

Title: To use DDA algorithm to draw a line between given points.

Theory:

To connect two points (x_1, y_1) and (x_2, y_2) in a 2D plane, we need to create a line segment. However, in computer graphics, we cannot directly draw a line between any two coordinate points. Instead, we must compute the coordinates of intermediate points and render a pixel for each of these points using functions like `putpixel(x, y, K)` in C, where (x, y) represents the coordinates and K specifies the color.

In graphics programming, the output screen is treated as a coordinate system. The top-left corner is designated as the coordinate $(0, 0)$, with the y-coordinate increasing as we move downward and the x-coordinate increasing as we move to the right. To create a line segment, we need to calculate the intermediate points, which can be done using a straightforward algorithm known as the DDA (Digital Differential Analyzer) line-generating algorithm.

DDA Algorithm:

- Step 1: Start
- Step 2: Take the coordinates of initial point (x_1, y_1) and the final point (x_2, y_2) .
- Step 3: Find the difference between the points by using the formula:
$$dx = x_2 - x_1;$$
$$dy = y_2 - y_1;$$
- Step 4: Compare the absolute value of the differences to get the number of steps.
If $abs(dx) \geq abs(dy)$
 $steps = dx;$
Else
 $steps = dy;$
- Step 5: Check the slope and update the values of x and y accordingly.
If $(m < 1)$
 $x = x_1 + 1;$
 $y = y_1 + m;$
Else if $(m > 1)$
 $x = x_1 + (1/m);$
 $y = y_1 + m;$
Else
 $x = x_1 + 1;$
 $y = y_1 + 1;$
- Step 6: Repeat the following for $i=0$ to $steps$:
 $putpixel(x_1, y_1, K);$
 $x_1 = x_1 + x;$
 $y_1 = y_1 + y;$
- Step 7: Stop

Source Code:

// DDA algorithm to draw a line.

```
#include <stdio.h>
#include <graphics.h>
#include <math.h>

int main(int argc, char const *argv[])
{
    int x,y,x1,y1,x2,y2,dx,dy,steps,m;
    printf("Created by Kushal Shah\nDDA Algorithm\n");
    printf("Enter the initial point\n");
    scanf("%d %d",&x1,&y1);

    printf("Enter the final point\n");
    scanf("%d %d",&x2,&y2);

    dx=x2-x1;
    dy=y2-y1;

    m=dy/dx;

    if (abs(dx)>=abs(dy))
    {
        steps=dx;
    }
    else
    {
        steps=dy;
    }

    if(m<1){
        x=x1+1;
        y=y1+m;
    }
    else if(m>1){
        x=x1+(1/m);
        y=y1+m;
    }
    else{
        x=x1+1;
        y=y1+1;
    }
}
```

```

    }

    int gd=DETECT,gm;

    initgraph(&gd,&gm,NULL);

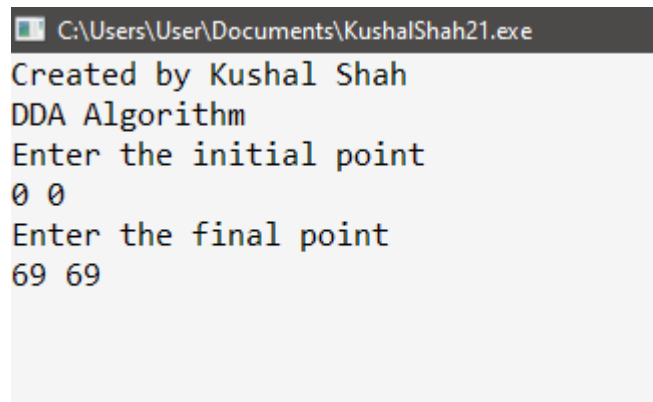
    outtextxy(70,70,"Kushal Shah");
    for (int i = 0; i < steps; i++)
    {
        putpixel(x1,y1,5);
        x1+=x;
        y1+=y;
        delay(100);
    }

    // Concluding the program.

    delay(500000);
    closegraph();
    return 0;
}

```

Output:



```

C:\Users\User\Documents\KushalShah21.exe
Created by Kushal Shah
DDA Algorithm
Enter the initial point
0 0
Enter the final point
69 69

```

Figure 1: Inserting initial and final points

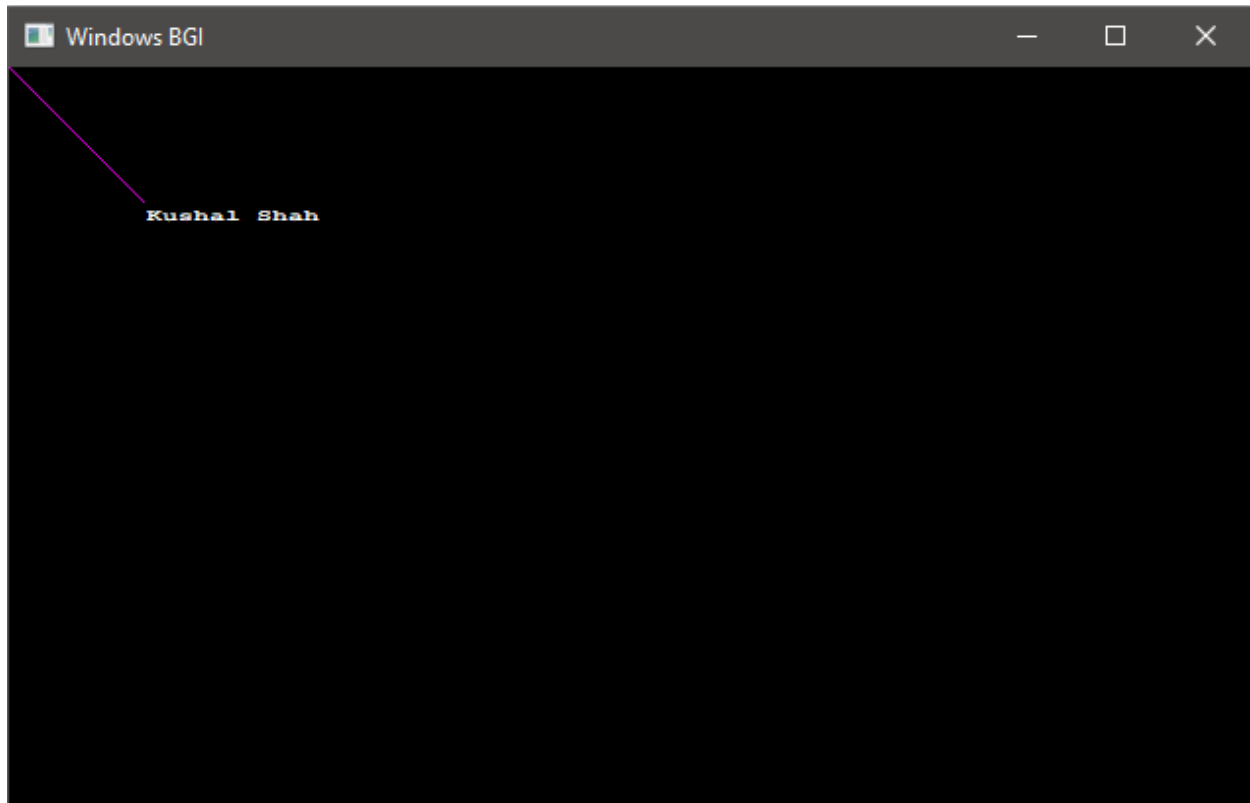


Figure 2: Line drawn using DDA algorithm

Conclusion:

In conclusion, the program demonstrates how to draw a line by calculating and rendering individual pixels using the Digital Differential Analyzer (DDA) algorithm. By leveraging the functions available in the 'graphics.h' header file, we can effectively visualize the line segment between two specified points. This approach highlights the importance of intermediate calculations in computer graphics to achieve smooth and accurate line rendering.