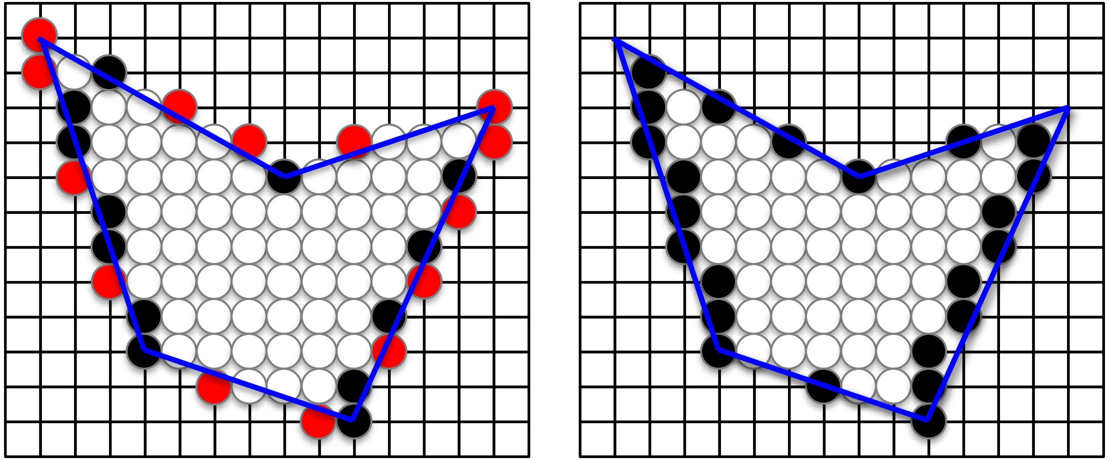
Scan Conversion

Prof. Dr. Markus Gross

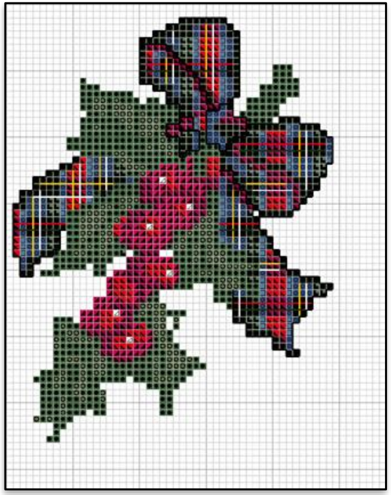


ETH zürich

cgl

Scan Conversion

- Also called rasterization, an old problem

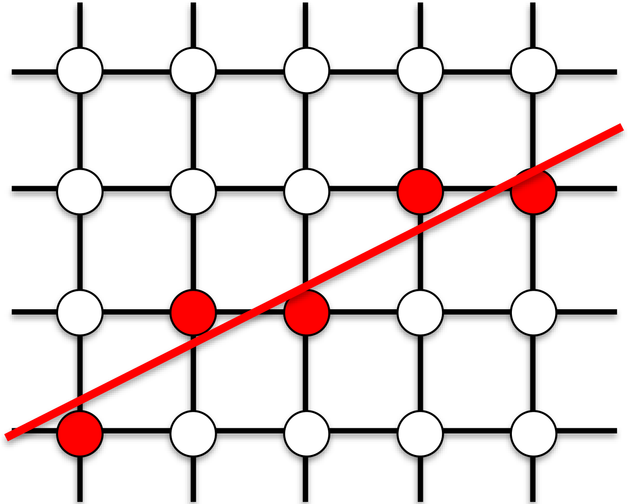


Stitchery (2D)

Lego (3D)

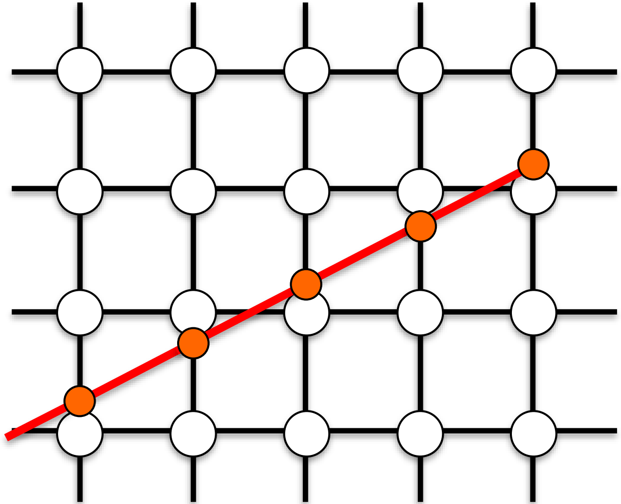
Scan Conversion of Lines

- Generation of discrete pixel values
- Approximation by a finite number of pixels



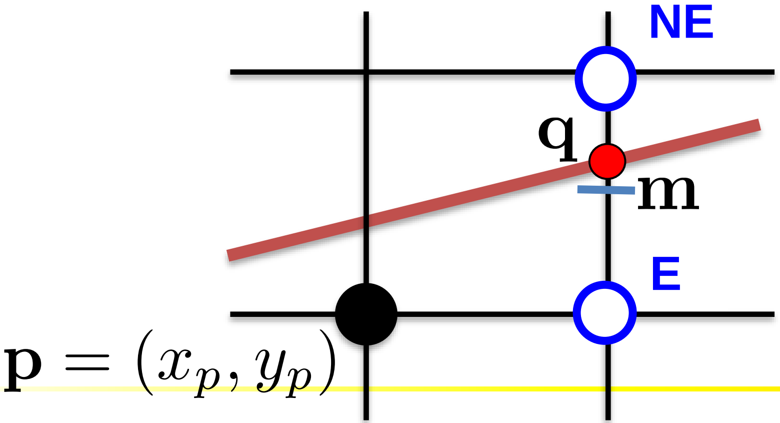
Scan Conversion of Lines

- Bresenham Line
- Choose the closest pixel at each intersection



Scan Conversion of Lines

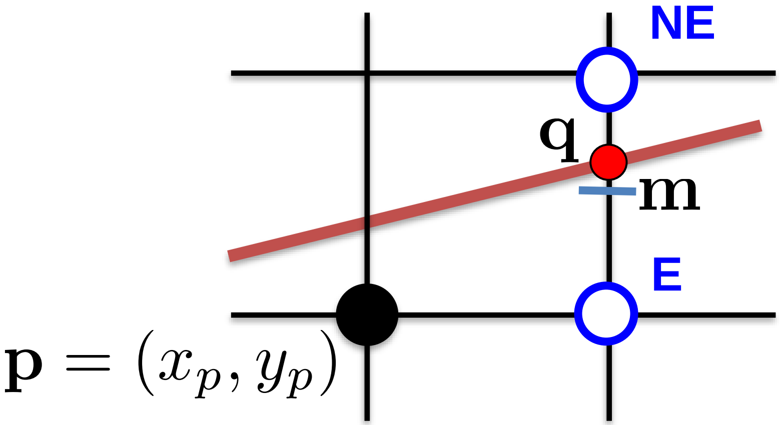
- Bresenham Line
  - Goal: Fast decision which pixel has to be drawn next
  - Criterion: position of the midpoint **m** with respect to the intersection point **q**



Previous pixel Choices for current pixel

Scan Conversion of Lines

- Bresenham Line
  - Goal: Fast decision which pixel has to be drawn next



Previous pixel

Choices for current pixel

Implicit equation of the straight line

$$f(x, y) = ax + by + c = 0$$

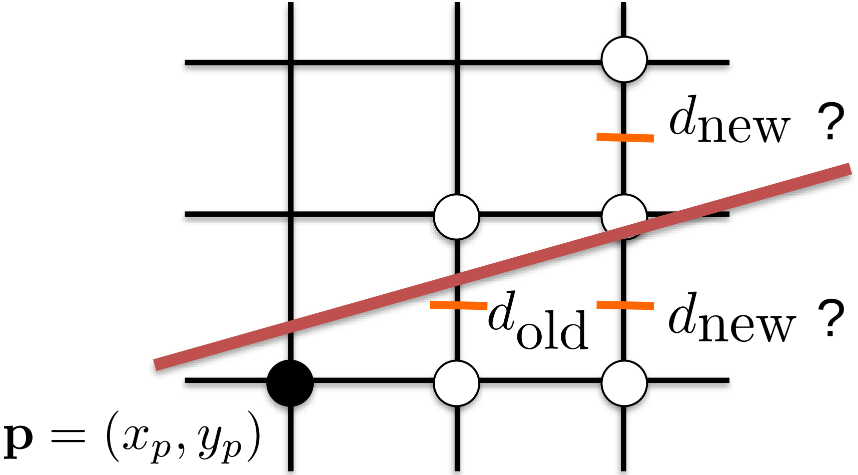
$$d = f(\mathbf{m}) = f(x_p + 1, y_p + 1/2)$$

$d > 0 \Rightarrow$ Select pixel **NE**

$d < 0 \Rightarrow$ Select pixel **E**

Scan Conversion of Lines

- Bresenham Line – update criterion



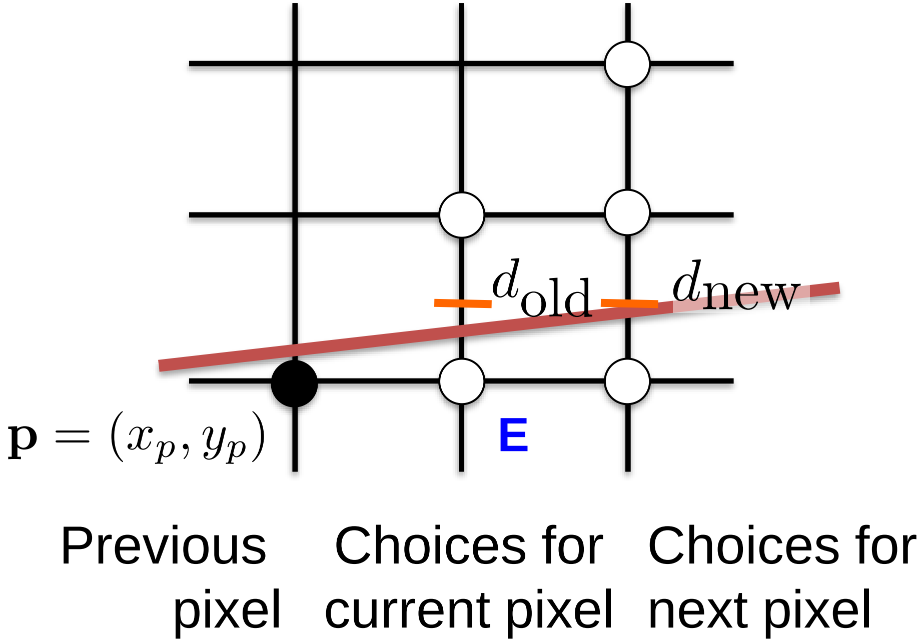
Previous pixel

Choices for current pixel

Choices for next pixel

Scan Conversion of Lines

- Bresenham Line – update criterion, for **E**



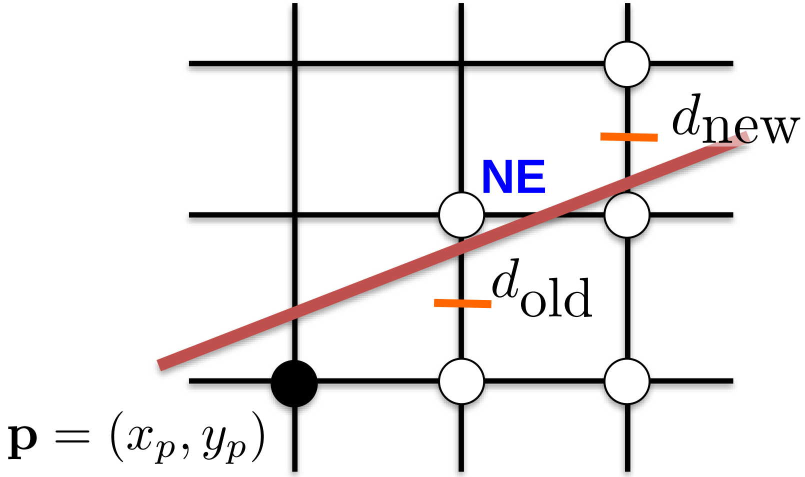
$$\begin{aligned} d_{\mathrm{old}} &= f(x_p + 1, y_p + 1/2) \\ &= a(x_p + 1) + b(y_p + 1/2) + c \end{aligned}$$

$$\begin{aligned} d_{\mathrm{new}} &= f(x_p + 2, y_p + 1/2) \\ &= a(x_p + 2) + b(y_p + 1/2) + c \end{aligned}$$

$$d_{\mathrm{new}} - d_{\mathrm{old}} = a = \Delta y$$

Scan Conversion of Lines

- Bresenham Line – update criterion, for **NE**



Previous pixel

Choices for current pixel

Choices for next pixel

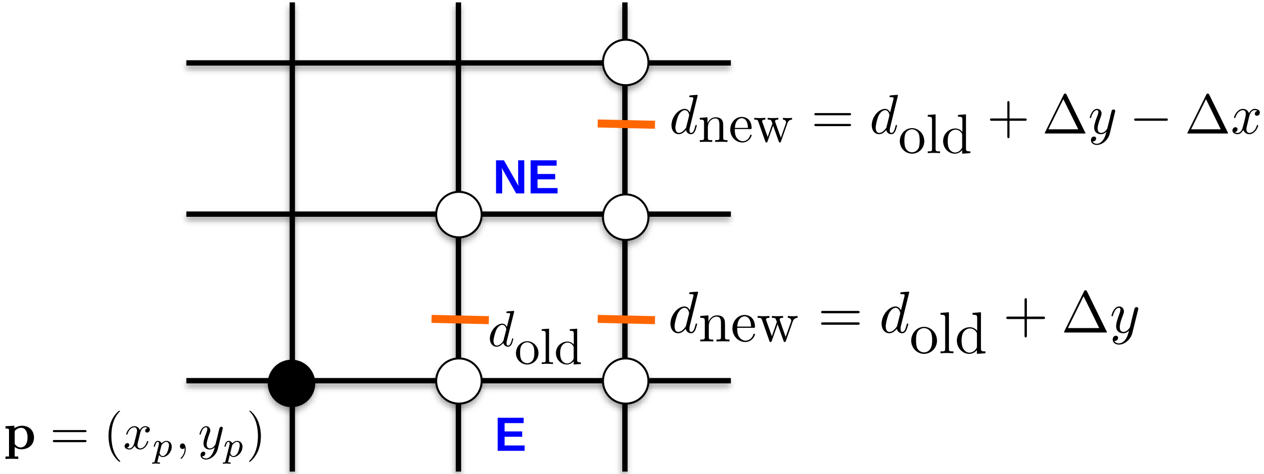
$$
\begin{aligned}
d_{\text{old}} &= f(x_p + 1, y_p + 1/2) \\
&= a(x_p + 1) + b(y_p + 1/2) + c
\end{aligned}
$$

$$
\begin{aligned}
d_{\text{new}} &= f(x_p + 2, y_p + 3/2) \\
&= a(x_p + 2) + b(y_p + 3/2) + c
\end{aligned}
$$

$$d_{\text{new}} - d_{\text{old}} = a + b = \Delta y - \Delta x$$

Scan Conversion of Lines

- Bresenham Line – update criterion



$$d_{\text{new}} = d_{\text{old}} + \Delta y - \Delta x$$

$$d_{\text{new}} = d_{\text{old}} + \Delta y$$

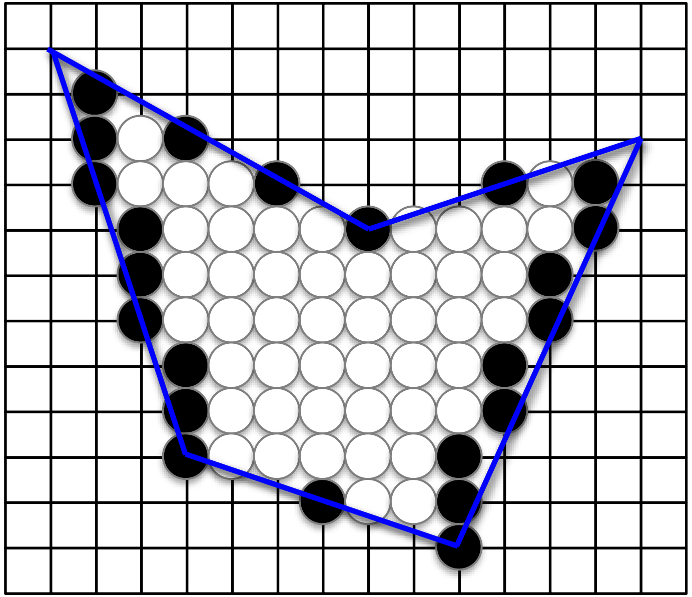
Previous pixel

Choices for current pixel

Choices for next pixel

Scan Conversion of Polygons

- Filled polygons (especially triangles) are the most important graphics primitives
- GPUs can process up to 50 millions of triangles/second



Scan Conversion of Polygons

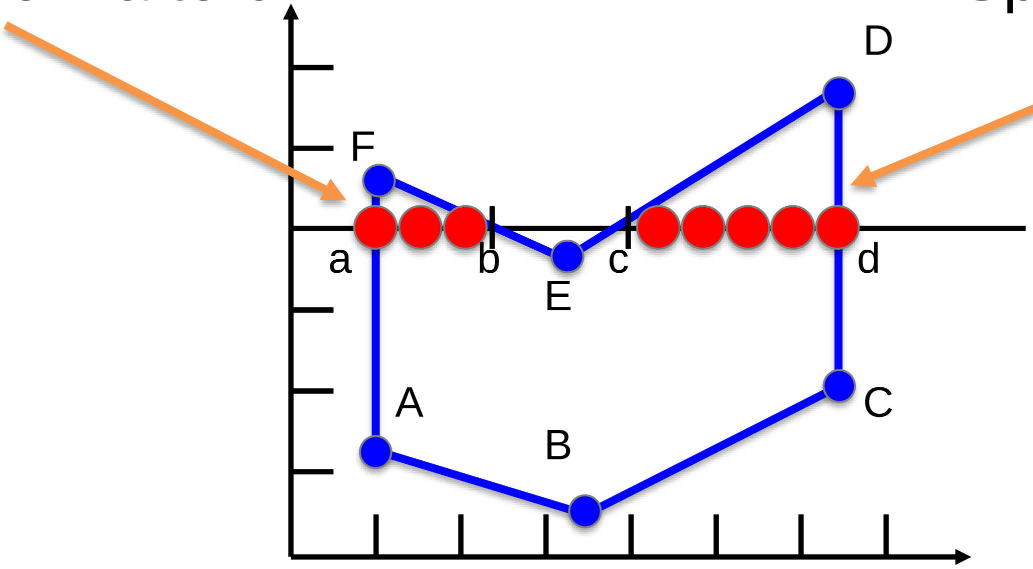
- Spatial coherence
  - Straightforward solution: inside test for each pixel - inefficient!
  - Instead: process scan-line after scan-line
  - Span: group of picked pixels inside a scan-line

Scan Conversion of Polygons

- Spans

Span 1: from a to b

Span 2: from c to d



Scan line

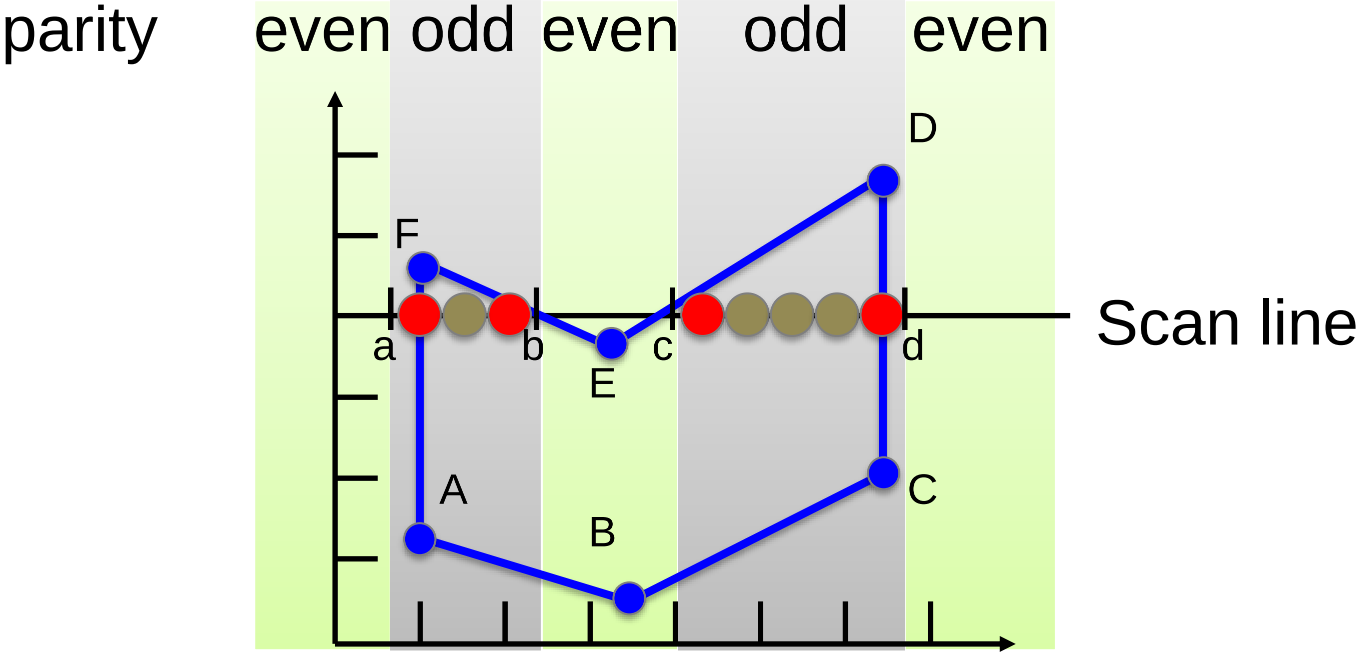
Scan Conversion of Polygons

- Algorithm

  1. Calculate all intersections on the scan line
  2. Sort the intersection points by ascending x-coordinates
  3. Fill all spans in between two consecutive intersection points if the parity is odd.

Scan Conversion of Polygons

- Algorithm


parity
even
odd
even
odd
even
D
F
a
b
c
d
E
A
B
C
Scan line