PROBLEM STATEMENT

To implement the Digital Differential Analyzer (DDA) Line Drawing Algorithm in C using OpenGL, and to draw a line between two points on a 2D plane. The program will also draw the coordinate axes to visualize the line in the context of a Cartesian coordinate system.

THEORY

The Digital Differential Analyzer (DDA) algorithm is a simple and efficient algorithm for drawing lines on raster displays. It works by calculating the intermediate points along the line based on the difference between the start and end points. The DDA algorithm incrementally plots points along the line by adding a fixed amount to the x and y coordinates at each step, which is determined by the slope of the line.

The DDA algorithm can handle any slope and ensures that the line is continuous and visually accurate. The steps are based on the larger difference between the x and y coordinates, ensuring the line is rendered correctly.

ALGORITHM

1. **Initialize**:

    - Start with the initial point (x_{start}, y_{start}).
    - Calculate the differences: $dx = x_{end} - x_{start}$ and $dy = y_{end} - y_{start}$.
    - Determine the number of steps: $steps = \max(|dx|, |dy|)$.
    - Calculate the increments: $xIncrement = \frac{dx}{steps}$ and $yIncrement = \frac{dy}{steps}$.
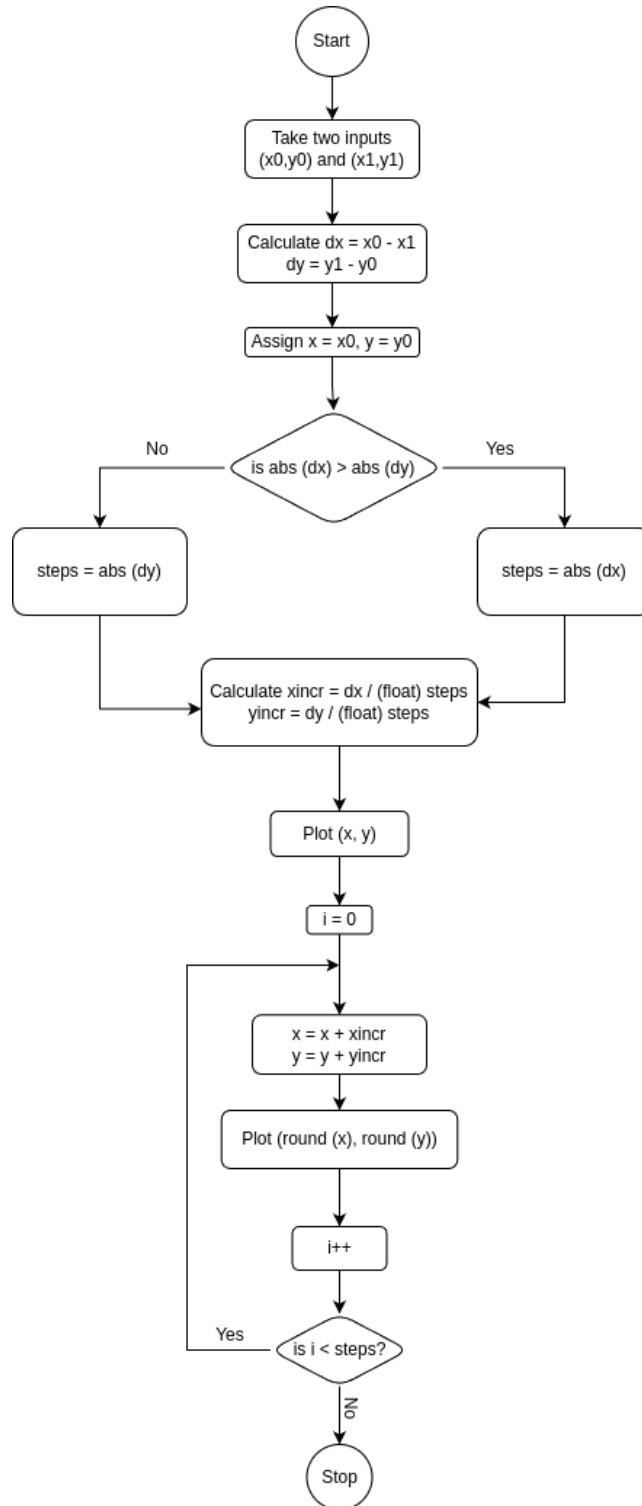
2. **Plotting**:

    - Initialize the current point: $(x, y) = (x_{start}, y_{start})$.
    - Plot the current point.
    - For each step, update the point: $x+ = xIncrement$ and $y+ = yIncrement$, then plot the new point.

3. **Termination**:

    - The loop terminates after all steps are completed.

FLOWCHART

```
                    ( Start )
                        |
                        v
              +-------------------+
              |  Take two inputs  |
              | (x0,y0) and (x1,y1)|
              +-------------------+
                        |
                        v
              +-------------------+
              | Calculate dx = x0 - x1 |
              |     dy = y1 - y0  |
              +-------------------+
                        |
                        v
              +-------------------+
              | Assign x = x0, y = y0 |
              +-------------------+
                        |
                        v
    No          < is abs (dx) > abs (dy) >          Yes
   +-------------                           -------------+
   |                                                     |
   v                                                     v
+-------------------+                         +-------------------+
|  steps = abs (dy) |                         |  steps = abs (dx) |
+-------------------+                         +-------------------+
   |                                                     |
   |            +----------------------------+           |
   +----------->| Calculate xincr = dx / (float) steps |<-+
                |     yincr = dy / (float) steps |
                +----------------------------+
                            |
                            v
                    +-------------+
                    |  Plot (x, y)|
                    +-------------+
                            |
                            v
                        +-------+
                        | i = 0 |
                        +-------+
                            |
                            v
                    +-------------+
                    | x = x + xincr |
                    | y = y + yincr |
                    +-------------+
                            |
                            v
              +----------------------------+
              | Plot (round (x), round (y)) |
              +----------------------------+
                            |
                            v
                        +------+
                        | i++  |
                        +------+
                            |
                            v
        Yes          < is i < steps? >
   +----------------                  
   |                    | No
   |                    v
   |                ( Stop )
```

SAMPLE I/O

Input:

```
Enter the coordinates of the first point (x_start, y_start): -50 -50
Enter the coordinates of the second point (x_end, y_end): 50 50
```

Output: A line is drawn from $(-50, -50)$ to $(50, 50)$ on a window with coordinate axes displayed.

DISCUSSIONS

- **Accuracy**: The DDA algorithm provides accurate and visually continuous lines by incrementally plotting points based on the line's slope.

- **Efficiency**: While not as efficient as Bresenham's algorithm due to floating-point operations, DDA is simple to implement and understand.

- **Applications**: The DDA algorithm is used in various applications where line drawing is required, such as computer graphics, game development, and CAD software.

CONCLUSION

The Digital Differential Analyzer (DDA) Line Drawing Algorithm is an effective method for rendering lines on raster displays. By incrementally plotting points based on the line's slope, the DDA algorithm ensures accurate and visually continuous lines. The implementation using OpenGL demonstrates the algorithm's capability to draw lines between any two points and visualize the coordinate axes.