**Universidade Federal de Roraima**
**Centro de Ciências e Tecnologia**
**Departamento de Ciência da Computação**
**Disciplina de Computação Gráfica**
**Professor: Dr. Luciano Ferreira Silva**
**Aluno: Felipe Derkian de Sousa Freitas – 1201424418**

# Rasterização de Semirretas com Método Analítico, DDA e Bresenham

**Boa Vista, 17 de Outubro de 2018**

# Sumário

# Algoritmo método analítico

O algoritmo método analítico é um algoritmo usado para rasterização de semirretas onde dados dois pontos p1(x1,y1) e p2(x2,y2), ele rasteriza na escolha dos melhores pixel usado para isso como base a equação da reta que é dada por *y=mx+b* . onde *m=(y2-y1)/(x2-x1)*, ou seja *m=(delta_y/delta_x)* e *b=y1-mx1*. Sabendo-se disso, basta substituir na fórmula da equação da reta para termos os pontos.

## Pontos fortes do método analítico

O método analítico é um excelente algoritmo para rasterizar semirretas que fiquem no primeiro octante que vai de 0° a 45°. Pois para retas com inclinação abaixo de 45° fica muito satisfatório por exemplo os pontos p1(0,0) e p2(5,2) onde a reta não tem muita inclinação.

## Pontos fracos do método analítico

O método analítico quando utilizado para retas com mais de 45°, ou seja segundo octante, começa a ter falhas na reta ficando impraticável o uso do algoritmo, como no caso de teste usando os pontos p1(0,0) e p2(2,5), onde a reta tem muita inclinação e acontece falhas por conta dos arredondamentos de ponto flutuante.

## Desempenho do algoritmo método analítico

O algoritmo é pouco eficiente, pois usa muito ponto flutuante e arredondamentos que são operações complexas e custosas para o computador processar. Para desenhar uma reta pode ser uma boa opção, mas quando precisamos rasterizar centenas de linhas passa não ser interessante visto que em certos cenários existem centenas de linhas.

## Pseudo código

Função metodoAnalitico recebe os pontos x1,y1,x2,y2:

```
se x1 for igual a x2 faça
    para y de y1 ate y2
        setPixel(x1,y, cor)

senão
    m = delta y / delta x
    b = y2-m*x2
    para x de x1 ate x2
        y = m*x+b
        setPixel(x,y,cor)
```

## Simulação com código em linguagem C

```c
void metodoAnalitico(int matrix[TAM_MATRIX][TAM_MATRIX], int x1, int y1, int x2, int y2){
    int y;
    if(x1==x2){
        for(y=y1 ; y<=y2; y++)
            matrix[x1][y] = BORDA;
    }else{
        double m = (double) (y2-y1) / (x2-x1);
        double b = (double) y2-m*x2;
        int x;
        for(x=x1 ; x<=x2 ; x++){
            double k = (double) m*x+b;
            y = round(k);
            matrix[x][y] = BORDA;
        }
    }
}
```

## Explicação do código feito em C

O código foi feito simulando em uma matriz, pois o código foi desenvolvido em Linux e a biblioteca graphics*.h* só funciona com MS-*DOS*, ou seja, com Windows e um compilador especial que só funciona em no sistema operacional Windows.

## Coordenada de pontos de teste

Usaremos os seguintes conjuntos de pontos para comparar com os algoritmos de rasterização de semirretas.

Conjunto 1: P1(0,0) e P2(5,2).

Conjunto 2: P1(0,0) e P2(2,5).

Conjunto 3: P1(0,0) e P2(3,20).

Conjunto 4: P1(0,0) e P2(20,3).

Conjunto 5: P1(0,0) e P2(20,15).

Conjunto 6: P1(0,0) e P2(15,20).

Conjunto 7: P1(15,10) e P2(2,3).

# Programa em execução com resultados

```
                Algoritmo metodo analitico
OBS: Matriz tamanho maximo 30x30

Informe as coordenadas de x1 e y1.
>> 0 0
Informe as coordenadas de x2 e y2.
>> 5 2

100000000000000000000000000000
100000000000000000000000000000
010000000000000000000000000000
010000000000000000000000000000
001000000000000000000000000000
001000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
```

*Figura 1: P1(0,0) e P2(5,2)*

```
                Algoritmo metodo analitico
OBS: Matriz tamanho maximo 30x30

Informe as coordenadas de x1 e y1.
>> 0 0
Informe as coordenadas de x2 e y2.
>> 2 5

100000000000000000000000000000
000100000000000000000000000000
000001000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
```

*Figura 2: P1(0,0) e P2(5,2)*

```
                Algoritmo metodo analitico
OBS: Matriz tamanho maximo 30x30

Informe as coordenadas de x1 e y1.
>> 0 0
Informe as coordenadas de x2 e y2.
>> 3 20

100000000000000000000000000000
000000010000000000000000000000
000000000000010000000000000000
000000000000000000001000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
```

*Figura 3: P1(0,0) e P2(3,20)*

```
                Algoritmo metodo analitico
OBS: Matriz tamanho maximo 30x30

Informe as coordenadas de x1 e y1.
>> 0 0
Informe as coordenadas de x2 e y2.
>> 20 3

100000000000000000000000000000
100000000000000000000000000000
100000000000000000000000000000
100000000000000000000000000000
010000000000000000000000000000
010000000000000000000000000000
010000000000000000000000000000
010000000000000000000000000000
010000000000000000000000000000
010000000000000000000000000000
001000000000000000000000000000
001000000000000000000000000000
001000000000000000000000000000
001000000000000000000000000000
001000000000000000000000000000
001000000000000000000000000000
001000000000000000000000000000
000100000000000000000000000000
000100000000000000000000000000
000100000000000000000000000000
000100000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
```

*Figura 4: P1(0,0) e P2(20,3)*

```
                Algoritmo metodo analitico
OBS: Matriz tamanho maximo 30x30

Informe as coordenadas de x1 e y1.
>> 0 0
Informe as coordenadas de x2 e y2.
>> 20 15

100000000000000000000000000000
010000000000000000000000000000
001000000000000000000000000000
001000000000000000000000000000
000100000000000000000000000000
000010000000000000000000000000
000001000000000000000000000000
000001000000000000000000000000
000000100000000000000000000000
000000010000000000000000000000
000000001000000000000000000000
000000001000000000000000000000
000000000100000000000000000000
000000000010000000000000000000
000000000001000000000000000000
000000000001000000000000000000
000000000000100000000000000000
000000000000010000000000000000
000000000000001000000000000000
000000000000001000000000000000
000000000000000100000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
```

*Figura 5: P1(0,0) e P2(20,15)*

```
                Algoritmo metodo analitico
OBS: Matriz tamanho maximo 30x30

Informe as coordenadas de x1 e y1.
>> 0 0
Informe as coordenadas de x2 e y2.
>> 15 20

100000000000000000000000000000
010000000000000000000000000000
000100000000000000000000000000
000010000000000000000000000000
000001000000000000000000000000
000000010000000000000000000000
000000001000000000000000000000
000000000100000000000000000000
000000000001000000000000000000
000000000000100000000000000000
000000000000010000000000000000
000000000000000100000000000000
000000000000000010000000000000
000000000000000001000000000000
000000000000000000010000000000
000000000000000000001000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
```

*Figura 6: P1(0,0) e P2(15,20)*

```
                Algoritmo metodo analitico
OBS: Matriz tamanho maximo 30x30

Informe as coordenadas de x1 e y1.
>> 15 10
Informe as coordenadas de x2 e y2.
>> 2 3

000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
```

*Figura 7: P1(15,10) e P2(2,3)*

# Algoritmo DDA (Analisador Diferencial Digital)

O algoritmo DDA analisador diferencial digital, veio para melhorar os resultados do método analítico como a falha que acontece quando a inclinação da reta é maior que 45°. Ele usa técnica baseada no calculo de delta X e delta Y, onde *m=delta_y/delta_X, delta_y=m*delta_X e delta_X=delta_y/m.*

*A ideia é parecida com a implementada no método analítico quando se trabalho entre 0 a 45°, mas quando estamos trabalhando entre 45 a 90° modifica-se o incremento para delta_y e calcula-se os sucessivos valores para x, com isso resolve-se o problema das retas falhadas com inclinação maior que 45°.*

## *Pontos fortes do DDA*

*Os pontos fortes do DDA em relação ao método analítico é que ele resolve o problema das semirretas com inclinação maiores que 45°, onde no método analítico ficavam falhadas e com um simples ajuste trocando quem incrementa com os sucessivos valores conseguiram resolver o problema das falhas.*

## Pontos fracos do DDA

Apesar de ter solucionado o problema de semirretas com  inclinação maior que 45°, ainda continua com operações custosas e pesadas para o computador realizar milhões de vezes por segundo como ponto flutuante e arredondamento.

## Desempenho do algoritmo DDA

O desempenho é melhor que do método analítico, mas ainda utiliza aritmética com ponto flutuante, arredondamentos e em grande escala acaba sendo lento pois são operações custosas para o computador processar.

## Pseudo Código

```
Função DDA recebe os pontos x1, y1, x2, y2:
    verifica-se qual é maior delta_x ou delta_y faça:
        incremento = delta_y/delta_x
        y = y1
        para x de x1 até x2 faça:
            setPixel(x,y,cor)
            y = y + incremento
    senão faça:
        incremento = delta_x/delta_y
        x=x1
        para y de y1 até y2 faça:
            setPixel(x,y,cor)
            x = x + incremento
```

## Código DDA implementado em linguagem C

```
void algoritmo_dda(int matrix[TAM_MATRIX][TAM_MATRIX], int x1, int y1, int x2, int y2){
    int abs_x = x2-x1;
    abs_x = abs(abs_x); //pega o valor abs do delta x
    int abs_y = y2-y1;
    abs_y = abs(abs_y); //pega o valor abs do delta y

    double incremento=0, totalIncrementos=0;
    int x , y;

    if(abs_x > abs_y){
        incremento = (double) (y2-y1) / (x2-x1);
        y = y1;
        for(x=x1 ; x<=x2 ; x++){
            //simula ligar o pixel
            matrix[x][y] = BORDA;
            totalIncrementos = totalIncrementos + incremento;
            y = round(totalIncrementos);
        }
    }else{
```

```
        incremento = (double) (x2-x1) / (y2-y1);

        x = x1;

        for(y=y1 ; y<=y2 ; y++){

            //simula ligar o pixel
            matrix[x][y] = BORDA;

            totalIncrementos = totalIncrementos + incremento;

            x = round(totalIncrementos);

        }

    }

}
```

## Coordenada de pontos de teste

Usaremos os seguintes conjuntos de pontos para comparar com os algoritmos de rasterização de semirretas.

Conjunto 1: P1(0,0) e P2(5,2).

Conjunto 2: P1(0,0) e P2(2,5).

Conjunto 3: P1(0,0) e P2(3,20).

Conjunto 4: P1(0,0) e P2(20,3).

Conjunto 5: P1(0,0) e P2(20,15).

Conjunto 6: P1(0,0) e P2(15,20).

Conjunto 7: P1(15,10) e P2(2,3).

## Programa em execução com resultados

```
                Algoritmo DDA
OBS: tamanho da matrix 30x30

Informe o valor de x1 e y1.
>> 0 0
Informe o valor de x2 e y2.
>> 5 2

INCREMETO = 0.400000
100000000000000000000000000000
100000000000000000000000000000
010000000000000000000000000000
010000000000000000000000000000
001000000000000000000000000000
001000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
```

*Figura 8: P1(0,0) e P2(5,2)*

```
                 Algoritmo DDA
OBS: tamanho da matrix 30x30

Informe o valor de x1 e y1.
>> 0 0
Informe o valor de x2 e y2.
>> 2 5

110000000000000000000000000000
001100000000000000000000000000
000011000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
```

*Figura 9: P1(0,0) e P2(2,5)*

```
                Algoritmo DDA
OBS: tamanho da matrix 30x30

Informe o valor de x1 e y1.
>> 0 0
Informe o valor de x2 e y2.
>> 3 20

111100000000000000000000000000
000011111110000000000000000000
000000000001111110000000000000
000000000000000001111000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
```

*Figura 10: P1(0,0) e P2(3,20)*

```
                Algoritmo DDA
OBS: tamanho da matrix 30x30

Informe o valor de x1 e y1.
>> 0 0
Informe o valor de x2 e y2.
>> 20 3

INCREMETO = 0.150000
100000000000000000000000000000
100000000000000000000000000000
100000000000000000000000000000
100000000000000000000000000000
010000000000000000000000000000
010000000000000000000000000000
010000000000000000000000000000
010000000000000000000000000000
010000000000000000000000000000
010000000000000000000000000000
010000000000000000000000000000
001000000000000000000000000000
001000000000000000000000000000
001000000000000000000000000000
001000000000000000000000000000
001000000000000000000000000000
001000000000000000000000000000
000100000000000000000000000000
000100000000000000000000000000
000100000000000000000000000000
000100000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
```

*Figura 11: P1(0,0) e P2(20,3)*

```
                Algoritmo DDA
OBS: tamanho da matrix 30x30

Informe o valor de x1 e y1.
>> 0 0
Informe o valor de x2 e y2.
>> 20 15

INCREMETO = 0.750000
100000000000000000000000000000
010000000000000000000000000000
001000000000000000000000000000
001000000000000000000000000000
000100000000000000000000000000
000010000000000000000000000000
000001000000000000000000000000
000001000000000000000000000000
000000100000000000000000000000
000000010000000000000000000000
000000001000000000000000000000
000000001000000000000000000000
000000000100000000000000000000
000000000010000000000000000000
000000000001000000000000000000
000000000001000000000000000000
000000000000100000000000000000
000000000000010000000000000000
000000000000001000000000000000
000000000000001000000000000000
000000000000000100000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
```

*Figura 12: P1(0,0) e P2(20,15)*

```
                Algoritmo DDA
OBS: tamanho da matrix 30x30

Informe o valor de x1 e y1.
>> 0 0
Informe o valor de x2 e y2.
>> 15 20

100000000000000000000000000000
010000000000000000000000000000
001100000000000000000000000000
000010000000000000000000000000
000001000000000000000000000000
000000110000000000000000000000
000000001000000000000000000000
000000000100000000000000000000
000000000011000000000000000000
000000000000100000000000000000
000000000000010000000000000000
000000000000001100000000000000
000000000000000010000000000000
000000000000000001000000000000
000000000000000000110000000000
000000000000000000001000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
```

*Figura 13: P1(0,0) e P2(15,20)*

```
                Algoritmo DDA
OBS: tamanho da matrix 30x30

Informe o valor de x1 e y1.
>> 15 10
Informe o valor de x2 e y2.
>> 2 3

INCREMETO = 0.538462
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
```

*Figura 14: P1(15,10) e P2(2,3)*

# Algoritmo Bresenham para semirretas

A ideia do algoritmo de Bresenham é analisar qual o próximo pixel que será ligado. Como pixel é discreto de um ponto p1 para um ponto p2 ele analisa qual pixel vizinho ele deve ligar por exemplo o (x+1,y) ou (x+1,y+1). Ou seja, fazendo uma análise com a inclinação da reta tirando a diferença entre os pixel ao ponto da semirreta, ele consegue determinar qual pixel acender ou não.

## Pontos fortes do algoritmo de Bresenham

O algoritmo de Bresenham é muito bom pois é leve e consegui fazer linha perfeitas por usar uma técnica que analisa o próximo pixel da vizinhança ao qual deve acender. Diferente dos outros algoritmos que usavam métodos matemáticos de geometria analítica e arredondamentos.

Um ponto que merece destaque para o Bresenham, é que ele consegue rasterizar tanto p1 > p2 quanto p2 > p1. Nos outros algoritmos não conseguiam o mesmo resultado.

## Desempenho do algoritmo de Bresenham

O algoritmo de Bresenham, é muito eficiente pois trabalha apenas com aritmética de inteiros e não precisa de arredondamentos, ou seja apenas operações que o computador sabe fazer muito bem e com eficiência. Sendo muito mais eficiente que os outros algoritmos por essas características.

## Pseudo Código

```
Função Bresenham recebe x1,y1,x2,y2:
    tira o delta_x, delta_y
    y=y1
    parâmentro = 2*delta_y-delta_x
    para x de x1 até x2 faça:
        setPixel(x,y,cor)
        se parâmetro for maior ou igual 0 faça:
            y = y+1
            parâmetro =  parâmetro + 2*(delta_y – delta_x)

        senão faça:
             parâmetro =  parâmetro + 2*delta_y
```

## Código Bresenham implementado em linguagem C

```
void bresenham_line(int matrix[TAM_MATRIX][TAM_MATRIX], int x0,int y0, int x1, int y1){
    int delta_x = abs(x0-x1);
    int delta_y = abs(y0-y1);
    int parametro = 2*delta_y-delta_x;
    int parametro2 = 2*delta_y;
    int xy2 = 2*(delta_y-delta_x);
    int xf,x,y;
    if(x0 > x1){
        x = x1;
        y = y1;
        xf = x0;
        matrix[x][y] = BORDA;

    }else{
        x = x0;
        y = y0;
        xf = x1;
        matrix[x][y] = BORDA;
    }
    while(x < xf){
            x = x+1;
            if(parametro < 0){
                parametro = parametro + parametro2;
            }else{
                y = y+1;
                parametro = parametro + xy2;
            }

            matrix[x][y] = BORDA;
```

}

}

## Coordenada de pontos de teste

Usaremos os seguintes conjuntos de pontos para comparar com os algoritmos de rasterização de semirretas.

Conjunto 1: P1(0,0) e P2(5,2).

Conjunto 2: P1(0,0) e P2(2,5).

Conjunto 3: P1(0,0) e P2(3,20).

Conjunto 4: P1(0,0) e P2(20,3).

Conjunto 5: P1(0,0) e P2(20,15).

Conjunto 6: P1(0,0) e P2(15,20).

Conjunto 7: P1(15,10) e P2(2,3).

## Programa em execução com resultados

```
                Algoritmo Bresenham semiretas
OBS: tamanho da matrix 30x30

Informe o valor de x1 e y1:
>> 0 0
Informe o valor de x2 e y2:
>> 5 2
100000000000000000000000000000
100000000000000000000000000000
010000000000000000000000000000
010000000000000000000000000000
001000000000000000000000000000
001000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
```

*Figura 15: P1(0,0) e P2(5,2)*

```
                Algoritmo Bresenham semiretas
OBS: tamanho da matrix 30x30

Informe o valor de x1 e y1:
>> 0 0
Informe o valor de x2 e y2:
>> 2 5
100000000000000000000000000000
010000000000000000000000000000
001000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
```

*Figura 16: P1(0,0) e P2(2,5)*

```
                Algoritmo Bresenham semiretas
OBS: tamanho da matrix 30x30

Informe o valor de x1 e y1:
>> 0 0
Informe o valor de x2 e y2:
>> 3 20
100000000000000000000000000000
010000000000000000000000000000
001000000000000000000000000000
000100000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
```

*Figura 17: P1(0,0) e P2(3,20)*

```
                Algoritmo Bresenham semiretas
OBS: tamanho da matrix 30x30

Informe o valor de x1 e y1:
>> 0 0
Informe o valor de x2 e y2:
>> 20 3
100000000000000000000000000000
100000000000000000000000000000
100000000000000000000000000000
100000000000000000000000000000
010000000000000000000000000000
010000000000000000000000000000
010000000000000000000000000000
010000000000000000000000000000
010000000000000000000000000000
010000000000000000000000000000
001000000000000000000000000000
001000000000000000000000000000
001000000000000000000000000000
001000000000000000000000000000
001000000000000000000000000000
001000000000000000000000000000
001000000000000000000000000000
000100000000000000000000000000
000100000000000000000000000000
000100000000000000000000000000
000100000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
```

*Figura 18: P1(0,0) e P2(20,3)*

```
                Algoritmo Bresenham semiretas
OBS: tamanho da matrix 30x30

Informe o valor de x1 e y1:
>> 0 0
Informe o valor de x2 e y2:
>> 20 15
100000000000000000000000000000
010000000000000000000000000000
001000000000000000000000000000
001000000000000000000000000000
000100000000000000000000000000
000010000000000000000000000000
000001000000000000000000000000
000001000000000000000000000000
000000100000000000000000000000
000000010000000000000000000000
000000001000000000000000000000
000000001000000000000000000000
000000000100000000000000000000
000000000010000000000000000000
000000000001000000000000000000
000000000001000000000000000000
000000000000100000000000000000
000000000000010000000000000000
000000000000001000000000000000
000000000000001000000000000000
000000000000000100000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
```

*Figura 19: P1(0,0) e P2(20,15)*

```
                Algoritmo Bresenham semiretas
OBS: tamanho da matrix 30x30

Informe o valor de x1 e y1:
>> 0 0
Informe o valor de x2 e y2:
>> 15 20
100000000000000000000000000000
010000000000000000000000000000
001000000000000000000000000000
000100000000000000000000000000
000010000000000000000000000000
000001000000000000000000000000
000000100000000000000000000000
000000010000000000000000000000
000000001000000000000000000000
000000000100000000000000000000
000000000010000000000000000000
000000000001000000000000000000
000000000000100000000000000000
000000000000010000000000000000
000000000000001000000000000000
000000000000000100000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
```

*Figura 20: P1(0,0) e P2(15,20)*

```
                Algoritmo Bresenham semiretas
OBS: tamanho da matrix 30x30

Informe o valor de x1 e y1:
>> 15 10
Informe o valor de x2 e y2:
>> 2 3
000000000000000000000000000000
000000000000000000000000000000
000100000000000000000000000000
000010000000000000000000000000
000010000000000000000000000000
000001000000000000000000000000
000001000000000000000000000000
000000100000000000000000000000
000000100000000000000000000000
000000010000000000000000000000
000000010000000000000000000000
000000001000000000000000000000
000000001000000000000000000000
000000000100000000000000000000
000000000100000000000000000000
000000000010000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
000000000000000000000000000000
```

*Figura 21: P1(15,10) e P2(2,3)*

# Conclusão

Vimos como o método analítico tinham suas falhas, o DDA resolveu as falhas que o método analítico apresenta, mas ainda era muito pesado por precisar de operações custosas para o computador como aritmética de ponto flutuante e arredondamentos. Então, Bresenham criou seu algoritmo que além de muito inteligente é muito eficiente computacionalmente, e resolve os problemas dos outros algoritmos anteriores.

# Referências

Hetem Junior, Annibal Computação gráfica/Annibal Hetem Junior. - Rio de janeiro : LTC, 2006.