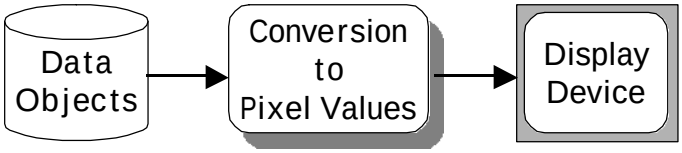
2-D Raster Graphics

Lecture 1

• Graphics Pipeline



- Geometric
- Vector Fields
- Character

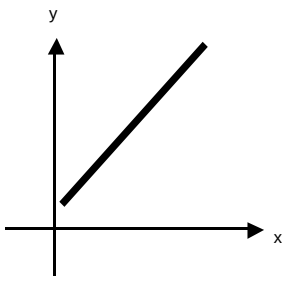
- Projection
- Illumination
- Shading

- Deflect Beam
- Active Phosphor

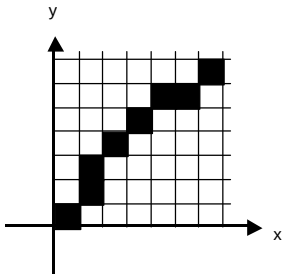
1

2-D Raster Graphics

Lecture 1



Continuous
Real
Coordinates



Discrete
Integer
Coordinates

Rasterization - Conversion of continuous, real valued, data to integer pixel values

2

Line Drawing

Lecture 1

- Line Equations

Slope - Intercept Equation $y = mx + b$

Implicit Equation $Ax + By + C = 0$

Parametric Vector Equation $\vec{P} = \vec{P}_0 + (\vec{P}_1 - \vec{P}_0)t$

Blended (Interpolation) Equation $\vec{P} = \vec{P}_0(1-t) + \vec{P}_1(t)$

3

Line Drawing Algorithms

Lecture 1

- Brute Force Solution

Given: (x_0, y_0) and (x_1, y_1) as endpoints

Compute slope: $m = \frac{\Delta y}{\Delta x} = \frac{y_1 - y_0}{x_1 - x_0}$

Compute y-intercept: $b = y_0 - mx_0$

$x_n = x_0 + n \quad n = \{1,2,3,...\} \qquad y_n = mx_n + b$

$i = \text{int}(x_n + 0.5) \qquad j = \text{int}(y_n + 0.5)$

- Least Effective / Efficient
- Error Accumulation

4

Line Drawing Algorithms

- Basic Incremental Approach

$$y_n = mx_n + b$$

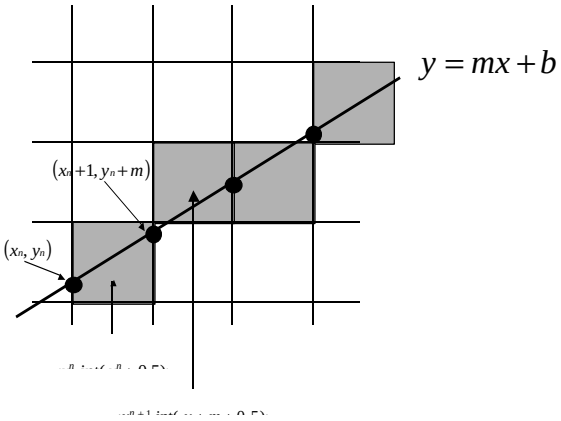
$$y_{n+1} = mx_{n+1} + b \quad \text{where} \quad x_{n+1} = x_n + \Delta x$$

$$y_{n+1} = m(x_n + \Delta x) + b$$
$$= mx_n + b + m\Delta x$$

$$y_{n+1} = y_n + m \quad \text{when} \quad \Delta x = 1$$

- Digital Difference Analyzer (DDA)

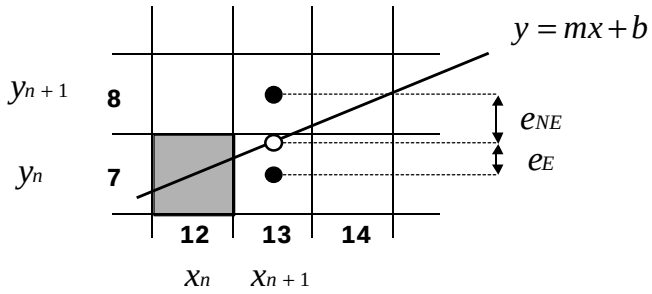
Line Drawing Algorithms



Line Drawing Algorithms

Lecture 1

- Midpoint Checking
- Minimize Errors



- Bresenham Algorithm

Line Drawing Algorithms

Lecture 1

- Compute Error at Each Potential Pixel

(x_n, y_n) = Coordinates of Current pixel

e_{NE} = Error at NorthEast pixel

$= y_n + 1 - y$ where y = a point on the line

$= y_n + 1 - m(x_n + 1) - b$

e_E = Error at East pixel

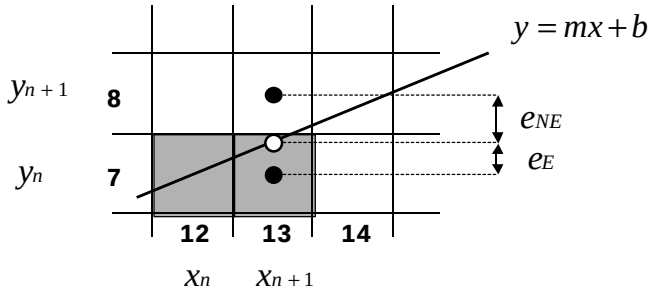
$= y - y_n$

$= m(x_n + 1) + b - y_n$

Line Drawing Algorithms

Lecture 1

If $(e_{NE} < e_N)$
  SetPixel$(x_n + 1, y_n + 1)$
else
  SetPixel$(x_n + 1, y_n)$



9

Line Drawing Algorithms

Lecture 1

- Define Decision Parameter p_n
- Based on Difference of Error Terms

$$p_n \equiv \Delta x(e_E - e_{NE})$$

$$\begin{aligned} e_E - e_{NE} &= \{m(x_n + 1) + b - y_n\} - \{y_n + 1 - m(x_n + 1) - b\} \\ &= 2m(x_n + 1) + 2b - 2y_n - 1 \\ &= 2\frac{\Delta y}{\Delta x}(x_n + 1) - 2y_n + 2b - 1 \end{aligned}$$

10

Line Drawing Algorithms

Lecture 1

$$p_n = 2\Delta y x_n + 2\Delta y - 2y_n\Delta x + 2b\Delta x - \Delta x$$
$$= 2\Delta y x_n - 2\Delta x y_n + C$$

$$C = 2\Delta y + \Delta x(2b-1)$$

Note: C is Independent of Current Position

- Set Pixel Based on the Sign of p_n

If $(p_N < 0)$ $\{e_E < e_{NE}\}$
  SetPixel(x_n+1, y_n)
else $\{e_{NE} \le e_E\}$
  SetPixel(x_n+1, y_n+1)

Line Drawing Algorithms

Lecture 1

- Iterative Calculation for Decision Parameter

$$p_{n+1} = 2\Delta y x_{n+1} - 2\Delta x y_{n+1} + C$$

$$p_{n+1} - p_n = 2\Delta y(x_{n+1} - x_n) - 2\Delta x(y_{n+1} - y_n)$$
$$= 2\Delta y(x_n + 1 - x_n) - 2\Delta x(y_{n+1} - y_n)$$

$$p_{n+1} = p_n + 2\Delta y - 2\Delta x(y_{n+1} - y_n)$$

$$\text{where } (y_{n+1} - y_n) = \begin{cases} 0 & \text{if Pixel E was chosen} \\ 1 & \text{if Pixel NE was chosen} \end{cases}$$

$$p_{n+1} = \begin{cases} p_n + 2\Delta y & p_n < 0 \\ p_n + 2\Delta y - 2\Delta x & p_n \ge 0 \end{cases}$$

Line Drawing Algorithms

- Bresenham Line Drawing Algorithm $|m| < 1$

1. Enter Endpoints: $(x_{start}, y_{start}), (x_{stop}, y_{stop})$
2. Compute Constants: $\Delta x = x_{stop} - x_{start}$, $\Delta y = y_{stop} - y_{start}$,
   $2\Delta y$, $2\Delta y - 2\Delta x$, $p_0 = 2\Delta y - \Delta x$
3. Plot (x_0, y_0)
4. For $n = 0 \cdots (\Delta x - 1)$
   If $(p_n < 0)$
   Plot $(x_n + 1, y_n)$
   $p_{n+1} = p_n + 2\Delta y$
   Else
   Plot $(x_n + 1, y_n + 1)$
   $p_{n+1} = p_n + 2\Delta y - 2\Delta x$

Line Drawing Algorithms

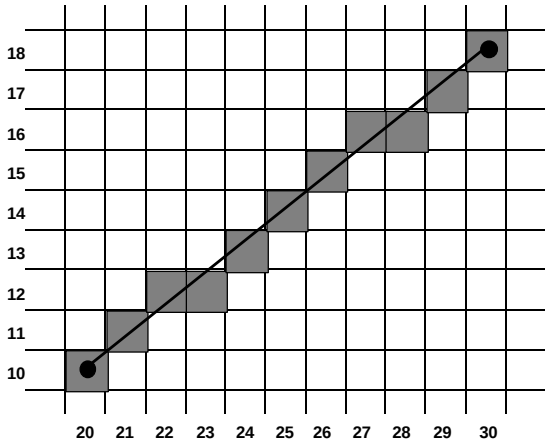
- Bresenham Example

Endpoints: (20,10), (30,18)
$\Delta x = 10, \quad \Delta y = 8, \quad 2\Delta y = 16, \quad 2\Delta y - 2\Delta x = -4$
$p_0 = 2\Delta y - \Delta x = 6 \qquad (x_0, y_0) = (20,10)$

n	p_n	(x_{n+1}, y_{n+1})	n	p_n	(x_{n+1}, y_{n+1})
0	6	(21, 11)	5	6	(26,)
1	2	(22, 12)	6	2	(27,)
2	-2	(23, 12)	7	- 2	(28,)
3	14	(24, 13)	8	14	(29,)
4	10	(25, 14)	9	10	(30,)

Line Drawing Algorithms

Lecture 1



Line Drawing Algorithms

Lecture 1

- Line Drawing Issues
  - Endpoint Order
  - Convention for p=0
  - Slope > 1
- Special Cases
  - Horizontal
  - Vertical
  - Diagonal

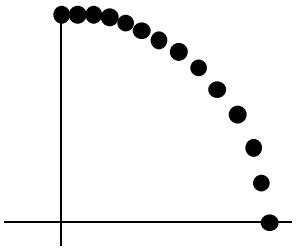
Circle Drawing Algorithms

Lecture 1

- Brute Force Solution

$$x^2 + y^2 = r^2$$
$$y = \sqrt{r^2 - x^2}$$

- Non-Uniform Samples



x	y	Δy
0	10	
1	9.95	0.05
2	9.80	0.15
3	9.54	0.26
4	9.17	0.37
5	8.66	0.51
⋮		
9	4.36	1.64
10	0	4.36

17

Circle Drawing Algorithms

Lecture 1

- Trigonometric Solution

$$x = r\cos q$$
$$y = r\sin q$$

- Sample Spacing Depends on Display Device

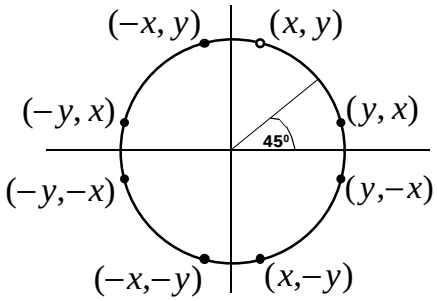
$$\Delta q \approx \frac{1}{r}$$

- Computationally Expensive

18

Circle Drawing Algorithms

- Circular Symmetry



- Compute in 2nd Octant

$x = 0 \quad \text{to} \quad x = y = r/\sqrt{2}$

Circle Drawing Algorithms

- Midpoint Checking
- Define Circle Function

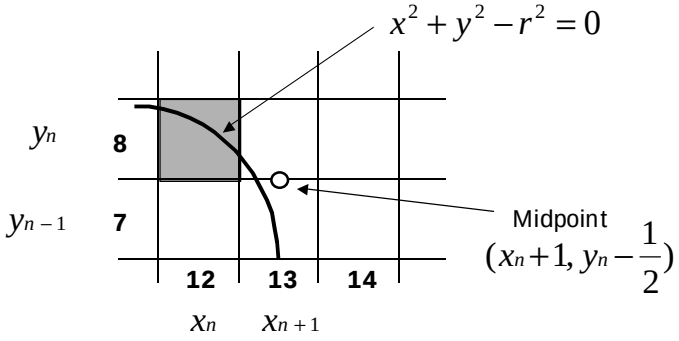
$$f(x, y) = x^2 + y^2 - r^2$$

$$f(x, y)\begin{cases} = 0 & \text{Point is ON the circle} \\ < 0 & \text{Point is INSIDE the circle} \\ > 0 & \text{Point is OUTSIDE the circle} \end{cases}$$

Circle Drawing Algorithms

Lecture 1

- Compute Decision Parameter at Midpoint



Circle Drawing Algorithms

Lecture 1

$$p_n = f(x_n+1, y_n-\frac{1}{2})$$
$$= (x_n+1)^2 + (y_n-\frac{1}{2})^2 - r^2$$

$$p_n \begin{cases} < 0 & \text{Midpoint INSIDE circle, choose } y_n \\ = 0 & \text{Midpoint ON circle, choose } y_n\text{-}1 \\ > 0 & \text{Midpoint OUTSIDE circle, choose } y_n\text{-}1 \end{cases}$$

Circle Drawing Algorithms

- Iterative Calculation for Decision Parameter

$$p_{n+1} = f(x_{(n+1)}+1, y_{(n+1)}-\frac{1}{2})$$

$$= [(x_n+1)+1]^2 + \left[(y_{n+1})-\frac{1}{2}\right]^2 - r^2$$

$$= p_n + 2(x_n+1) + (y_{n+1}^2 - y_n^2) - (y_{n+1}-y_n) + 1$$

$$\text{where} \quad y_{n+1} = \begin{cases} y_n & \text{if } p_n < 0 \\ y_n - 1 & \text{if } p_n \geq 0 \end{cases}$$

Circle Drawing Algorithms

$$p_{n+1} = p_n + 2(x_n+1) + (y_{n+1}^2 - y_n^2) - (y_{n+1}-y_n) + 1$$

$$\text{where} \quad y_{n+1} = \begin{cases} y_n & \text{if } p_n < 0 \\ y_n - 1 & \text{if } p_n \geq 0 \end{cases}$$

$$p_{n+1} = p_n + 2(x_n+1) + \begin{cases} (y_n^2 - y_n^2) - (y_n - y_n) + 1 \\ [(y_n-1)^2 - y_n^2] - [(y_n-1) - y_n] + 1 \end{cases}$$

$$p_{n+1} = p_n + \begin{cases} 2(x_n+1)+1 & \text{if } p_n < 0 \\ 2(x_n+1)+3-2y_n & \text{if } p_n \geq 0 \end{cases}$$

$$= p_n + \begin{cases} 2x_n+3 & \text{if } p_n < 0 \\ 2x_n+5-2y_n & \text{if } p_n \geq 0 \end{cases}$$

Circle Drawing Algorithms

Lecture 1

- Initializing the Decision Parameter

$$p_n = f(x_n+1, y_n-\frac{1}{2}) = (x_n+1)^2 + (y_n-\frac{1}{2})^2 - r^2$$

$$(x_0, y_0) = (0, r)$$

$$p_0 = f(x_0+1, y_0-\frac{1}{2}) = f(1, r-\frac{1}{2})$$

$$= (1)^2 + (r-\frac{1}{2})^2 - r^2$$

$$= 1 + r^2 - r + \frac{1}{4} - r^2$$

$$= \frac{5}{4} - r$$

Circle Drawing Algorithms

Lecture 1

- Simplify Calculations With Integer Arithmetic
- Change of Variables in Iterative Equations

$$q \equiv p - \frac{1}{4}$$

$$p_0 = \frac{5}{4} - r \qquad \Rightarrow \qquad q_0 = 1 - r$$

Decision Parameter Becomes

$$p_n < 0 \quad \Rightarrow \quad q_n < \frac{1}{2} \quad \Rightarrow \quad q_n < 0$$

Since q starts as an int and is incremented by int's

Circle Drawing Algorithms

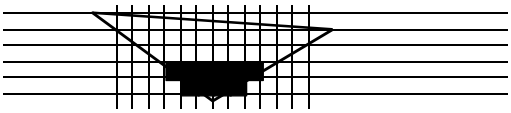
- Midpoint Circle Algorithm

1. Enter Center and Radius: $(x_c, y_c), r$
2. Initialize Parameters: $(x, y) = (0, r), q = 1 - r$
3. Determine Symmetry Points $(0,-r), (r,0), (-r,0)$
4. Translate Points to Circle Center (x_c, y_c) and Plot
5. While $(x \le y)$
   - If $(q < 0)$
     - $q = q + 2x + 3$
     - $x = x + 1$
     - $y = y$
   - Else
     - $q = q + 2(x - y) + 5$
     - $x = x + 1$
     - $y = y - 1$
   - Determine Symmetry Points
   - Translate Points to Circle Center (x_c, y_c) and Plot

   Endwhile

Polygon Filling

- A First Look At Scan Conversion
- Simple Overview
  - For each scanline. . .
    - For $x=x_{start}$ to $x=x_{end}$
      - Color Interior Pixels

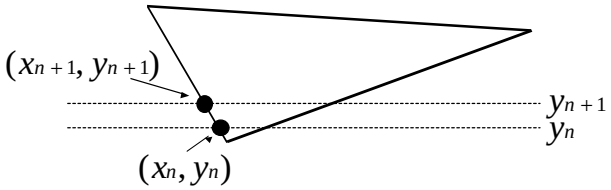


- Potential Problem Areas
  - Computing Intersections
  - Determining Interior
  - Special Cases
    - Intersections at a Vertex
    - Horizontal Lines

Polygon Filling

Lecture 1

- Computing Intersections



$$m = \frac{y_{n+1} - y_n}{x_{n+1} - x_n}$$

$$x_{n+1} = x_n + \frac{1}{m} = x_n + \frac{\Delta x}{\Delta y}$$

29

Polygon Filling

Lecture 1

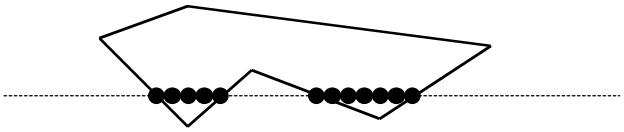
- Sorted Edge Table
  - Sort All Edges
    - Based on y_{min} of each edge
    - Linked list for each scanline bucket
      - Ordered by increasing x
  - Each Entry in Linked List Contains:
    - y_{max}
    - x-intercept at y_{min} vertex
    - x increment (1/m)
  - Process Scanline Linked Lists

30

Polygon Filling

Lecture 1

- Determining Interior Pixels
  - Parity Rule
    - Initialize parity to even
    - Each intersection inverts parity bit
    - Draw pixel when parity is odd
    - Don't draw when parity is even
  - Approaching an Intersection
    - If inside, round down
    - If outside round up

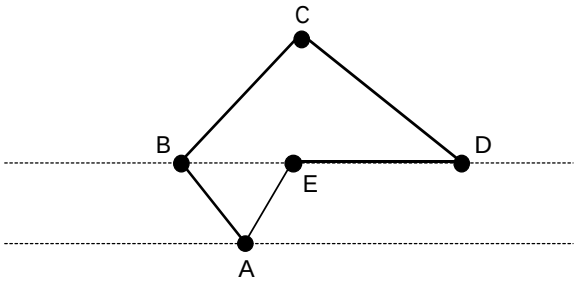


31

Polygon Filling

Lecture 1

- Shared Vertex
  - Include in parity count if vertex is a y_{min} for the edge
- Horizontal Edges
  - Don't include in parity count



32