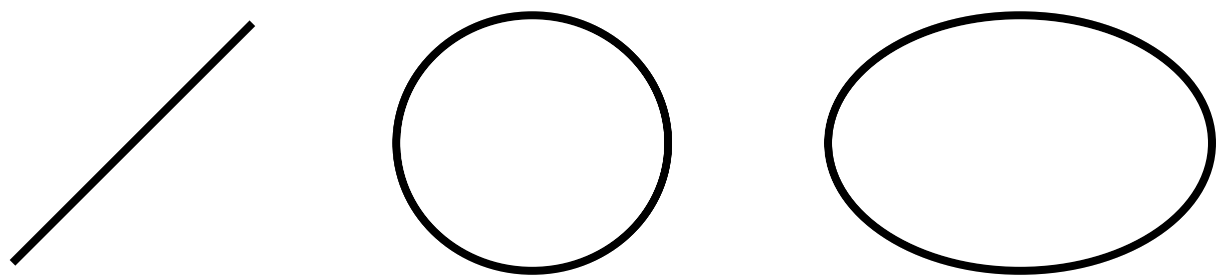
Primitivas Gráficas

Prof. Márcio Bueno
{cgtarde,cgnoite}@marciobueno.com

Fonte: Material do Prof. Robson Pequeno de Sousa e do Prof. Robson Lins

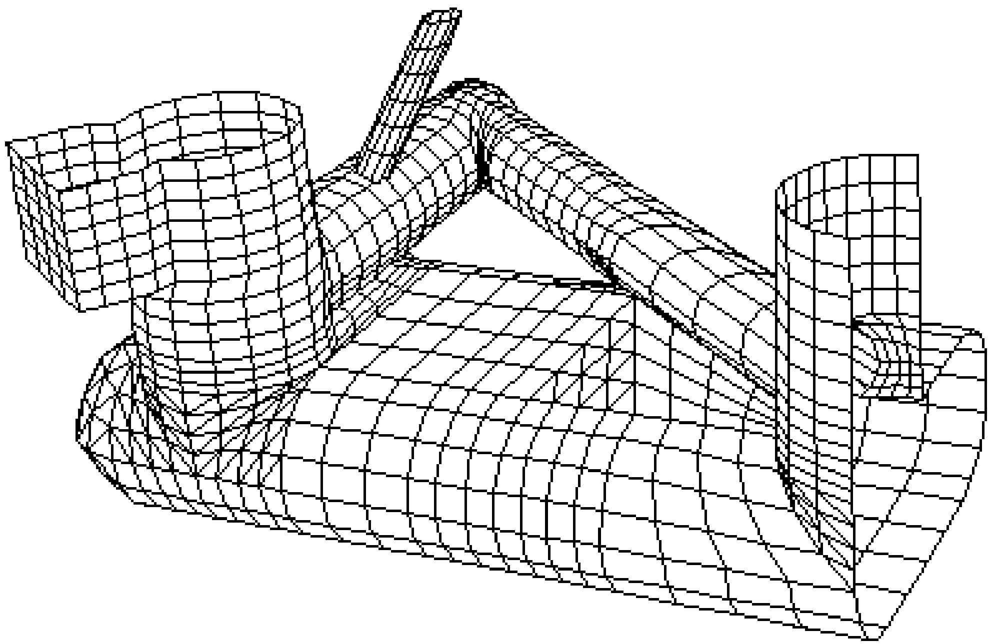
Traçado de Primitivas em Dispositivos Matriciais

- Conversão matricial (ou por varredura)
  - processo que permite realizar a conversão de um desenho qualquer armazenado na memória de imagem para um dispositivo matricial (ou *raster*)
- Primitivas Gráficas
  - reta, circunferência, elipse



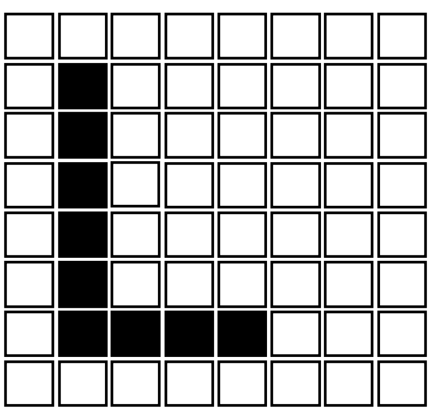
Traçado de Retas

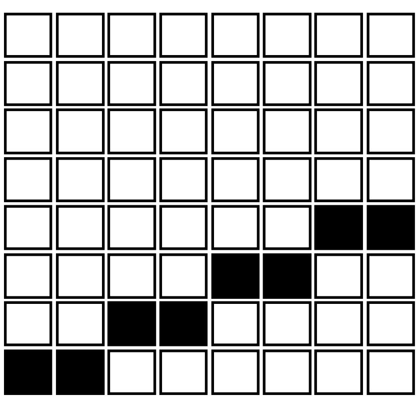
- A reta é a primitiva 2D mais comum
- Todos *wireframes* (modelos de arame) 3D são eventualmente retas 2D
  - Os algoritmos aperfeiçoados contêm numerosas técnicas e truques que ajudam a projetar algoritmos mais avançados

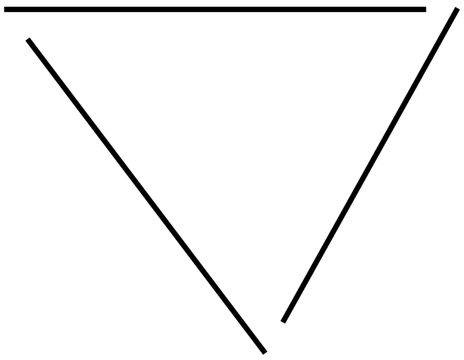


Características Desejáveis para o Traçado de Retas

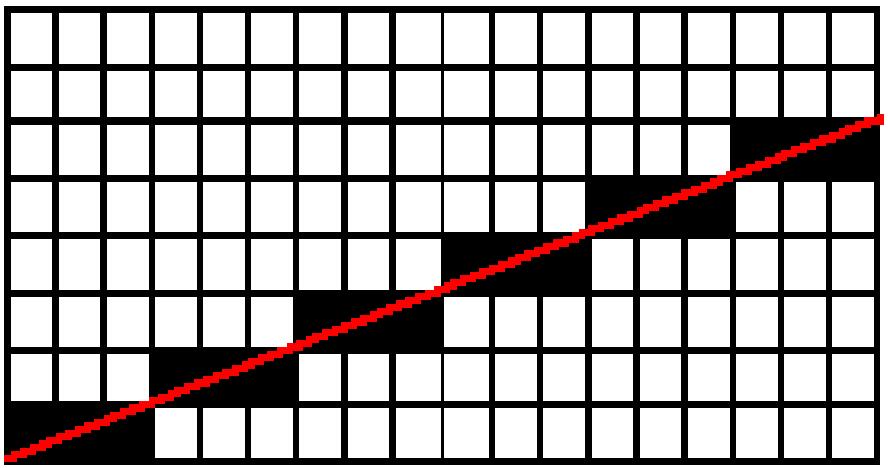
- Linearidade
  - Os *pixels* traçados devem dar a aparência de que estão sobre uma reta
- Espessura (densidade) uniforme
  - A densidade da reta é dada pelo <u>número de *pixels* traçados dividido pelo comprimento da reta</u>. Para manter a densidade constante, os pixels devem ser igualmente espaçados. A imagem do segmento de reta não deve variar de espessura ao longo de sua extensão.







Correção no Traçado de Retas



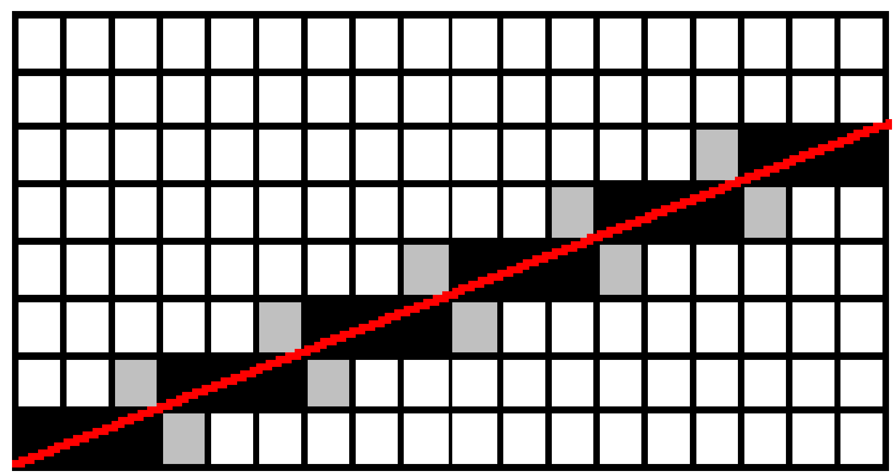
Reta Original

Aproximação feita no dispositivo raster

"Rasterização"

Suavização dos contornos da reta

"Anti-Aliasing"



Propriedades Exigidas no Traçado de Retas

- Precisão
  - os segmentos devem iniciar e terminar nos pontos especificados. Caso isso não ocorra, pequenos *gaps* podem surgir entre o final de um segmento e o início de outro
- Densidade independente da inclinação
  - para segmentos de retas de diferentes inclinações
- Continuidade
  - a imagem não apresenta interrupções indesejáveis
- Rapidez no traçado dos segmentos

Método do Declive - Algoritmo DDA (Digital Diferential Analyzer)

- Algoritmo DDA (ou do declive) usa o método incremental
  - Equações para $0 \leq m \leq 1$

    $y_{i+1} = mx_{i+1} + b = m(x_i + dx) + b$ **=>**

    $y_{i+1} = y_i + mdx$, *faça* $dx = 1$ =>

    $y_{i+1} = y_i + m$
  - Equações para $m > 1$

    $x_{i+1} = x_i + 1/m$, *faça* $dy = 1$

Implementação do Algoritmo DDA (em C)

```
void DDA(int X1,Y1,X2,Y2)
{
    int    Length, I;
    float X,Y,Xinc,Yinc;

    Length = ABS(X2 - X1);
    if (ABS(Y2 - Y1)  > Length)
            Length = ABS(Y2-Y1);
    Xinc = (X2 - X1)/Length;
    Yinc = (Y2 - Y1)/Length;
    X = X1;
    Y = Y1;
    while(X<X2){
      setpixel(Round(X),Round(Y));
      X = X + Xinc;
      Y = Y + Yinc;

    }
}
```

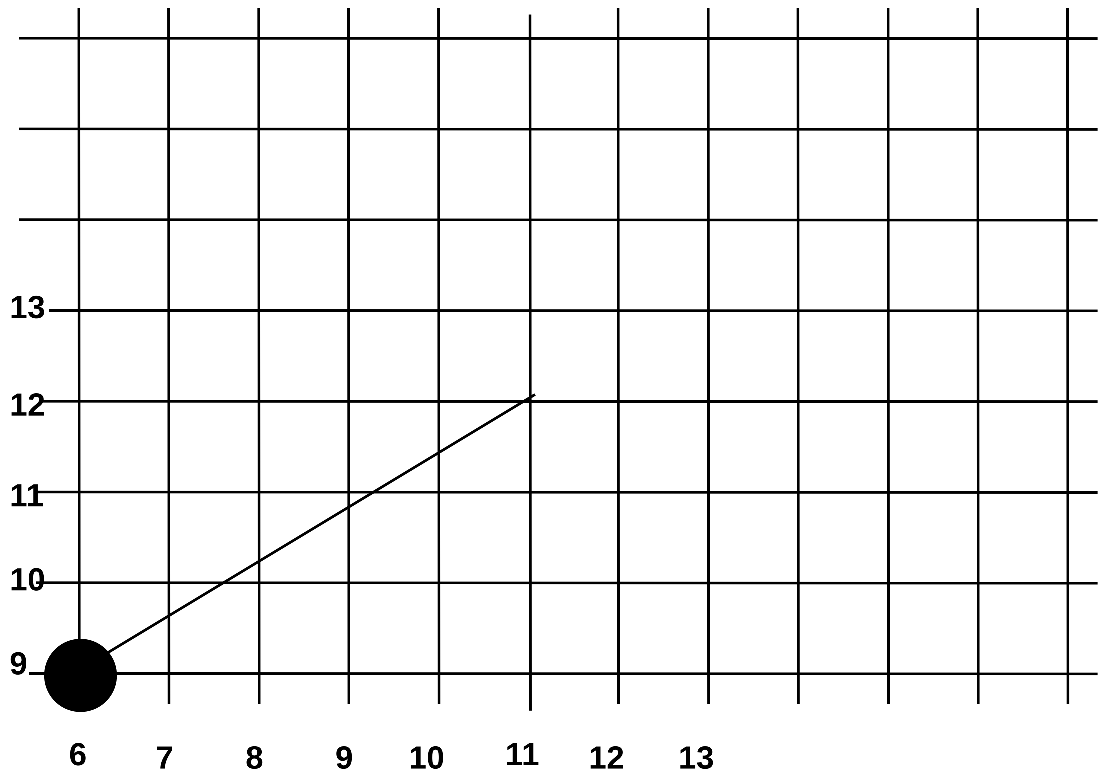
DDA Cria boas linhas, mas consome muito tempo devido as funções de arredondamento.

*DDA: *Digital Differential Analyzer* (Método do Declive)

Exemplo do DDA

- Compute quais pixeis devem ser escolhidos para representar a reta de (6,9) to (11,12).

- Length = ?
- Xinc = ?
- Yinc = ?



DDA Exemplo

Reta de (6,9) para (11,12).

Length := Max of (ABS(11-6), ABS(12-9)) = 5

Xinc := 1

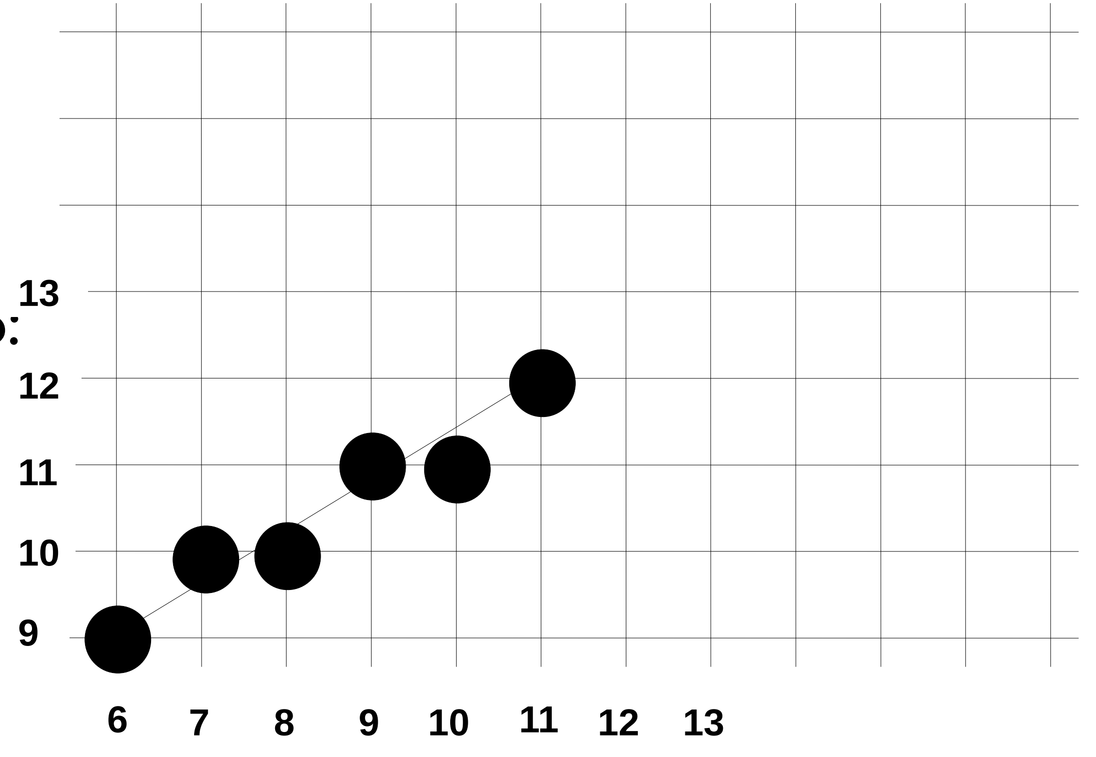
Yinc := 0.6

Os Valores computados são:

(6,9), (7,9.6),

(8,10.2), (9,10.8),

(10,11.4), (11,12)



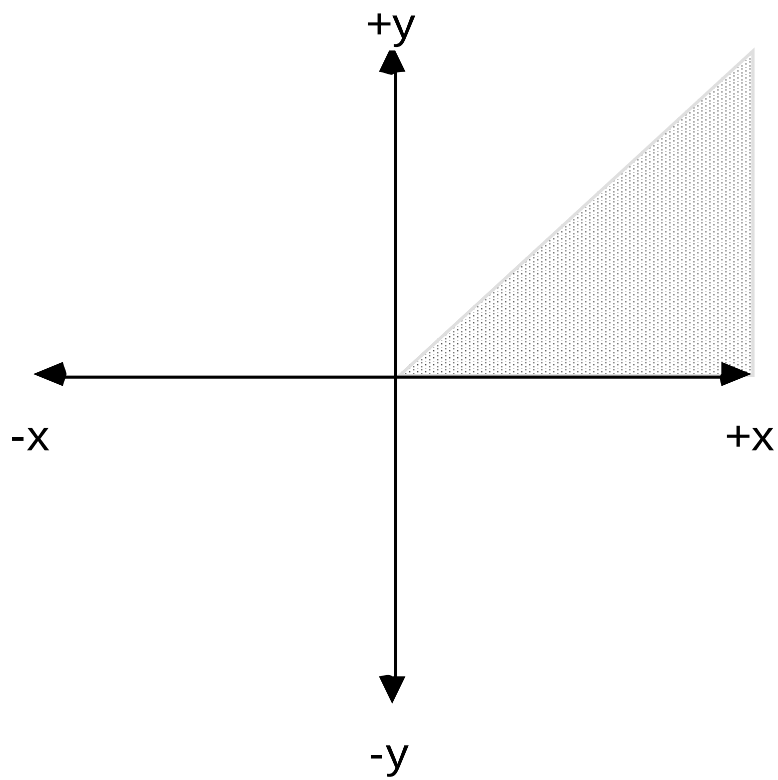
Retas Rápidas – Método do Ponto Médio

- Vamos estudar uma versão do Algoritmo de *Bresenham* (1965) denominada *Mid Point Line Algorithm* ("Algoritmo do Ponto-Médio ou Meio-Ponto para Retas") por *Foley* (1997) .

- *Bresenham* desenvolveu um algoritmo clássico que usa apenas variáveis inteiras, e permite que o cálculo de (x_i+1, y_i+1) seja feito incrementalmente, usando os cálculos já feitos para (x_i, y_i).

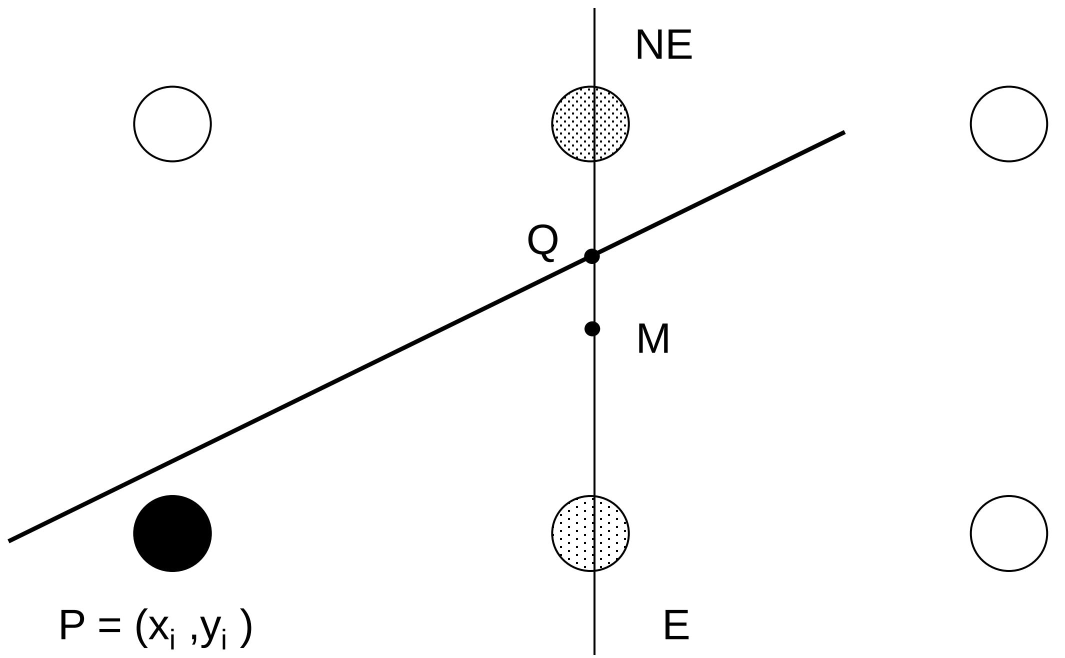
Retas Rápidas – Método do Ponto Médio

- Queremos desenhar uma reta entre os pontos (x_1,y_1) e (x_2,y_2) com declive m entre 0 e 1 (i.e. reta dentro do 1° octante)



Retas Rápidas (cont.)

- Para as retas do 1° octante, dado um pixel sobre a reta, o próximo pixel é para direita (E) ou para Direita acima (NE)
- Tendo o pixel (x_i, y_i), o próximo pixel é NE em (x_i+1, y_i+1) ou E em (x_i+1, y_i)



Forma Implícita da Reta

- Vamos determinar um método para calcular de que lado da reta está do ponto M. Para isso, considere sua função implícita, $F(x,y) = ax + by + c = 0$.
  - Se $dy = y2 - y1$, e $dx = x2 - x1$,
- A equação da reta em termos de sua inclinação pode ser escrita como:

$$y = \frac{dy}{dx}x + B$$

Estudo do valor da Função

- Forma Implícita

$$F(x, y) = dy.x - dx.y + dx.B = 0$$

- Em que
  - a = dy, b = - dx e c = dx.B

- É fácil verificar que:
  - F(x,y) = 0, ponto sobre a linha
  - F(x,y) > 0, para pontos abaixo da linha
  - F(x,y) < 0, para pontos acima da linha

Escolha de E ou NE

- Para o teste do ponto-médio, basta calcular

  $F(M) = F(x_p + 1, y_p + 1/2)$ e verificar o seu sinal.
- A decisão será tomada com base no valor da função no ponto $(x_p + 1, y_p + 1/2)$
- define-se uma "variável de decisão"

  - $$d = a(x_p + 1) + b(y_p + 1/2) + c$$

  - Se $d > 0$, escolhemos o pixel NE
  - Se $d < 0$, escolhemos o pixel E
  - Se $d = 0$ pode-se escolher qualquer um deles

Determinando a Forma Incremental

- Se E for escolhido, M é incrementado de 1 na direção x.

$$d_{new} = F(x_p + 2, y_p + \frac{1}{2}) = a(x_p + 2) + b(y_p + \frac{1}{2}) + c$$

$$d_{old} = a(x_p + 1) + b(y_p + \frac{1}{2}) + c$$

- Subtraindo $\boldsymbol{d_{old}}$ de $\boldsymbol{d_{new}}$ para obter a diferença incremental, tem-se
  - $\boldsymbol{d_{new} = d_{old} + a}$.

Determinando a Forma Incremental

- Se NE é escolhido, M é incrementado de 1 em ambas as direções, x e y.

$$d_{new} = F(x_p + 2, y_p + 1 + \frac{1}{2}) = a(x_p + 2) + b(y_p + \frac{3}{2}) + c$$

$$d_{old} = a(x_p + 1) + b(y_p + \frac{1}{2}) + c$$

- Subtraindo *dold* de *dnew*, tem-se
  - $\boldsymbol{d_{new} = d_{old} + a + b}$

Determinando o Ponto Inicial

- Como o primeiro pixel corresponde ao ponto *(x1,y1)*, pode-se calcular diretamente o valor inicial de d para escolher entre E e NE. O primeiro ponto-médio está em *(x1 + 1,y1 + 1/2)* temos:

$$d_{start} = F(x_1 + 1, y_1 + \frac{1}{2}) = a(x_1 + 1) + b(y_1 + \frac{1}{2}) + c$$

$$d_{start} = ax_1 + a + by_1 + b.\frac{1}{2} + c = \underbrace{ax_1 + by_1 + c}_{F(x1,\ y1)} + a + \frac{1}{2}b$$

$$d_{start} = a + \frac{1}{2}b$$

- Como $F(x_1, y_1)$ está sobre a reta, temos que F(x1 , y1) = 0, daí o resultado acima.

Algoritmo

```
int x1,x2, y1,y2, dx, dy, incE, incNE, d, x,y;
int valor;
{
  dx=x2-x1;
  dy=y2-y1;
  d=2*dy-dx;    /* valor inicial para o fator de decisão */
  incE=2*dy;  /* Incr. que move para E */
  incNE=2*(dy-dx);  /* Incr. que move para NE */
  x=x1;  y=y1;
  write_Pixel (x,y,valor);  /* Pinta pixel inicial */
  while (x < x2) {
          if (d <= 0) {
                    d=d+incE;   /* Escolhe E */
                    x=x+1;
                    }
            else {            /* Escolhe NE */
                    d=d+incNE;
                    x=x+1;  y=y+1;  /* pois é maior que 45° */
                    }
          write_pixel (x,y,valor);
          }  /* fim do while */
}
```

Exemplo

- Indicar que localizações serão calculadas pelo algoritmo de Bresenham quando se gera por varredura um segmento de reta entre (1,1) e (8,5) em coordenadas de tela.

d	x	y
$1 + inc_2 = -5$	1	1
$-5 + inc_1 = 3$	2	2
$3 + inc_2 = -3$	3	2
$-3 + inc_1 = 5$	4	3
$5 + inc_2 = -1$	5	3
$-1 + inc_1 = 7$	6	4

$dx = 8 - 1,\ dy = 5 - 1$

$Inc_1 = 2dy = 2*4 = 8$

$Inc_2 = 2*(dy - dx) = 2*(4 - 7) = -6$

$d_{start} = inc_1 - dx = 8 - 7 = 1$

Traçado de Circunferência

- A equação de uma circunferência com centro na origem e raio R, em coordenadas cartesianas, é dada por

$$x^2 + y^2 = R^2$$

- A equação explicita da circunferência

$$y = f(x) : y = \pm\sqrt{R^2 - x^2}$$

Traçado de Circunferência

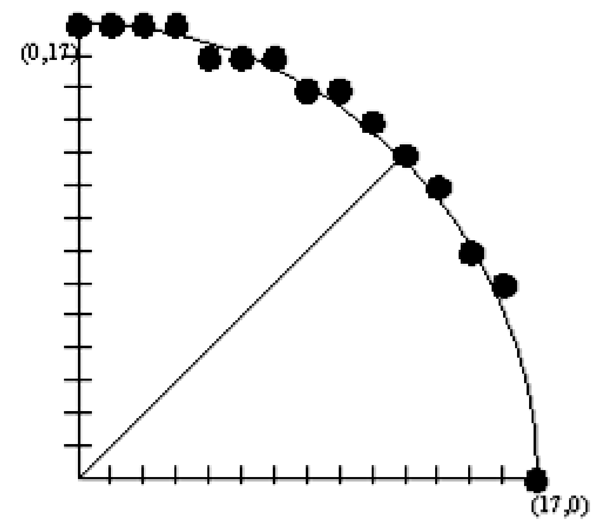
- Algoritmos Alternativos
  - polígono regular de *n* lados é usado como aproximação para a circunferência

  - Desvantagem

    - *n* deve ser suficientemente grande para se ter uma boa aproximação
      - Quanto maior for o n $\Rightarrow$ algoritmo mais lento
      - Estratégias de aceleração precisam ser usadas

Traçado de Circunferência

- Algoritmos Alternativos
  - equação explícita da circunferência, *y=f(x)* para desenhar ¼ da circunferência (os outros ¾ são desenhados por simetria)
  - Poderíamos variar x de 0 a R, em incrementos de uma unidade, calculando +y a cada passo através da equação acima
- Desvantagens
  - Requer operações de multiplicação e raiz quadrada
  - Grandes *gaps* nas regiões onde a tangente à circunferência é infinita



Simetria de Ordem 8

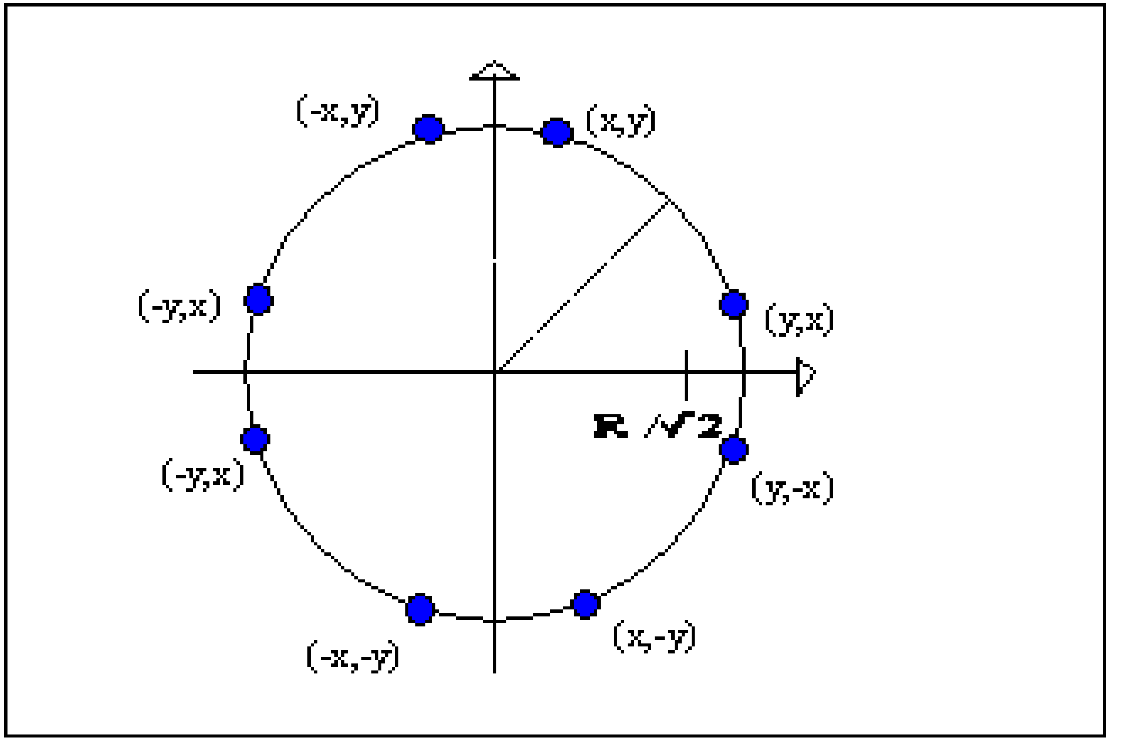
- O traçado de uma circunferência pode tirar proveito de sua simetria

  - Suponha que a circunferência esteja centrada na origem
  - Se o ponto *(x,y)* pertence à circunferência, pode-se calcular os sete outros pontos da circunferência

⇨ Basta calcular um arco de circunferência de 45° para obter a circunferência toda

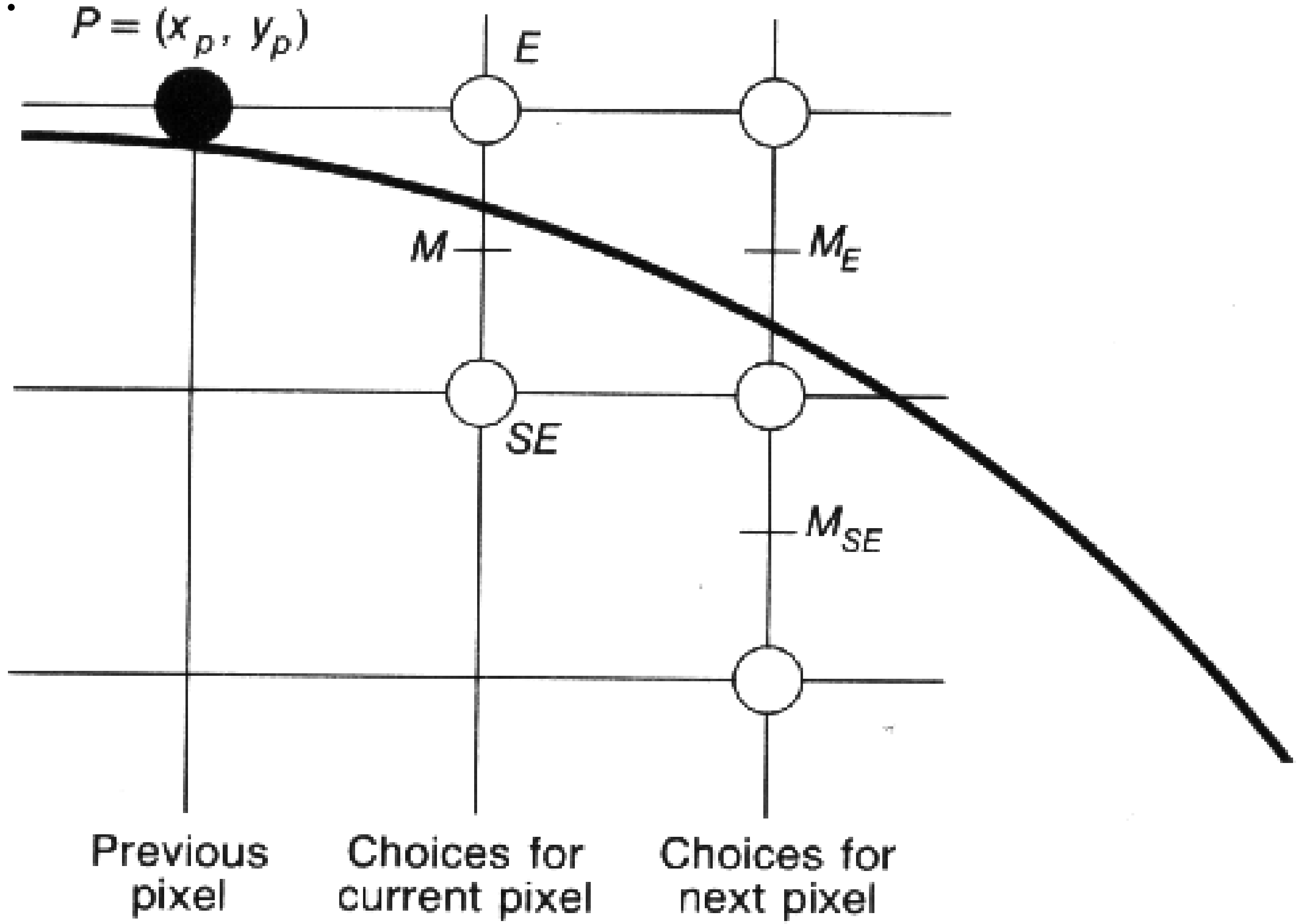
Simetria de Ordem 8

```
void PontosCircunferência(int x,int y,int valor)
{
   PintaPixel(x,y,valor);
   PintaPixel(y,x,valor);
   PintaPixel(y,-x,valor);
   PintaPixel(x,-y,valor);
   PintaPixel(-x,-y,valor);
   PintaPixel(-y,-x,valor);
   PintaPixel(-y,x,valor);
   PintaPixel(-x,y,valor);
}
```



Algoritmo do Ponto Médio para Circunferência

- Suponha que desenhamos o pixel (x_p, y_p). Qual o proximo pixel a ser desenhado?



Algoritmo do Ponto Médio para Circunferência

- Considere apenas um arco de 45° da circunferência, o 2° octante, de $x=0, y=R$ a $x=y= R/(2)^{1/2}$.

- Use o procedimento PontosCircunferência para traçar todos os pontos da circunferência.

- Assim como o algoritmo gerador de linhas, a estratégia é selecionar entre 2 pixels na malha aquele que está mais próximo da circunferência, utilizando o sinal da função no ponto intermediário entre os dois possíveis pixels (E ou SE).

ALGORITMO DO PONTO MÉDIO PARA CIRCUNFERÊNCIA

- Sinal da função através da equação implícita da circunferência

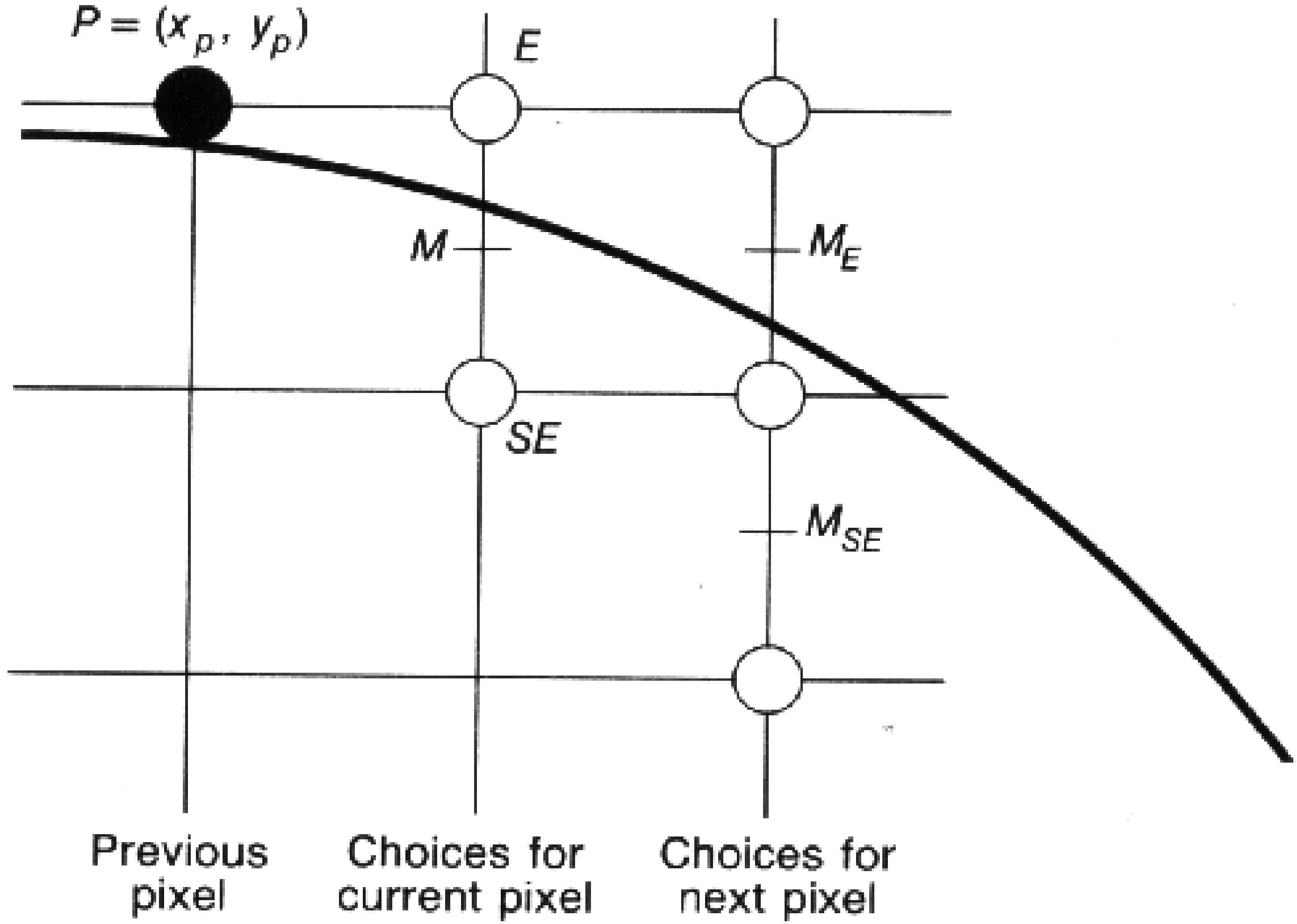
$$F(x,y) = x^2 + y^2 - R^2$$

$= 0$ sobre a circunferência

> 0 fora da circunferência

< 0 dentro da circunferência

Gerando a Variável de Decisão



Gerando a Variável de Decisão

- De forma semelhante a rasterização da reta, determina-se a variável de decisão d_{old}
  - $$d_{old} = F(x_p+1, y_p-½) = (x_p+1)^2 + (y_p-½)^2 - R^2$$
  - Se $d_{old} < 0$, E é escolhido e o próximo ponto médio será incrementado de 1 na direção x
  - $$d_{new} = F(x_p+2, y_p-½) = d_{old} + (2x_p + 3)$$
  - Se $d_{old} \geq 0$, SE é escolhido e o próximo ponto médio será incrementado de 1 na direção x e decrementado de 1 na direção y
  - $$d_{new} = F(x_p+2, y_p-½-1) = d_{old} + (2x_p - 2y_p + 5)$$

Variável de Decisão Inicial

- Se iniciamos em (0,R), o próximo ponto médio estará em (1, R – ½), portanto o valor inicial de d é dado por:

  - $d_{init} = F(1, R - ½)$
    $= 5/4 - R.$

- Se R for inteiro, faça h = d – ¼ -> d = h + ¼ , segue que a inicialização será h = 1 - R.

  $d_{init} = 1 - R$

Algoritmo do Ponto Médio para Circunferência

```
void pontomedio(int raio, int valor) {
 int x = 0;
 int y = raio;
 double d = 5/4 – raio;
 ponto_circulo (x, y, valor);
 while (y > x) {
   if (d < 0) /* escolhe E */
     d + = 2.0*x + 3.0;
   else {  /* escolhe SE */)
     d + = 2.0*(x – y) + 5;
     y - -;
   }
   x ++;
   ponto_circulo (x, y, valor);
 } /* while*/
} /*pontomedio*/
void ponto_circulo(int x, int y, int valor)
{
   writepixel(x, y, valor);
   writepixel(y, x, valor)
   writepixel(y, -x, valor)
   writepixel(x,-y , valor)
   writepixel(-x, -y, valor)
   writepixel(-y, -x, valor)
   writepixel(-y, x, valor)
   writepixel(-x, y, valor)
} /*ponto_circulo*/
```

Resolução Gráfica

- Todos os dispositivos de entrada e saída gráficos utilizam virtualmente uma malha retangular de posições endereçáveis a qual é definida como retângulo de visualização.

- Resolução gráfica de um dispositivo é o número de posições (*pixels*) horizontais e verticais que ele pode distinguir.

Resolução Gráfica

- Quatro parâmetros definem a resolução gráfica
  - *ndh* - o número de posições endereçáveis horizontalmente
  - *ndv* - o número de posições endereçáveis verticalmente.
  - *width* - a largura do retângulo de visualização em mm.
  - *height* - a altura do retângulo de visualização em mm.

Resolução Gráfica

- A partir dos 4 parâmetros as seguintes medidas são definidas:
  - resolução horizontal: *horiz_res = ndh / width*
  - tamanho do ponto horizontal: *horiz_dot_size = width / ndh*
  - resolução vertical: *vert_res = ndv / height*
  - tamanho do ponto vertical: *vert_dot_size = height / ndv*
  - total de pontos endereçáveis: *total_nr_dots = ndh * ndv*
  - resolução de área: area_res = *total_nr_dots / (width * height)*
  - razão de aspecto gráfica: aspect_ratio = *vert_dot_size / horiz_dot_ size*
  - razão de aspecto físico: physical_aspect_ratio = height / width

Sistemas de Coordenadas

- De forma geral são definidos três sistemas de coordenadas:
  - Sistemas de coordenadas do mundo
    - É um conjunto de coordenadas cartesianas em um intervalo qualquer definido pelo usuário.
  - Sistemas de coordenadas do dispositivo
    - É um conjunto de *pixels* endereçáveis pelo dispositivo. Os pixels são endereçados por dois números inteiros que dão suas coordenadas horizontal e vertical.

Sistemas de Coordenadas

- Sistema de coordenadas normalizadas do dispositivo (NDC).
  - Este sistema é intermediário entre os dois anteriores, definido de tal forma que todo conteúdo da janela de visualização possua as coordenadas variando no intervalo [0,1] X [0,1]

  - Vantagem da utilização de NDCs é que padrões gráficos podem ser discutidos usando um sistema de coordenadas independente de dispositivos gráficos específicos.

Sistemas de Coordenadas

- Sistema de Coordenada do mundo.

$$x\min \le x \le x\max$$

$$y\min \le y \le y\max$$

- Sistema de Coordenada dispositivo

$$0 \le dcx \le ndh - 1$$

$$0 \le dcy \le ndv - 1$$

- Sistema de Coordenada NDC

$$0 \le ndcx \le 1$$

$$0 \le ndcy \le 1$$

Sistemas de Coordenadas

- Transformação de NDCs para dispositivos

$$dcx = round(ndcx * (ndh - 1))$$

$$dcy = round(ndcy * (ndv - 1))$$

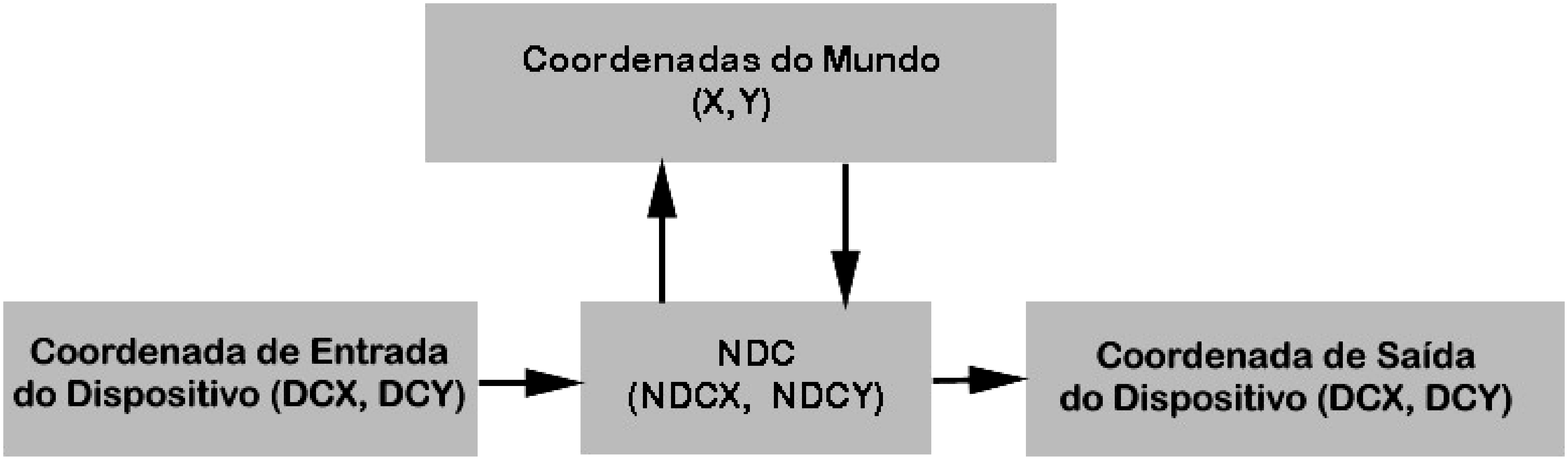
- Transformação de coordenadas do mundo para NDCs.

$$ndcx = \frac{(x - x\min)}{(x\max - x\min)}$$

$$ndcy = \frac{(y - y\min)}{(y\max - y\min)}$$

Sistemas de Coordenadas

- Seqüência de transformações



Exercício

- Escreva os procedimentos "inp_to_ndc", "ndc_to_user", "user_to_ndc" e "ndc_to_user", que transformam dados entre os vários sistemas de coordenadas, conforme ilustrado na figura anterior. Repita o exercício assumindo que o intervalo de variação do sistema NDC vai de:
  - -1 a +1 (coordenadas normalizadas centradas)
  - 0 a 100
- Implemente os procedimentos acima.