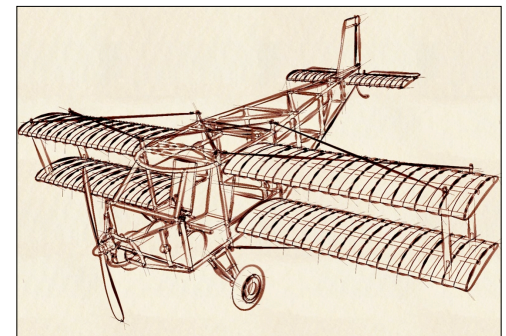
- Basically, its easy... but for the details
- Lines are a basic primitive that needs to be done well...



3

Drawing a Line

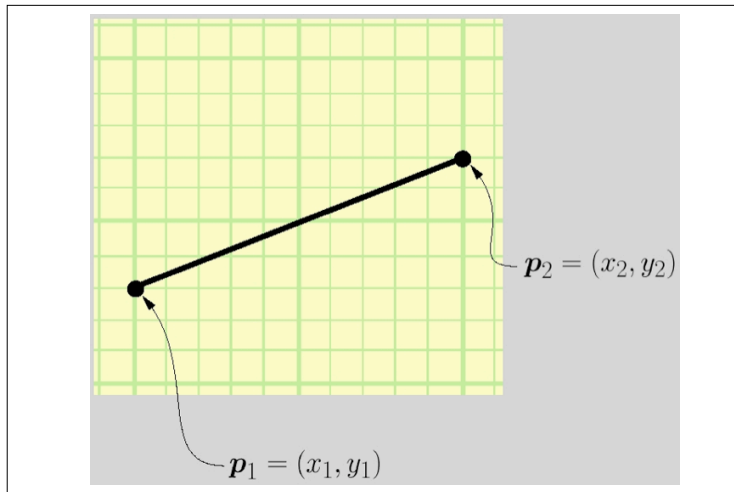
- Basically, its easy... but for the details
- Lines are a basic primitive that needs to be done well...



From "A Procedural Approach to Style for NPR Line Drawing from 3D models,"
by Grabli, Durand, Turquin, Sillion

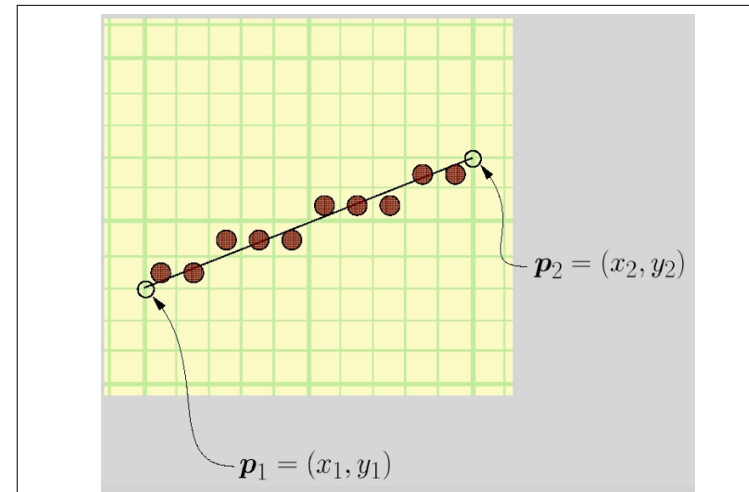
4

Drawing a Line



5

Drawing a Line

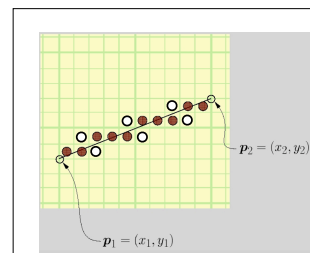


6

Drawing a Line

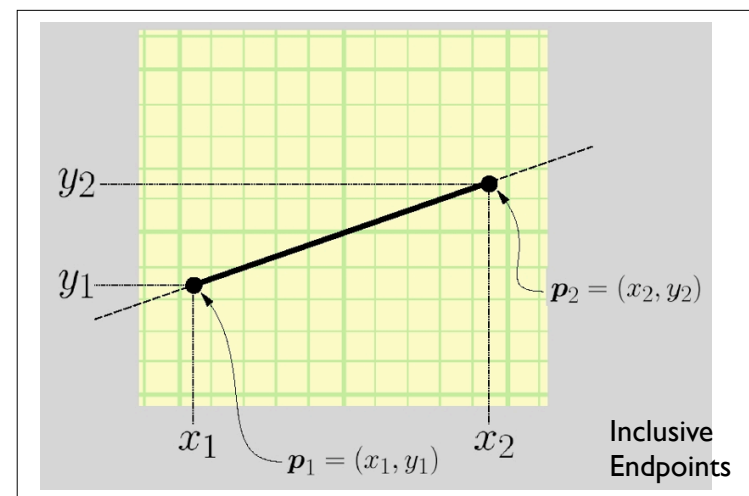
- Some things to consider
 - How thick are lines?
 - How should they join up?
 - Which pixels are the right ones?

For example:



7

Drawing a Line



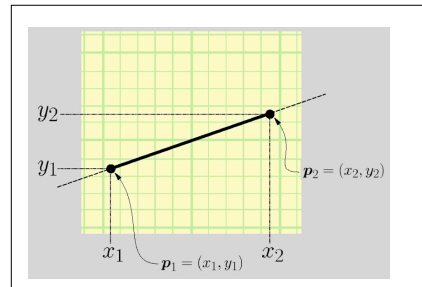
8

Drawing a Line

$$y = m \cdot x + b, x \in [x_1, x_2]$$

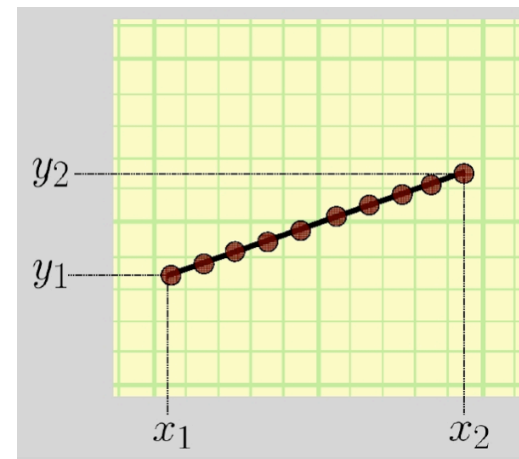
$$m = \frac{y_2 - y_1}{x_2 - x_1}$$

$$b = y_1 - m \cdot x_1$$



9

Drawing a Line



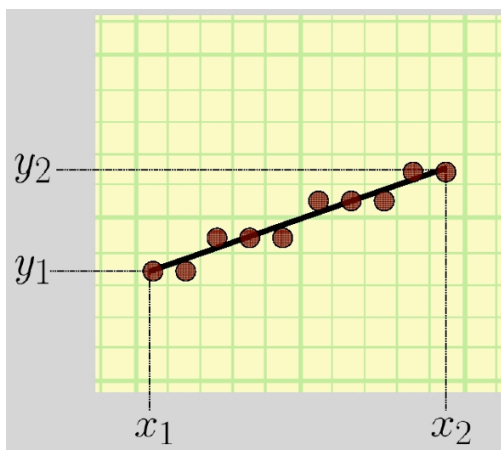
$$\Delta x = 1$$

$$\Delta y = m \cdot \Delta x$$

```
x=x1
y=y1
while(x<=x2)
  plot(x,y)
  x++
  y+=Dy
```

10

Drawing a Line



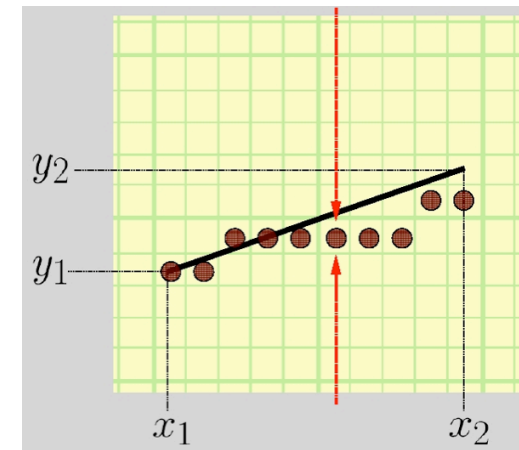
$$\Delta x = 1$$

$$\Delta y = m \cdot \Delta x$$

After rounding

11

Drawing a Line



$$\Delta x = 1$$

$$\Delta y = m \cdot \Delta x$$

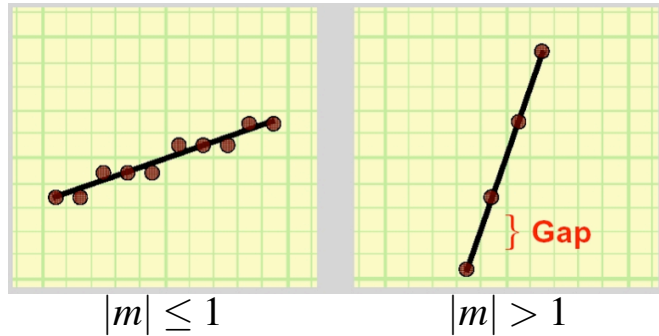
$$y += \Delta y$$

Accumulation of
roundoff errors

How slow is float-
to-int conversion?

12

Drawing a Line



13

Drawing a Line

```
void drawLine-Error1(int x1,x2, int y1,y2)
```

```
float m = float(y2-y1)/(x2-x1)
int x = x1
float y = y1
```

Not exact math

```
while (x <= x2)
```

```
    setPixel(x,round(y),PIXEL_ON)
```

```
    x += 1
    y += m
```

Accumulates errors

14

Drawing a Line

```
void drawLine-Error2(int x1,x2, int y1,y2)
```

```
float m = float(y2-y1)/(x2-x1)
int x = x1
int y = y1
float e = 0.0
```

```
while (x <= x2)
```

```
    setPixel(x,y,PIXEL_ON)
```

```
    x += 1
    e += m
    if (e >= 0.5)
        y+=1
        e-=1.0
```

No more rounding

15

Drawing a Line

```
void drawLine-Error3(int x1,x2, int y1,y2)
```

```
int x = x1
int y = y1
float e = -0.5
```

```
while (x <= x2)
```

```
    setPixel(x,y,PIXEL_ON)
```

```
    x += 1
    e += float(y2-y1)/(x2-x1)
    if (e >= 0.0)
        y+=1
        e-=1.0
```

16

Drawing a Line

```
void drawLine-Error4(int x1,x2, int y1,y2)

int x = x1
int y = y1
float e = -0.5*(x2-x1)           // was -0.5

while (x <= x2)

    setPixel(x,y,PIXEL_ON)

    x += 1
    e += y2-y1                   // was /(x2-x1)
    if (e >= 0.0)                // no change
        y+=1
    e-=(x2-x1)                   // was 1.0
```

17

Drawing a Line

```
void drawLine-Error5(int x1,x2, int y1,y2)

int x = x1
int y = y1
int e = -(x2-x1)                // removed *0.5

while (x <= x2)

    setPixel(x,y,PIXEL_ON)

    x += 1
    e += 2*(y2-y1)              // added 2*
    if (e >= 0.0)              // no change
        y+=1
    e-=2*(x2-x1)                // added 2*
```

18

Drawing a Line

```
void drawLine-Bresenham(int x1,x2, int y1,y2)

int x = x1
int y = y1
int e = -(x2-x1)

while (x <= x2)

    setPixel(x,y,PIXEL_ON)

    x += 1
    e += 2*(y2-y1)
    if (e >= 0.0)
        y+=1
    e-=2*(x2-x1)
```

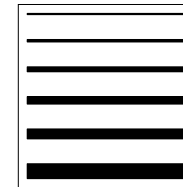
Faster
Not wrong

$$|m| \leq 1$$
$$x_1 \leq x_2$$

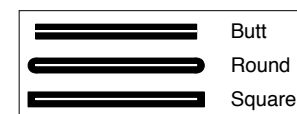
19

Drawing a Line

- How thick?



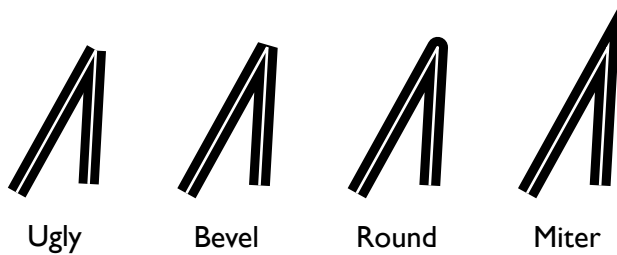
- Ends?



20

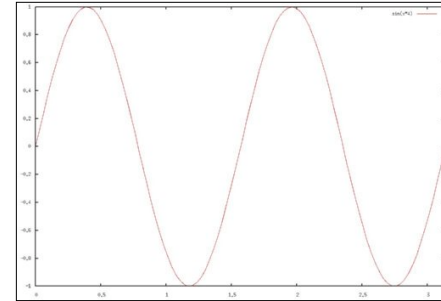
Drawing a Line

- Joining?



21

Drawing Curves



$$y = f(x)$$

Only one value of y for each value of x ...

22

Drawing Curves

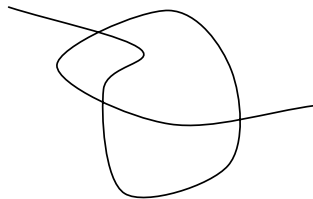
- Parametric curves

- Both x and y are a function of some third parameter

$$x = f(u)$$
$$y = f(u)$$

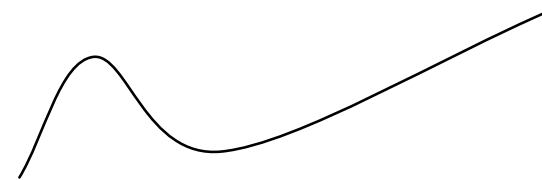
$$\mathbf{x} = \mathbf{f}(u)$$

$$u \in [u_0 \dots u_1]$$



23

Drawing Curves



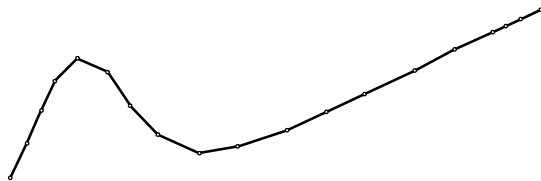
$$\mathbf{x} = \mathbf{f}(u)$$

$$u \in [u_0 \dots u_1]$$

24

Drawing Curves

- Draw curves by drawing line segments
 - Must take care in computing end points for lines
 - How long should each line segment be?



$$\mathbf{x} = \mathbf{f}(u) \quad u \in [u_0 \dots u_1]$$

25

Drawing Curves

- Draw curves by drawing line segments
 - Must take care in computing end points for lines
 - How long should each line segment be?
 - Variable spaced points

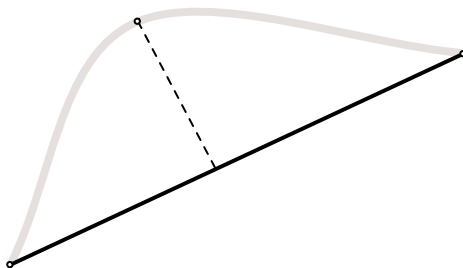


$$\mathbf{x} = \mathbf{f}(u) \quad u \in [u_0 \dots u_1]$$

26

Drawing Curves

- Midpoint-test subdivision

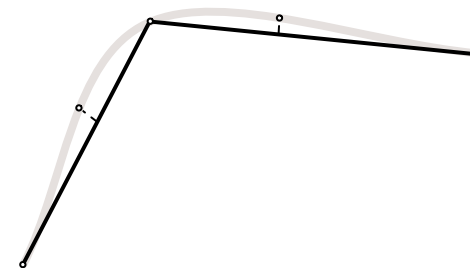


$$|\mathbf{f}(u_{mid}) - \mathbf{l}(0.5)|$$

27

Drawing Curves

- Midpoint-test subdivision

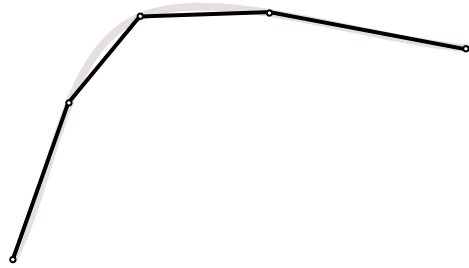


$$|\mathbf{f}(u_{mid}) - \mathbf{l}(0.5)|$$

28

Drawing Curves

- Midpoint-test subdivision

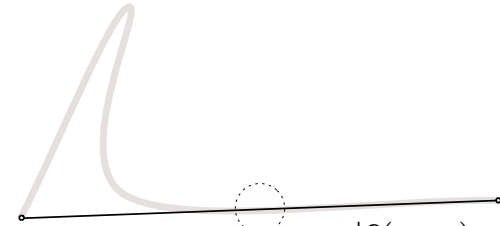


$$|\mathbf{f}(u_{mid}) - \mathbf{l}(0.5)|$$

29

Drawing Curves

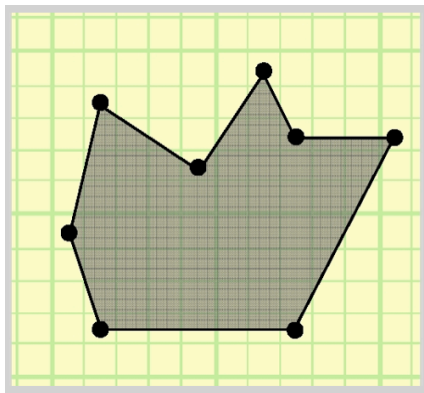
- Midpoint-test subdivision
 - Not perfect
 - We need more information for a guarantee...



$$|\mathbf{f}(u_{mid}) - \mathbf{l}(0.5)|$$

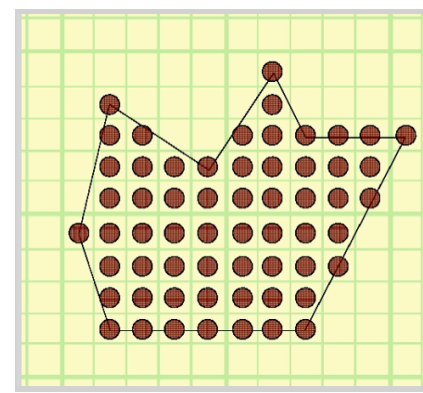
30

Filled Polygons



31

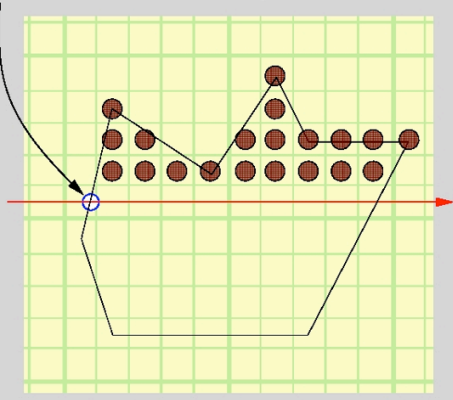
Filled Polygons



32

Filled Polygons

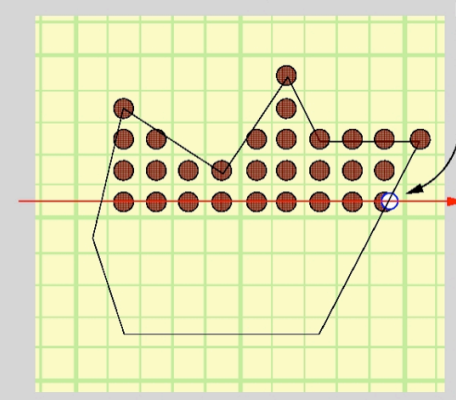
Toggle inside/outside flag to "INSIDE"



33

Filled Polygons

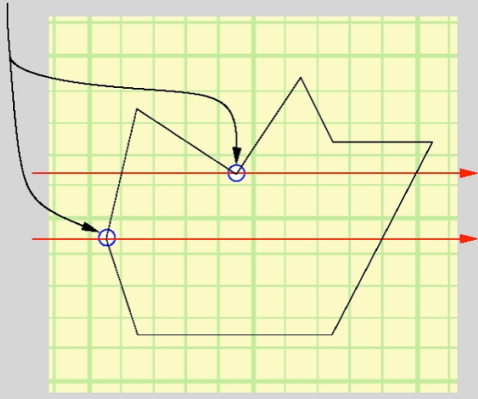
Toggle inside/outside flag to "OUTSIDE"



34

Filled Polygons

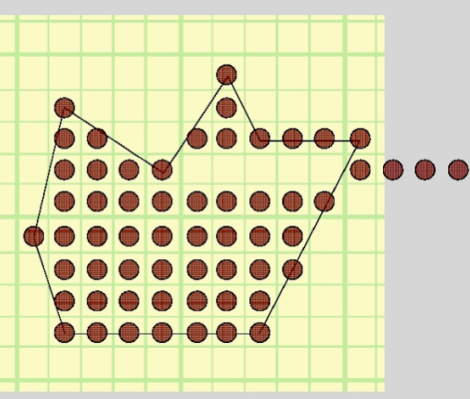
What happens at these locations?



35

Filled Polygons

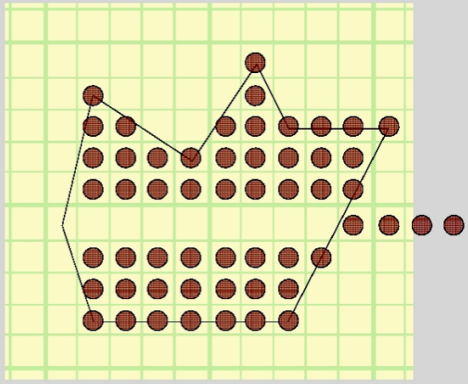
If we count ONCE...



36

Filled Polygons

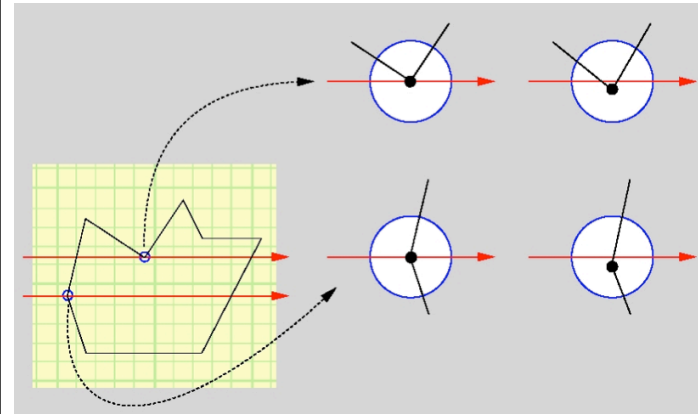
If we count TWICE...



37

Filled Polygons

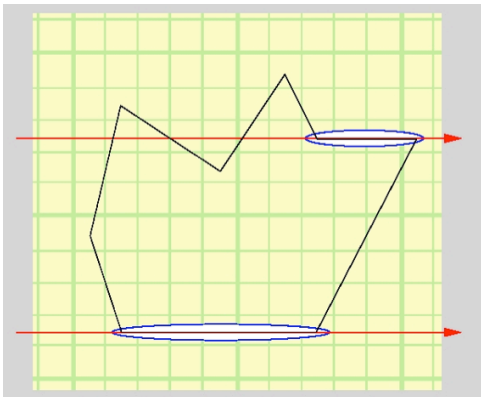
Treat (scan y = vertex y) as (scan y > vertex y)



38

Filled Polygons

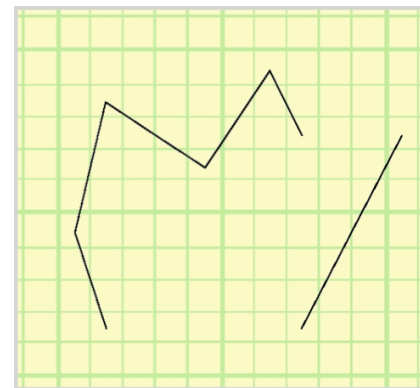
Horizontal edges



39

Filled Polygons

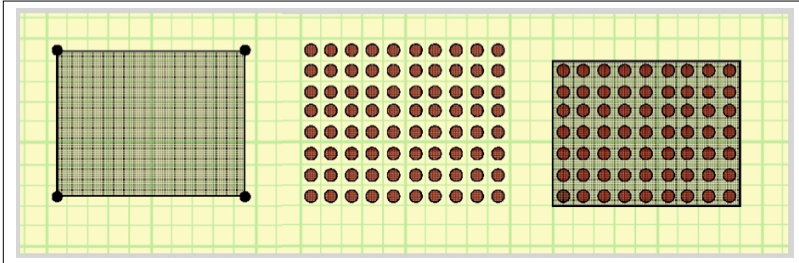
Horizontal edges



40

Filled Polygons

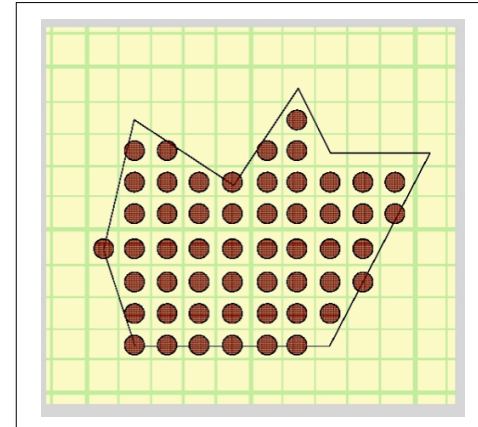
- “Equality Removal” applies to all vertices
- Both x and y coordinates



41

Filled Polygons

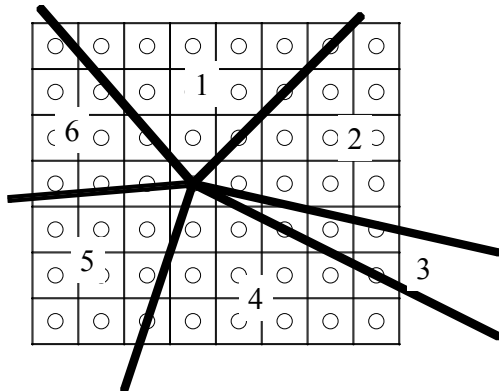
- Final result:



42

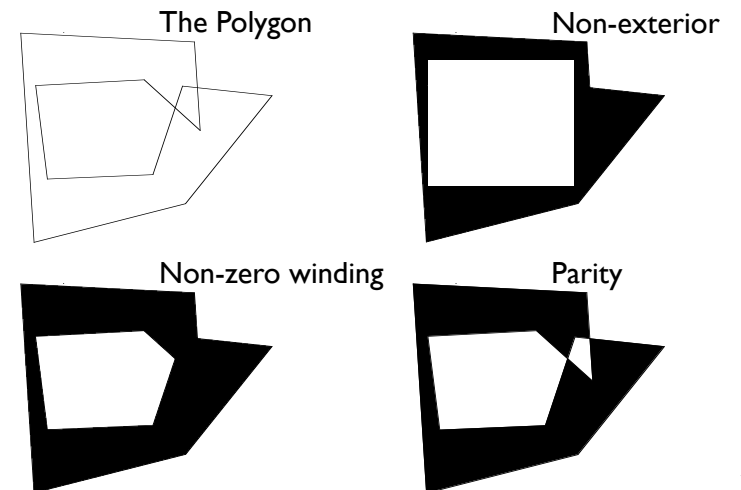
Filled Polygons

- Who does this pixel belong to?



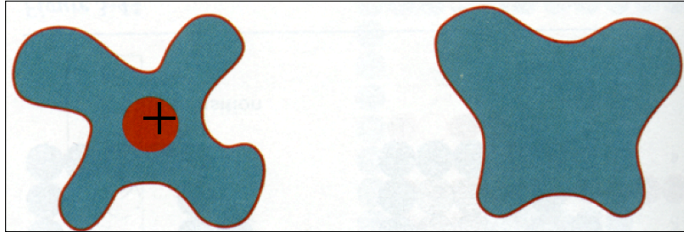
43

Inside/Outside Testing



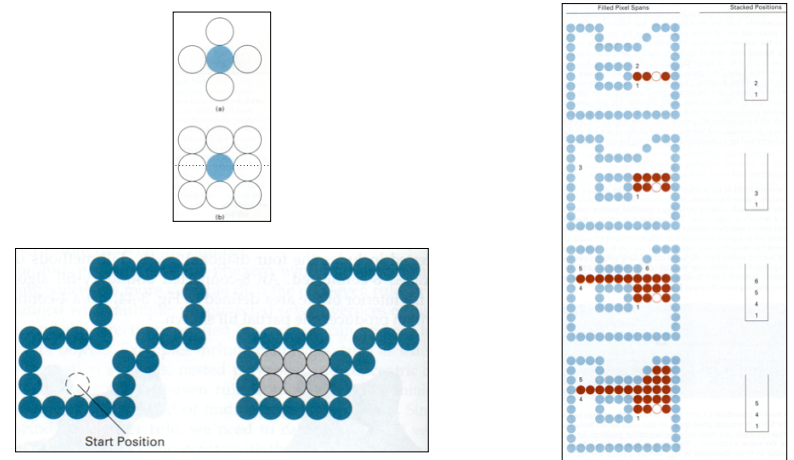
44

Flood Fill



45

Flood Fill



46