Line Drawing Algorithm

- Modified from Prof. James O’ Brien’s lecture note on Scan Conversion for CS184 SP06

Drawing a Line

$p_1 = (x_1, y_1)$

$p_2 = (x_2, y_2)$
Drawing a Line

Some things to consider

- How thick are lines?
- How should they join up?
- Which pixels are the right ones?

For example:
Drawing a Line

Inclusive Endpoints

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

\[ m = \frac{y_2 - y_1}{x_2 - x_1} \]

\[ b = y_1 - m \cdot x_1 \]
**Drawing a Line**

\[ \Delta x = 1 \]
\[ \Delta y = m \cdot \Delta x \]

- \( x = x_1 \)
- \( y = y_1 \)
- \( \text{while}(x \leq x_2) \)
  - \( \text{plot}(x, y) \)
  - \( x++ \)
  - \( y+=\Delta y \)

**After rounding**
Drawing a Line

\[ \Delta x = 1 \]
\[ \Delta y = m \cdot \Delta x \]
\[ y' = y + \Delta y \]

Accumulation of roundoff errors

How slow is float-to-int conversion?
Drawing a Line

**void drawLine_Error1(int x1, x2, int y1, y2)**

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

while (x <= x2)
    setPixel(x, round(y), PIXEL_ON)
    x += 1
    y += m
```

*Not exact math*

*Accumulates errors*

---

**void drawLine_Error2(int x1, x2, int y1, y2)**

```c
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*
void drawLine(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 += (y2-y1)/(x2-x1)
    if (e >= 0.0)
        y += 1
        e -= 1.0
void drawLine-Error5(int x1, x2, int y1, y2)

int x = x1
int y = y1
int e = -(x2-x1) \hspace{1cm} // removed *0.5

while (x <= x2)

setPixel(x, y, PIXEL_ON)

x += 1
e += 2*(y2-y1) \hspace{1cm} // added 2*
if (e >= 0.0) \hspace{1cm} // no change
y+=1
e-=2*(x2-x1) \hspace{1cm} // added 2*

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) \hspace{1cm} |m| \leq 1 \hspace{1cm} x_1 \leq x_2

x += 1
 e += 2*(y2-y1) \hspace{1cm} Faster
if (e >= 0.0) \hspace{1cm} Not wrong
  y+=1
  e=2*(x2-x1)
Drawing a Line

- How thick?

- Ends?

Drawing a Line

- Joining?

Ugly  Bevel  Round  Miter