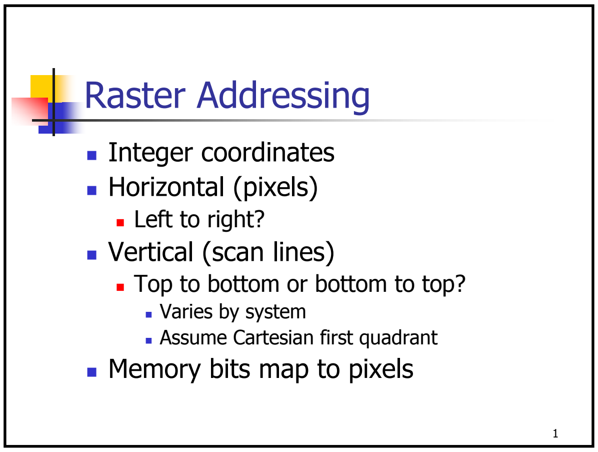
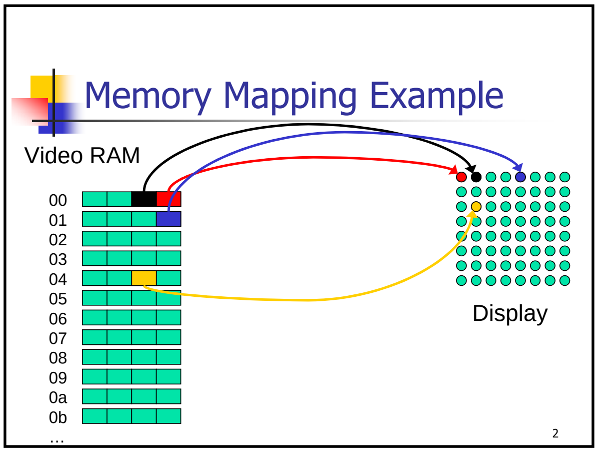


Raster Addressing

- Integer coordinates
- Horizontal (pixels)
  - Left to right?
- Vertical (scan lines)
  - Top to bottom or bottom to top?
    - Varies by system
    - Assume Cartesian first quadrant
- Memory bits map to pixels

1

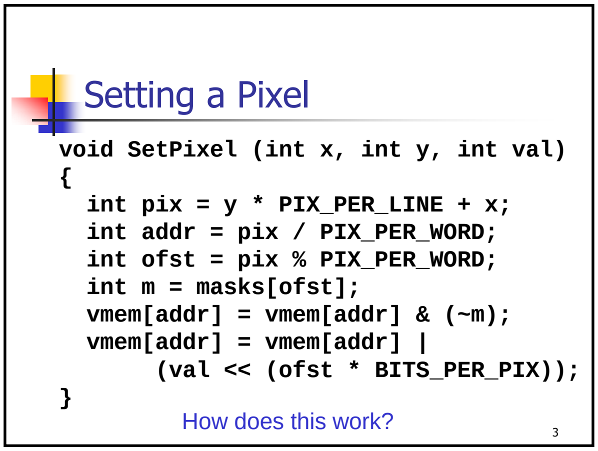


Memory Mapping Example

Video RAM

Display

2

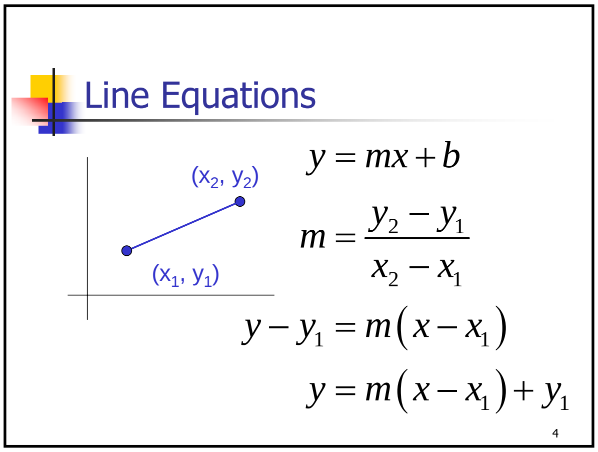


Setting a Pixel

```
void SetPixel (int x, int y, int val)
{
  int pix = y * PIX_PER_LINE + x;
  int addr = pix / PIX_PER_WORD;
  int ofst = pix % PIX_PER_WORD;
  int m = masks[ofst];
  vmem[addr] = vmem[addr] & (~m);
  vmem[addr] = vmem[addr] |
       (val << (ofst * BITS_PER_PIX));
}
```

How does this work?

3



Line Equations

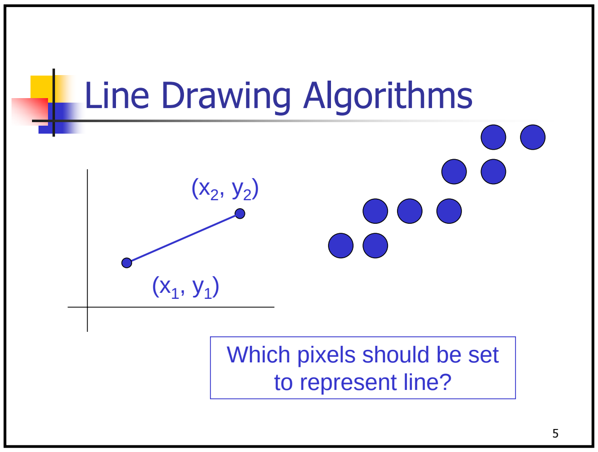
$$y = mx + b$$

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

$$y - y_1 = m\left(x - x_1\right)$$

$$y = m\left(x - x_1\right) + y_1$$

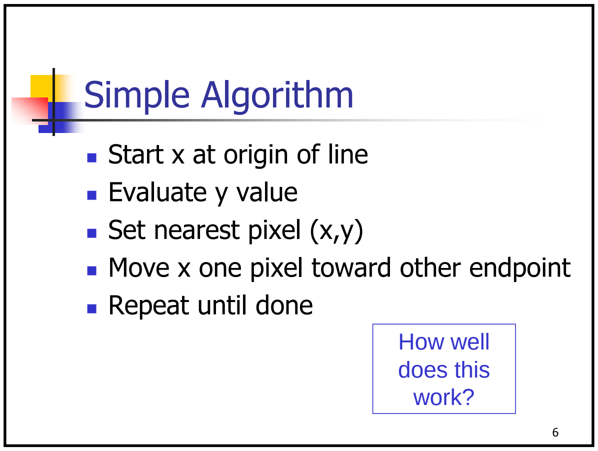
4



Line Drawing Algorithms

Which pixels should be set to represent line?

5

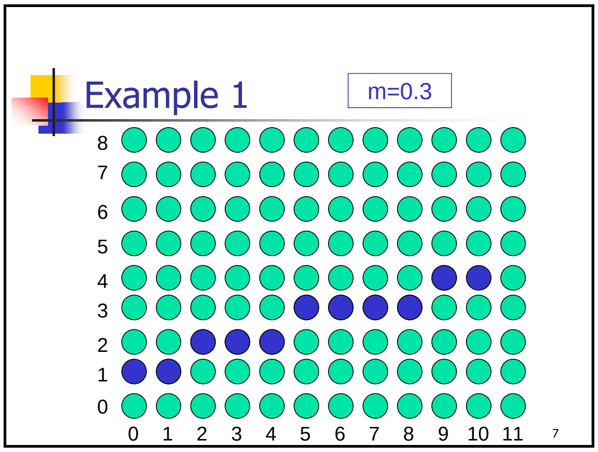


Simple Algorithm

- Start x at origin of line
- Evaluate y value
- Set nearest pixel (x,y)
- Move x one pixel toward other endpoint
- Repeat until done

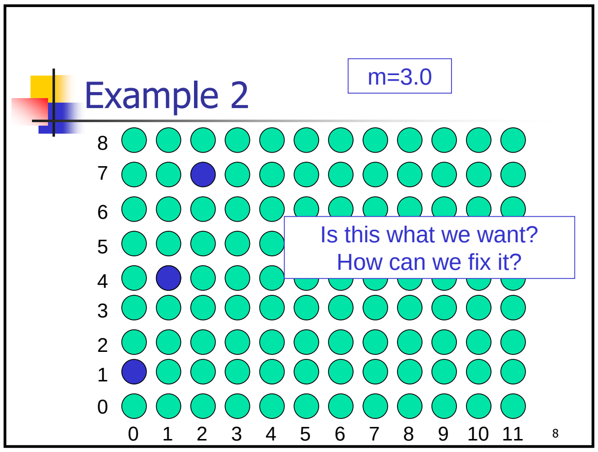
How well does this work?

6



Example 1

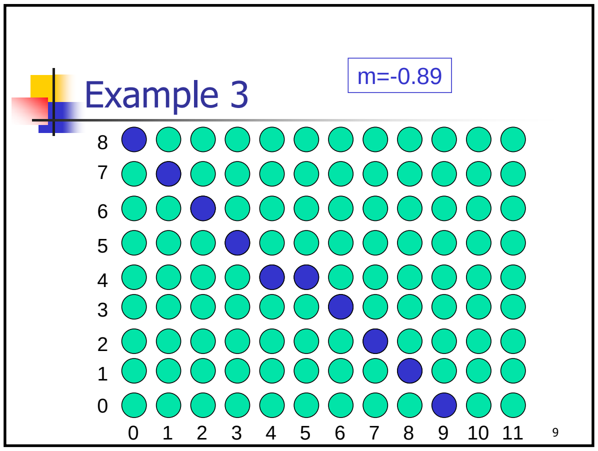
m=0.3



Example 2

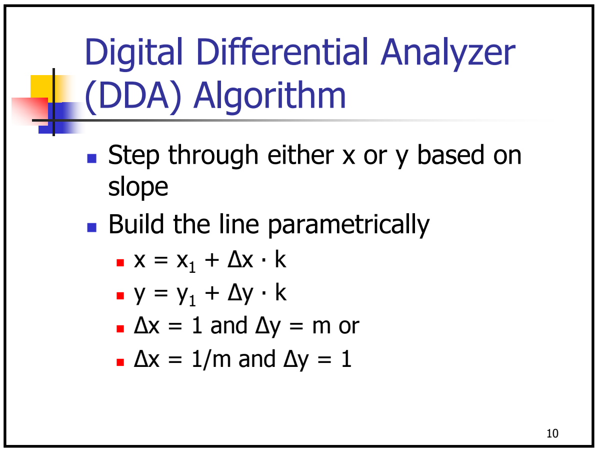
m=3.0

Is this what we want?
How can we fix it?



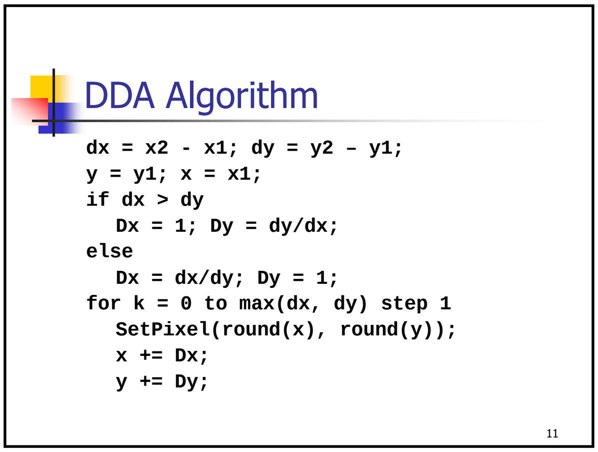
Example 3

m=-0.89



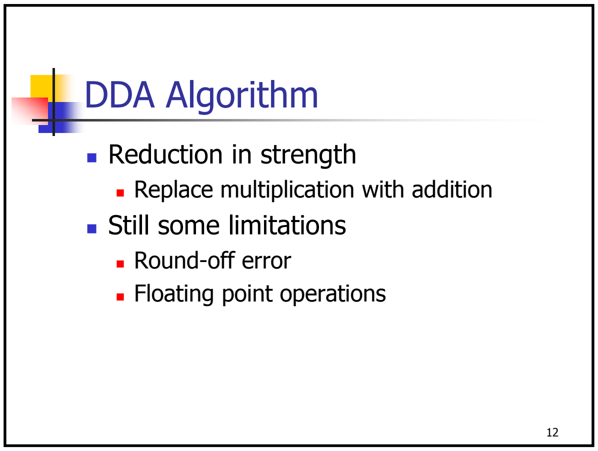
Digital Differential Analyzer (DDA) Algorithm

- Step through either x or y based on slope
- Build the line parametrically
  - $x = x_1 + \Delta x \cdot k$
  - $y = y_1 + \Delta y \cdot k$
  - $\Delta x = 1$ and $\Delta y = m$ or
  - $\Delta x = 1/m$ and $\Delta y = 1$



DDA Algorithm

```
dx = x2 - x1; dy = y2 – y1;
y = y1; x = x1;
if dx > dy
   Dx = 1; Dy = dy/dx;
else
   Dx = dx/dy; Dy = 1;
for k = 0 to max(dx, dy) step 1
   SetPixel(round(x), round(y));
   x += Dx;
   y += Dy;
```



DDA Algorithm

- Reduction in strength
  - Replace multiplication with addition
- Still some limitations
  - Round-off error
  - Floating point operations