1. Digital Differential Analyzer (DDA) Line Drawing Algorithm

(14 Marks Answer)

Introduction

The **Digital Differential Analyzer (DDA)** is a **line drawing algorithm** used in computer graphics to generate a straight line between two given points on a raster display. It is a **scan-conversion algorithm** that incrementally calculates intermediate points using floating-point arithmetic.

Principle of DDA Algorithm

The DDA algorithm is based on the **line equation**:

$$y = mx + c$$

where

$$m = \frac{\Delta y}{\Delta x}$$

Instead of calculating all points directly, DDA increments one coordinate and computes the other using the slope.

Steps of DDA Algorithm

Steps of DDA Algorithm

Let the two endpoints of the line be:

$$(x_1, y_1) \text{ and } (x_2, y_2)$$

1. Compute differences:

$$\Delta x = x_2 - x_1$$

$$\Delta y = y_2 - y_1$$

2. Determine number of steps:

$$\text{steps} = \max(|\Delta x|, |\Delta y|)$$

3. Compute increments:

$$x_{inc} = \frac{\Delta x}{steps}$$

$$y_{inc} = \frac{\Delta y}{steps}$$

1.

4. Initialize:

$$x = x_1, \quad y = y_1$$

5. For each step:

- Plot the pixel at $(round(x), round(y))$
- Increment:

$$x = x + x_{inc}$$

$$y = y + y_{inc}$$

Example

Draw a line from (2, 3) to (10, 7)

$$\Delta x = 8, \quad \Delta y = 4$$

$$steps = 8$$

$$x_{inc} = 1, \quad y_{inc} = 0.5$$

The algorithm plots pixels incrementally until the end point is reached.

Advantages

- Simple and easy to understand
- Suitable for lines with any slope
- Useful for educational purposes

Disadvantages

- Uses floating-point arithmetic
- Rounding errors may occur
- Slower compared to Bresenham's algorithm

Applications

- Basic graphics systems
- Educational tools
- Simple drawing applications

Conclusion

The DDA algorithm is a straightforward method for line generation. Although simple, its reliance on floating-point calculations makes it less efficient than integer-based algorithms.

---

2. Bresenham's Line Drawing Algorithm

(14 Marks Answer)

Introduction

Bresenham's Line Drawing Algorithm is an efficient raster line drawing algorithm that generates a straight line using **only integer arithmetic**. It is faster and more accurate than the DDA algorithm and is widely used in real-time graphics systems.

Principle of Bresenham's Algorithm

The algorithm determines which pixel is closer to the actual line by evaluating a **decision parameter**. At each step, it chooses between two possible pixels based on this parameter.

Assumptions

- The line slope lies between **0 and 1**
- Line is drawn from left to right
  (Other cases are handled by symmetry)

Algorithm Steps

Given endpoints:

$$(x_1, y_1) \text{ and } (x_2, y_2)$$

1. Compute differences:

$$\Delta x = x_2 - x_1$$

$$\Delta y = y_2 - y_1$$

2. Initialize decision parameter:

$$p_0 = 2\Delta y - \Delta x$$

3. Initialize:

$$x = x_1, \quad y = y_1$$

1. 4. Plot the first point
For each x from (x_1) to (x_2):

4.plot the first point.

5. For each x from x_1 to x_2:

- If $p_k < 0$:
  - Choose East pixel
  - $p_{k+1} = p_k + 2\Delta y$
- Else:
  - Choose North-East pixel
  - $y = y + 1$
  - $p_{k+1} = p_k + 2\Delta y - 2\Delta x$
- Increment x
- Plot the selected pixel

Example

Draw a line from (2, 2) to (10, 6)

$$\Delta x = 8, \quad \Delta y = 4$$

$$p_0 = 2(4) - 8 = 0$$

Using the decision parameter, pixels are chosen efficiently at each step.

Advantages

- Uses only integer calculations
- Faster and more accurate
- No rounding errors
- Suitable for real-time graphics

Disadvantages

- Slightly complex compared to DDA
- Initial explanation is less intuitive

Applications

- Computer graphics hardware
- Game engines
- Window systems
- Embedded graphics displays

Conclusion

Bresenham’s algorithm is a highly efficient line drawing method that outperforms DDA in speed and accuracy, making it the preferred choice in practical graphics systems.

---

1. Mid-Point Circle Drawing Algorithm

(14 Marks Answer)

Introduction

The **Mid-Point Circle Drawing Algorithm** is an efficient raster scan algorithm used to generate a circle on a pixel-based display. It is an **incremental algorithm** that determines the next pixel position by evaluating a **decision parameter** using only **integer arithmetic**, making it faster and more accurate.

Principle of Mid-Point Circle Algorithm

The circle is represented by the equation:
[
x^2 + y^2 = r^2
]

The algorithm works by:

- Calculating the midpoint between two possible pixels
- Deciding whether the midpoint lies **inside or outside** the circle
- Selecting the pixel closest to the actual circle path

Circle Symmetry

A circle is symmetric in **8 octants**.
Hence, calculating pixels for **one octant** and reflecting them reduces computation.

Algorithm Steps

Algorithm Steps

Given:

- Center of circle: (x_c, y_c)
- Radius: r

1. Initialize:

$$x = 0, \quad y = r$$

2. Initialize decision parameter:

$$p_0 = 1 - r$$

3. Plot initial points using 8-way symmetry

4. While $x \leq y$:
    - If $p_k < 0$:
        - Midpoint is inside circle
        - Choose East pixel
        - $p_{k+1} = p_k + 2x + 3$
    - Else:
        - Midpoint is outside circle
        - Choose South-East pixel
        - $y = y - 1$
        - $p_{k+1} = p_k + 2x - 2y + 5$
    - Increment $x = x + 1$
    - Plot symmetrical points

Advantages

- Uses only integer arithmetic
- Faster than trigonometric methods
- Exploits symmetry for efficiency
- Accurate and smooth circle generation

Disadvantages

- Limited to raster displays
- Not suitable for vector graphics directly

Applications

- CAD systems
- Computer games
- GUI design
- Embedded systems displays

Conclusion

The Mid-Point Circle Algorithm is a highly efficient and accurate method for drawing circles, widely used in real-time graphics applications due to its simplicity and speed.

---

2. Mid-Point Ellipse Drawing Algorithm

(14 Marks Answer)

Introduction

The **Mid-Point Ellipse Drawing Algorithm** is used to generate an ellipse on a raster display. It is an extension of the mid-point concept used for circles and uses **integer calculations** to determine the pixel positions closest to the ellipse boundary.

Ellipse Equation

The standard equation of an ellipse is:

$$\frac{x^2}{a^2} + \frac{y^2}{b^2} = 1$$

where:

- a = semi-major axis
- b = semi-minor axis

Principle of Mid-Point Ellipse Algorithm

The algorithm divides the ellipse into **two regions** based on slope:

- **Region 1**: Slope magnitude < 1
- **Region 2**: Slope magnitude ≥ 1

In each region, a decision parameter is used to choose the next pixel.

Ellipse Symmetry

An ellipse is symmetric about both axes.
Hence, pixels calculated in **one quadrant** can be reflected to other quadrants.

Algorithm Steps

Given:

- Center: ((x_c, y_c))
- Semi-major axis (a)
- Semi-minor axis (b)

Region 1

1. Initialize:

$$x = 0, \quad y = b$$

2. Initial decision parameter:

$$p_{1_0} = b^2 - a^2 b + \frac{1}{4}a^2$$

3. While $2b^2x \leq 2a^2y$:
   - If $p_1 < 0$:
     - Choose East pixel
     - $p_1 = p_1 + 2b^2x + b^2$
   - Else:
     - Choose South-East pixel
     - $y = y - 1$
     - $p_1 = p_1 + 2b^2x - 2a^2y + b^2$
   - Increment x

Region 2

1. Initialize decision parameter:

$$p_{2_0} = b^2(x + 0.5)^2 + a^2(y - 1)^2 - a^2b^2$$

2. While $y \geq 0$:
   - If $p_2 > 0$:
     - Choose South pixel
     - $p_2 = p_2 - 2a^2y + a^2$
   - Else:
     - Choose South-East pixel
     - $x = x + 1$
     - $p_2 = p_2 + 2b^2x - 2a^2y + a^2$
   - Decrement y

Advantages

- Uses integer arithmetic
- Efficient and accurate
- Exploits symmetry
- Suitable for real-time rendering

Disadvantages

- More complex than circle algorithm
- Two-region handling required

Applications

- Elliptical object modeling
- Graph plotting
- CAD and CAM systems
- GUI design

Conclusion

The Mid-Point Ellipse Algorithm provides an efficient and accurate method for rasterizing ellipses and is widely used in computer graphics applications.