Kathmandu University

Dhulikhel, Kavre

LAB-II
LINE DRAWING ALGORITHM

[Code No:COMP342]

Submitted by

Sudip Bhattarai (11)

Submitted to:-

Mr. Dhiraj Shrestha

Department of Computer Science and Engineering

Submission Date:

9th MAY

Title: Implementation of Line Drawing Algorithm To generate Histogram for different frequencies

Algorithm Implemented:

- **Bresenham Line Drawing (BLA):**

Step1: Start Algorithm
Step2: Input two end points (x_start,y_start) and (x_end,y_end)
Step3: Calculate
dy = y_end-y_start
dx=x_end-x_start
Step4: Check if the line ends at smaller value i.e ending point is on left of starting point
if(x_end>=x_start): sx=1
else: sx=-1
if(y_end>=y_start): sy=1
else: sy=-1
Step5: Set X=x_start and Y=y_start
Step6: Check if the dx>dy to calculate intital decision parameter
if(dx>dy): p=2*dy-dx
else: p=2*dx-dy
Step7:
if(dx>dy):
For each point X_k along the line we calculate:
Save(X,Y)
If(p<0):Y=Y
p= p+2*dy
else: Y=Y+sy
p=p+2*dy-2*dx
X=X +sx

else:
For each point Y_k along the line we calculate:
Save(X,Y)
if(p<0): X=X
p= p+2*dx
else: Y=Y+sy
p=p+2*dx-2*dy
Y=Y+sy

- **Digital Differential Analyzer**

Step1: Start Algorithm

Step2: Input two end points (x_start,y_start) and (x_end,y_end)

Step3: Calculate

dy = y_end-y_start

dx=x_end-x_start

Step4: Set X=x_start and Y=y_start

Step5: Check if the dx>dy to stepsizes

if(|dx|>|dy|): stepsize=|dx|

else: stepsize=|dy|

Step6: Calculate increment in X and Y

X_inc=dx/stepsize

Y_inc=dy/stepsize

Step7: for i=0 to stepsize:

Save(X,Y)

X=X+X_inc

Y=Y+Y_inc

Code Implementation

- **BLA_algorithm.py**

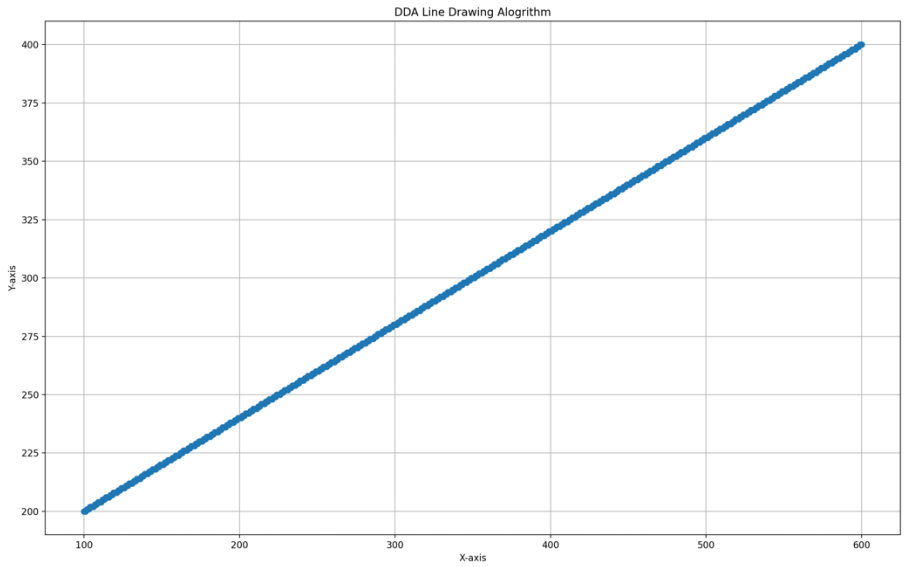
```python
import matplotlib.pyplot as plt
from plotpoints import plot
from draw_histogram import generate_histogram
def BLA(x_start ,y_start,x_end,y_end):
    points=[]  #for storing the calculated points
    dy=y_end-y_start
    dx=x_end-x_start

    #for checking if the points start from  right i.e end point is smaller then starting
pointif x_end >= x_start:
        sx = 1
    else:
        sx = -1
    if y_end >= y_start:
        sy = 1
    else:
        sy = -1
    x=x_start
    y=y_start
    if dx>dy:
        p=2*dy-dx
    else:
        p=2*dx-dy


    if dx>dy:
        while x!=x_end:
            points.append((int(round(x)), int(round(y))))
            if p<0:
                y=y
                p=p+2*dy
            else:
                y=y+sy
                p=p+2*dy-2*dx
            x=x+sx
    else:
        while y!=y_end:
            points.append((int(round(x)), int(round(y))))
            if p<0:
                x=x
                p=p+2*dx
            else:
                x=x+1
                p=p+2*dx-2*dy
            y=y+sy
    points.append((x_end,y_end))
    return points

def main():
    x_start = int(input("Enter the x-coordinate of the starting point(integer): "))
    y_start = int(input("Enter the y-coordinate of the starting point(integer): "))
    x_end = int(input("Enter the x-coordinate of the ending point(integer): "))
    y_end = int(input("Enter the y-coordinate of the ending point(integer): "))

    #BLA LINE DRAWING GRID
    all_points=BLA(x_start=x_start,y_start=y_start,x_end=x_end,y_end=y_end)
    all_x_coordinates=[point[0] for point in all_points]
    all_y_coordinates=[point[1] for point in all_points]
    plot(all_x_coordinates,all_y_coordinates,'BLA Line Drawing Algorithm')


    #BLA Histogram
    histogram_data = [20, 30, 75, 100, 50, 45, 95]
    generate_histogram(histogram_data,BLA)


if __name__ == "__main__":
    main()
```

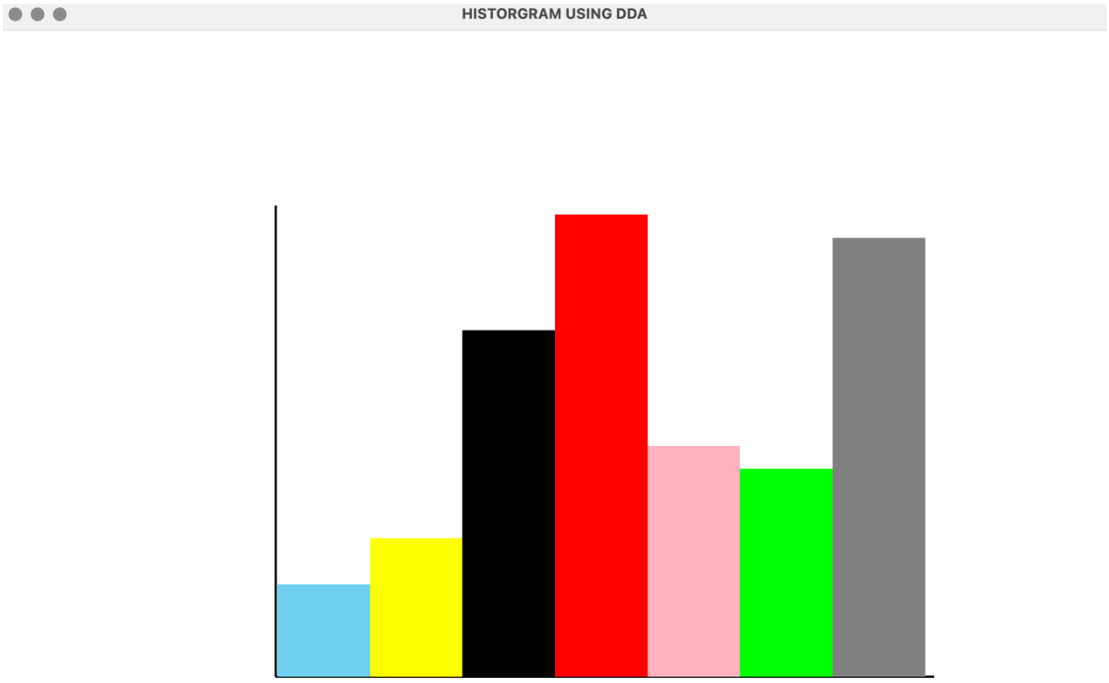
- **DDA_algorithm.py**

```python
from plotpoints import plot
from draw_histogram import generate_histogram
def DDA(x_start ,y_start,x_end,y_end):
    x=x_start
    y=y_start
    dx=x_end-x_start
    dy=y_end-y_start
    if(abs(dx)>abs(dy)):
        stepsize=abs(dx)
    else:
        stepsize=abs(dy)

    x_inc=dx/stepsize
    y_inc=dy/stepsize


    points = []
    for i in range(stepsize):
        points.append((int(round(x)), int(round(y))))
        x=x+x_inc
        y=y+y_inc
    points.append((x_end,y_end))

    return points

def main():
    x_start = int(input("Enter the x-coordinate of the starting point(integer):
")) y_start = int(input("Enter the y-coordinate of the starting point(integer):
")) x_end = int(input("Enter the x-coordinate of the ending point(integer): "))
    y_end = int(input("Enter the y-coordinate of the ending point(integer): "))
    #DDA LINE DRAWING
    all_points=DDA(x_start=x_start,y_start=y_start,x_end=x_end,y_end=y_end)
    all_x_coordinates=[point[0] for point in all_points]
    all_y_coordinates=[point[1] for point in all_points]

    plot(all_x_coordinates,all_y_coordinates,'DDA Line Drawing Alogrithm')


    #DDA histogram
    histogram_data = [20, 30, 75, 100, 50, 45, 95]
    generate_histogram(histogram_data,DDA,"DDA")

if __name__ == "__main__":
    main()
```

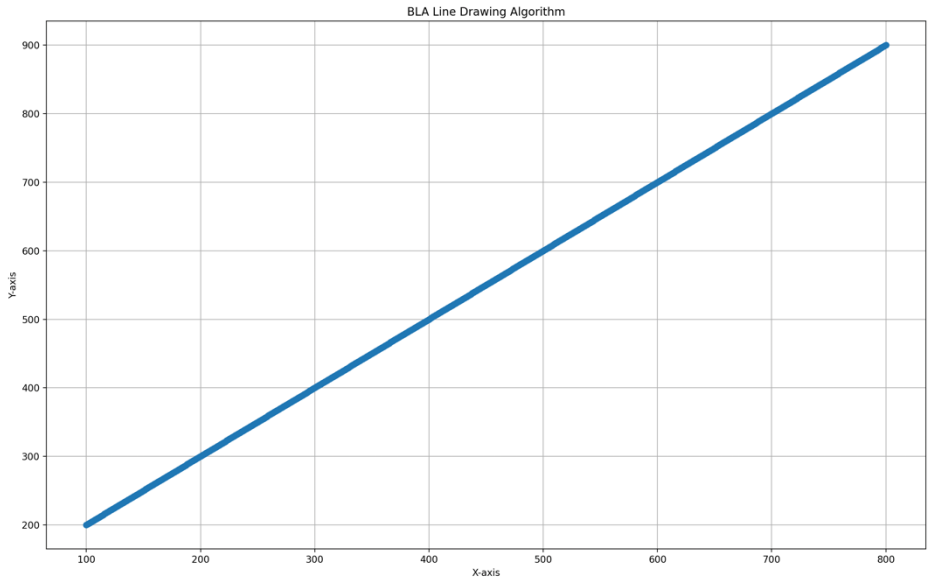
Output:

- **Using DDA Algorithm**

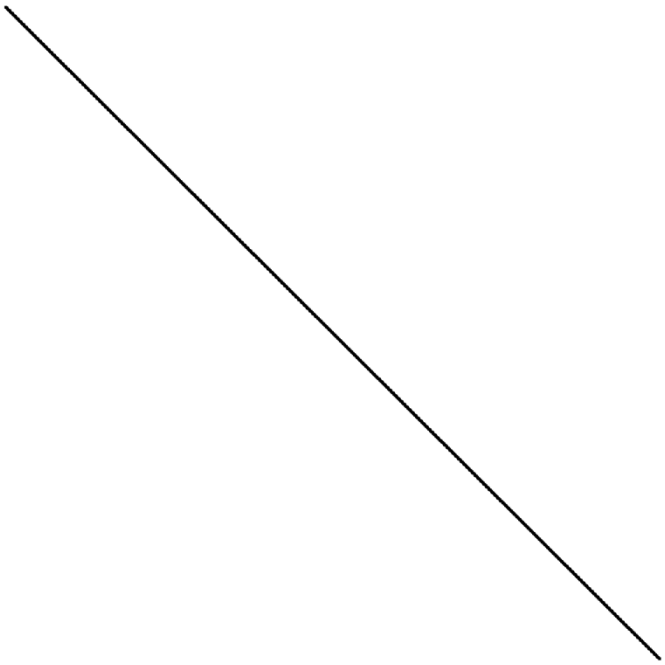



HISTORGRAM USING DDA

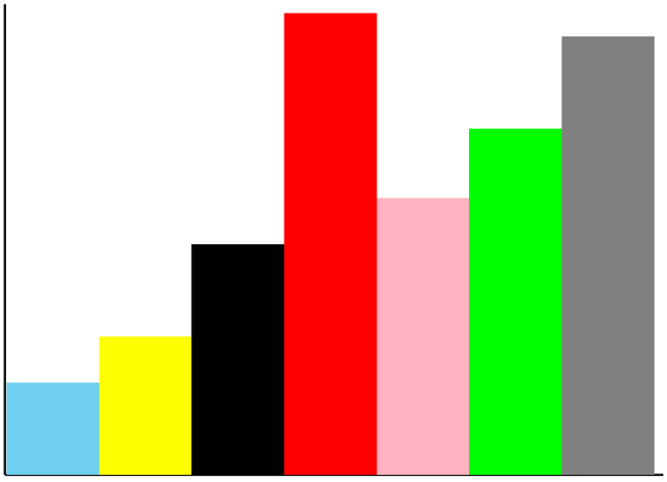
- **Using BLA**


BLA Line Drawing Algorithm
900
800
700
600
500
400
300
200
Y-axis
100
200
300
400
500
600
700
800
X-axis

LINE USING BLA



HISTORGRAM USING BLA



Conclusion:

In conclusion we can see that the basic line drawing has been implemented using python open gl. We implemented BLA, which efficiently calculates points along a straight line between two given points. The BLA algorithm ensures that the line stays on the grid's straight path and generates points accurately. We implemented the DDA algorithm, which calculates points along a straight line but takes a different approach than BLA. DDA calculates points based on the slope of the line, making it ideal for line drawing in continuous spaces.

We extended our implementation of the above BLA and DDA to draw bar of histogram which output and code snippet is shown above.