Chapter 7

Scan Conversion – Drawing on Raster Display (part 1 – Lines)

Lines and Curves

- Explicit - one coordinate as function of the others
  \[ y = f(x) \]
  \[ z = f(x, y) \]
  \[ y = ax + b \]
  \[ y = \sqrt{x^2 - y^2} \]

- Parametric – all coordinates as functions of common parameter
  \[ (x, y, z) = (f(u), f(v)) \]

- Circle
  \[ x(t) = r \cos(t) \]
  \[ y(t) = r \sin(t) \]

Basic Line Drawing

Assume \( x_1 < x_2 \) & line slope absolute value \( k \leq 1 \)

- Midpoint (Bresenham) Algorithm

  - Assumptions:
    \[ x_1 < x_2, y_1 < y_2 \text{ and } \frac{dy}{dx} = \frac{y_2 - y_1}{x_2 - x_1} < 1 \]

  - Idea:
    - Proceed along the line incrementally
    - Have ONLY 2 choices
    - Select one that minimizes error (distance to line)

Midpoint Line Drawing (cont’d)

- Insanely efficient version (less computations inside the loop)
  - Compute \( d \) incrementally
    - At \((x, y)\)
      \[ d_{\text{next}} = d(x, y) + 1 + \text{abs}(dy/dx) \]
  - Increment in \( d \) (after each step)
    - If move east
      \[ \Delta x = 1, \Delta y = 0, d = d + 1, dx = 1, dy = 0 \]
    - If move northeast
      \[ \Delta x = 1, \Delta y = 1, d = d + 1, dx = 1, dy = 1 \]

Midpoint Line Algorithm (version 1)

- Midpoint Line Drawing (cont’d)

  - Given point \( P = (x, y) \), distance \( d(x, y) \) is signed distance of \( \rho \) to \( \Gamma \) (up to normalization factor)

  - \( \theta \) is zero for \( P \in \Gamma \)

Scan Conversion - Rasterization

- Convert continuous rendering primitives into discrete fragments/pixels

  - Lines
    - Bresenham
  - Triangles
  - Flood Fill
  - Scanline
  - Implicit formulation

Scan Conversion - Lines

- Given segment equation fill in the pixels

  - In drawings below – grid points = centers of pixels

Scan Conversion – Lines

Given segment equation fill in the pixels

- Lines
  - Bresenham
- Triangles
- Flood Fill
- Scanline
- Implicit formulation

Scan Conversion – Drawing on Raster Display (part 1 – Lines)

Lines and Curves

- Implicit - define as “zero set” of function of all the parameters
  \[ (x, y): F(x, y) = 0 \]
  \[ (x, y): F(x, y) = 0 \]

- Defines partition of space
  \[ \{(x, y): F(x, y) > 0 \} \cup \{(x, y): F(x, y) = 0 \} \cup \{(x, y): F(x, y) < 0 \} \]

Midpoint Line Algorithm

- Idea:
  - Can this algorithm use integer arithmetic ?

Midpoint Line Algorithm (version 1)

- Example: extending to higher order curves – e.g. circles

Midpoint Line Algorithm (cont’d)

- Starting point satisfies \( d(x, y) = 0 \)
- Each step moves right (east) or upper right (northeast)
- Sign of \( d(x + 1; y + 2) \) indicates if to move east or northeast

Midpoint Examples

- Question: Is there a problem with this algorithm (horizontal vs. diagonal lines)?
Error Function Intuition

- Error function $d$ can be viewed as an explicit surface:

$$d(x,y) = 2(adx - sye + c)$$