International Journal on Science and Technology (IJSAT)
E-ISSN: 2229-7677 ● Website: www.ijsat.org ● Email: editor@ijsat.org

A Relative Study of Line Drawing Algorithms: Digital Differential Analyzer and Bresenham's

Ashish Kumar Gupta[1], Akash Yadav[2]

[1]Assistant Professor,
Department of Computer Science,
Microtek College of Management & Technology
[2]Assistant Professor,
Department of Computer Science,
Microtek College of Management & Technology

Abstract

Line drawing is a fundamental operation in computer graphics where continuous lines are represented using discrete pixels. Efficient algorithms are required to produce accurate and fast results. This paper presents a detailed comparison between the Digital Differential Analyzer (DDA) and Bresenham's line drawing algorithms. The comparison is based on mathematical formulation, computational complexity, accuracy, and performance. The results show that Bresenham's algorithm performs better due to its use of integer calculations and reduced computational costs.

Keywords

DDA Algorithm, Bresenham's Algorithm, Line Drawing, Computer Graphics, Rasterization

Introduction

In raster graphics systems, images are formed using a grid of pixels. Drawing a straight line between two points is one of the most basic operations. However, selecting the correct pixels to represent a straight line is challenging.

The main challenges are:

- Reducing jagged edges
- Maintaining accuracy
- Improving computational efficiency

To solve these problems, line drawing algorithms such as DDA and Bresenham's are used.

Mathematical Formulation

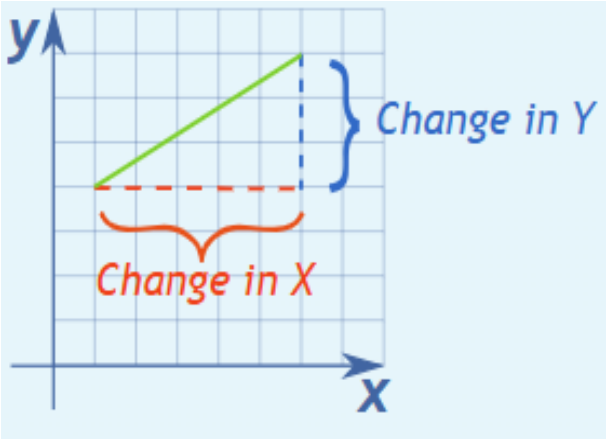
A straight line is represented as:

$$y = mx + c$$

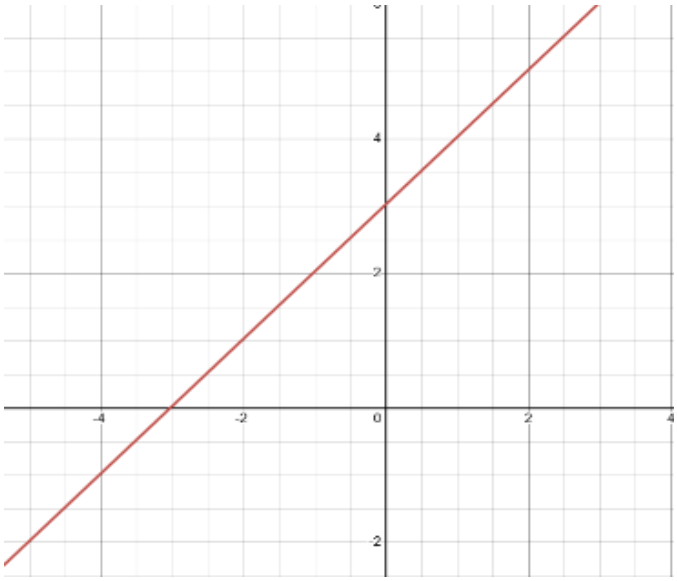
When two points are given (x1, y1) and (x2, y2). We can calculate the slope m

Where slope or gradient is:

$$m = \frac{y2-y1\ (\text{Change in Y})}{x2-x1\ (\text{Change in X})} \text{ or } m = \frac{y1-y2\ (\text{Change in Y})}{x1-x2\ (\text{Change in X})}$$



When m = 1 and c = 3.03, Line looks like in the above graph.



Error in Rasterization: Error = Actual Line − Rasterized Line

Objective: Minimize error and maximize speed.

1. Digital Differential Analyzer (DDA) Algorithm

Step 1: Start
Step 2: Input starting point (x1, y1) and ending point (x2, y2)
Step 3: Calculate dx and dy
- dx = x2 - x1
- dy = y2 - y1

Step 4: Find number of steps
- steps = max(|dx|, |dy|)

Step 5: Calculate increment values
- x_inc = dx / steps
- y_inc = dy / steps

Step 6: Initialize
- x = x1
- y = y1

Step 7: Repeat for i = 1 to steps
- Plot (round(x), round(y))
- x = x + x_inc
- y = y + y_inc

Step 8: Stop

Formula

$$x_{k+1} = x_k + x_{inc}$$
$$y_{k+1} = y_k + y_{inc}$$

Where:

$$dy = y2 - y1$$
$$dx = x2 - x1$$
$$steps = \max(|dx|, |dy|)$$
$$x_{inc} = \frac{dx}{steps}, y_{inc} = \frac{dy}{steps}$$

Working

- DDA calculates intermediate points using floating point increments.

Advantages

- Simple to implement
- Easy to understand

Disadvantages

- Uses floating point operations
- Rounding errors occur
- Slower performance

2. Bresenham's Line Drawing Algorithm

Step 1: Start
Step 2: Input starting point (x1, y1) and ending point (x2, y2)
Step 3: Calculate dx and dy
- dx = x2 - x1
- dy = y2 - y1

Step 4: Initialize decision parameter
- p = 2*dy – dx

Step 5: Initialize starting point
- x = x1

- y = y1

Step 6: Plot initial point (x, y)

Step 7: Repeat while x < x2

- x = x + 1
- If p < 0 then
  - p = p + 2*dy
- Else
  - y = y + 1
  - p = p + 2*dy - 2*dx
- Plot (x, y)

Step 8: Stop

Decision Parameter Formula

Where:

$$dy = y2 - y1$$
$$dx = x2 - x1$$
$$p_k = 2dy - dx$$

Update rules:

$$p_{k+1} = p_k + 2dy \text{ (if } p < 0)$$
$$p_{k+1} = p_k + 2dy - 2dx \text{ (if } p \geq 0)$$

Working

- Bresenham's uses integer calculations to select the nearest pixel.

Advantages

- Faster than DDA
- No floating-point operations
- High accuracy

Numerical Example

Given points: (2,2) and (10,6)

dx = 8, dy = 4

DDA Points:

(2,2), (3,2.5), (4,3), (5,3.5), (6,4), (7,4.5), (8,5), (9,5.5), (10,6)

Bresenham's Points:

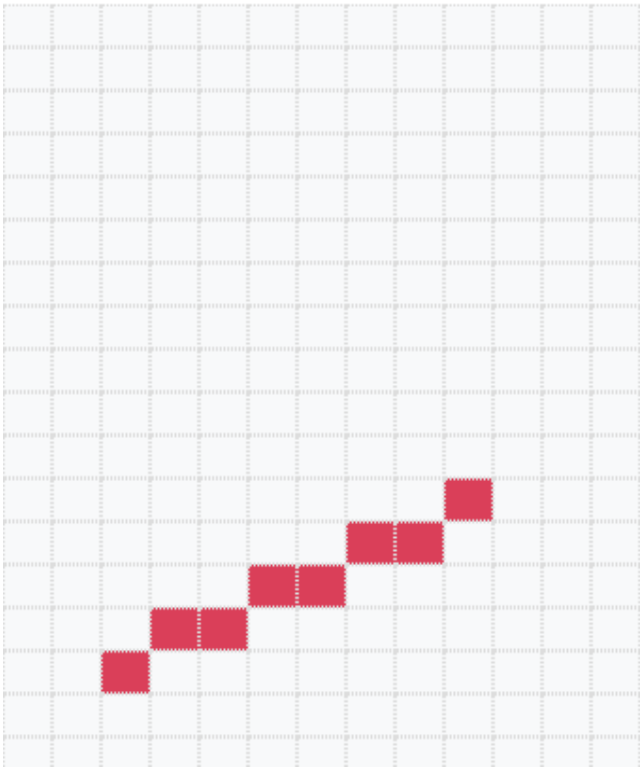
(2,2), (3,2), (4,3), (5,3), (6,4), (7,4), (8,5), (9,5), (10,6)

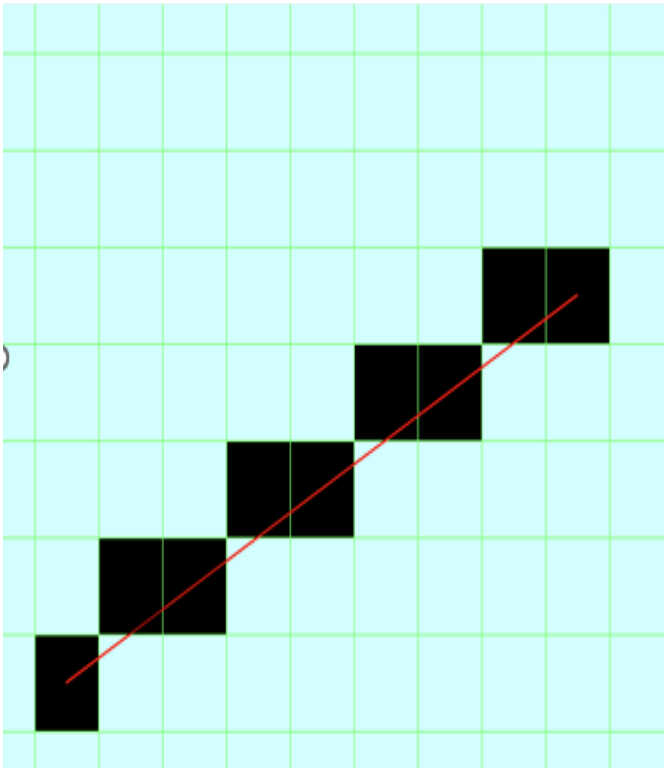
Note: Bresenham's gives better pixel accuracy.

Diagram: Line Drawing Comparison

DDA Algorithm

Bresenham's Alogrithm





8. Performance Table

Parameter	DDA	Bresenham's
Speed	Medium	High
Accuracy	Medium	High
Computation	Floating point	Integer
Efficiency	Low	High

Results and Discussion

- DDA has rounding errors due to decimal values
- Bresenham's provides better accuracy
- Bresenham's is computationally efficient

That's why Bresenham's is better for real-time systems.

Applications

- Computer graphics
- CAD systems
- Video games
- Simulation systems
- Embedded devices

Conclusion

This paper presented a detailed comparison between DDA and Bresenham's algorithms. DDA is simple but less efficient. Bresenham's is faster and more accurate due to integer calculations. That's why Bresenham's is preferred in modern graphics systems.

Future Scope

- Anti-aliasing techniques
- GPU implementation
- Hybrid algorithms
- Artificial intelligence optimization

References

1. Foley, J. D., Van Dam, A., Feiner, S. K., Hughes, J. F., Computer Graphics: Principles and Practice, Addison-Wesley, 2013.
2. Hearn, D., Baker, M. P., Computer Graphics with OpenGL, Pearson Education, 2014.
3. Newman, W. M., Sproull, R. F., Principles of Interactive Computer Graphics, McGraw-Hill, 1979.
4. Rogers, D. F., Procedural Elements for Computer Graphics, McGraw-Hill, 1985.
5. Gonzalez, R. C., Woods, R. E., Digital Image Processing, Pearson, 2018.
6. Bresenham's, J. E., "Algorithm for Computer Control of a Digital Plotter," IBM Systems Journal, Vol. 4, No. 1, pp. 25–30, 1965.
7. Pittaway, M. L. V., "Algorithm for Drawing Ellipses or Hyperbolae with a Digital Plotter," Computer Journal, 1967.
8. Sproull, R. F., "Using Program Transformations to Derive Line-Drawing Algorithms," ACM Transactions on Graphics, Vol. 1, No. 4, 1982.
9. Wu, X., "An Efficient Antialiasing Technique," ACM SIGGRAPH Computer Graphics, 1991.
10. Amanatides, J., Woo, A., "A Fast Voxel Traversal Algorithm for Ray Tracing," Eurographics, 1987.