Computer Graphics

Line Drawing Techniques

What is Point

- A point(a position in a plane) is specified by a set of ordered pair(x, y) where x is the horizontal distance from the origin and y is the vertical distance from the origin.
- Two points will specify a line.

Line Cont…

- Lines are described by line equation such that if a point satisfies the line equation, then the point is on the line
- Given two points, in Euclidean geometry, one can always find exactly one line that passes through the two points.
- The slope-intercept form of the line, slope m is change in y divided by change in x and intercept b is the height at which the line crossed the y-axis.i.e (0, b)

$$y = mx + b$$

Line Cont…

A line segment is specified by its two end points. i.e (x_1,y_1) and (x_2, y_2). Any point will be on the line , if it satisfies the following conditions.

- $y_3 = mx_3 + b$
- $Min(x_1, x_2) <= x_3 <= max(x_1, x_2)$
- $Min(y_1, y_2) <= y_3 <= max(y_1, y_2)$

Line Cont…

A line may have three forms with respect to slope:

- slope = 1
- slope > 1 // Sharp Slope
- slope < 1 // Gentle Slope

Line Cont…

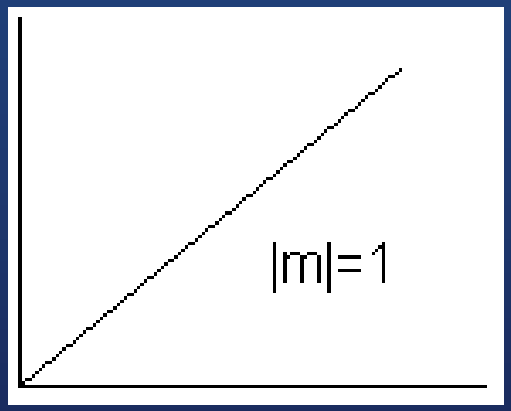

figure (a)

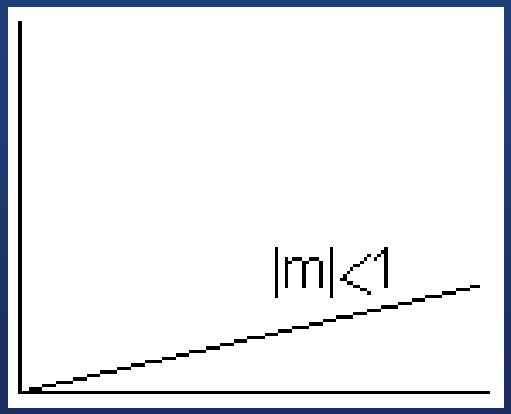

figure (b)

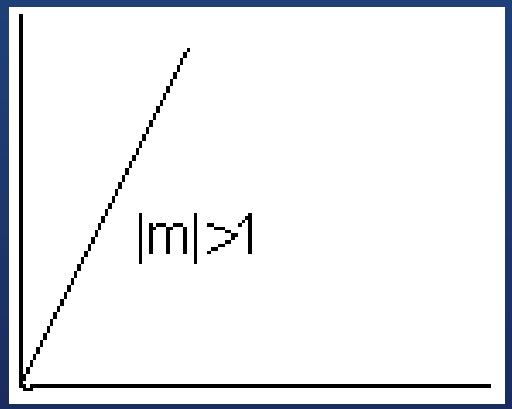

figure (c)

Line Drawing Techniques

There are three techniques to be discussed to draw a line involving different time complexities that will be discussed along. These techniques are:

- Incremental line algorithm
- DDA line algorithm
- Bresenham line algorithm

Incremental Line Algorithm

This algorithm exploits simple line equation

$$y = m x + b$$

where $m = dy / dx$

and $b = y – m x$

now check if

$|m| < 1$ then

$x = x + 1$ whereas

$y = m x + b$

why to check |m|

suppose a line has points

p1 (10, 10) – p2 (20, 18)

dy = y2 – y1 = 18 – 10 = 8

dx = x2 – x1 = 20 – 10 = 10

This means that there will be 10 pixels on the line in which for x-axis there will be distance of 1 between each pixel and for y-axis the distance will be 0.8.

Now consider the reverse case

suppose a line has points

p1 (10, 10) , p2 (16, 20)

dy = y2 – y1 = 20 – 10 = 10

dx = x2 – x1 = 16 – 10 = 6

This means that there will be 10 pixels on the line in which for x-axis there will be distance of 0.6 between each pixel and for y-axis the distance will be 1.

Now sum-up all discussion in algorithm to fully understand.

The algorithm will take two points P1 and P2 and draw line between them whereas each point consists of x,y coordinates.

dx = p2.x – p1. x

dy = p2.y – p1. y

m = dy / dx

x = p1.x

y = p1.y

b = y – m * x

```
    if |m| < 1
    for counter = p1.x to p2.x
            drawPixel (x, y)
            x = x + 1
            y = m * x + b
else
    for counter = p1.y to p2.y
            drawPixel (x, y)
            y = y + 1
            x = ( y – b ) / m
```

Discussion on algorithm:

- quite simple and easy to understand
- but involves a lot of mathematical calculations 24

We have another algorithm that works fine in all directions and involves less calculations; mostly only additions.

DDA Algorithm

DDA abbreviated for digital differential analyzer has a very simple technique.

Find difference, dx and dy as:

$$dy = y2 – y1$$

$$dx = x2 – x1$$

```
if |dx| > |dy| then
        step = |dx|
else
        step = |dy|
```

Now very simple to say that step is the total number of pixels required for a line.

Next step is to find xIncrement and yIncrement:

xIncrement = dx/step

yIncrement = dy/step

Next a loop is required that will run ‘step’ times.

In the loop drawPixel and add xIncrement to x1 and yIncrement to y1.

Now sum-up all above in the algorithm:

DDA_Line (Point p1, Point p2)

```
dx = p2.x – p1. x
dy = p2.y – p1. y
x1=p1.x
y1=p1.y
if |dx| > |dy| then
        step = |dx|
else
        step = |dy|
```

```
xIncrement = dx/step
yIncrement = dy/step
for counter = 1 to step
        drawPixel (x1, y1)
        x1 = x1 + xIncrement
        y1 = y1 + yIncrement
```

Criticism on Algorithm:

Use of floating point calculation

An algorithm based on integer type calculations is likely to be more efficient

Therefore, after some effort, finally we came up with an algorithm called “Bresenham Line Drawing Algorithm” which would be discussed next.