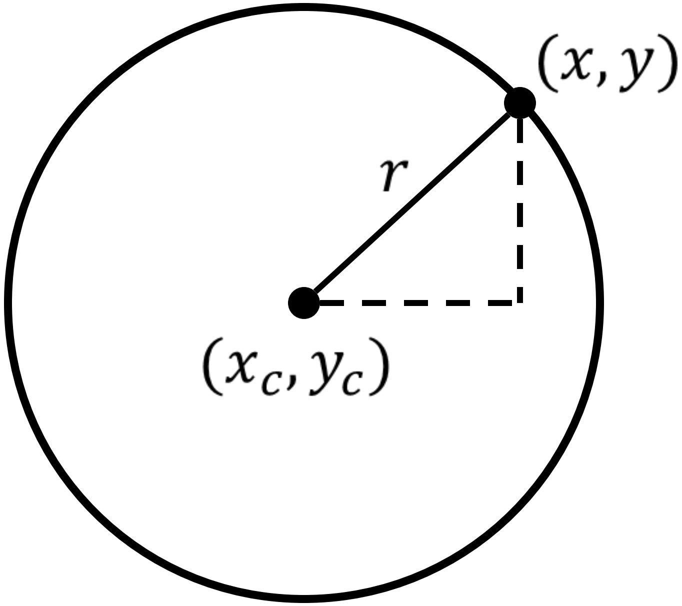
CÍRCULOS, ELIPSES E CURVAS

Prof. Dr. Bianchi Serique Meiguins
Prof. Dr. Carlos Gustavo Resque dos Santos

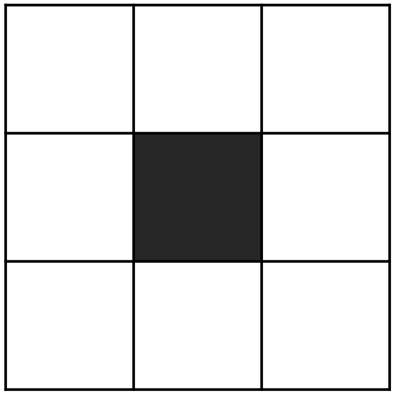
Círculo

- Matemáticamente:
  - $(x - x_c)^2 + (y - y_c)^2 = r^2$

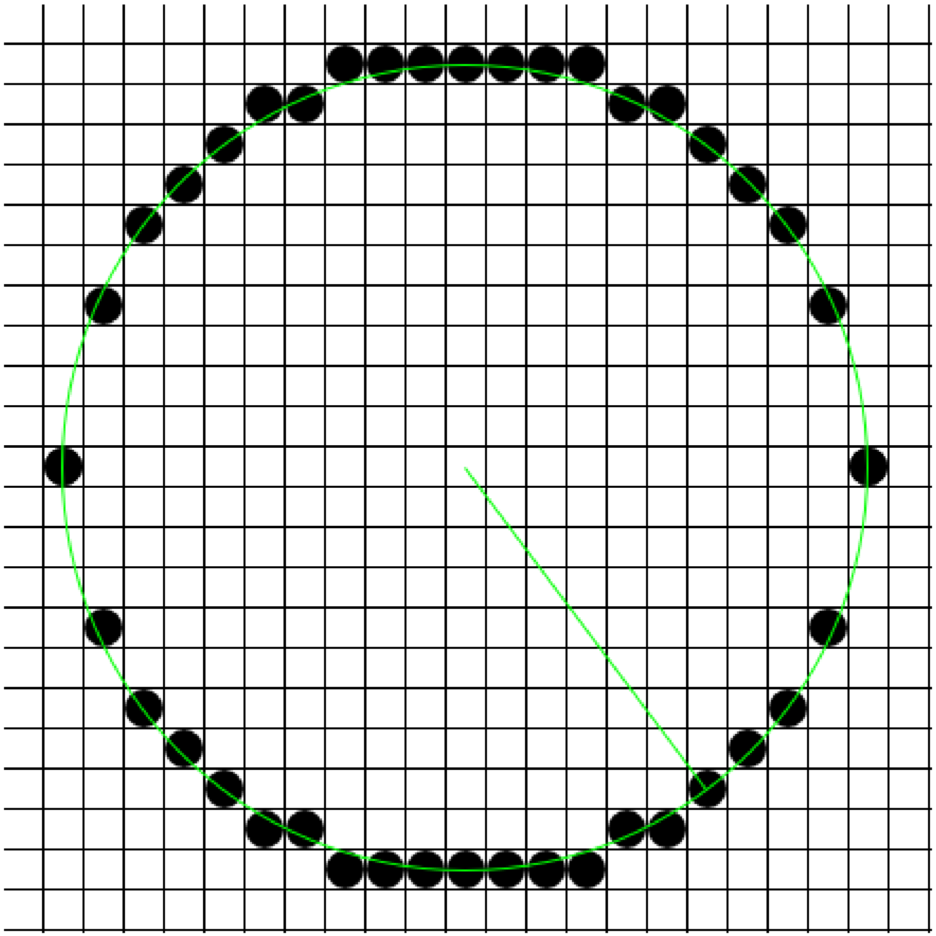


Rasterizar o Círculo

- O semelhante ao da linha. Mas neste caso, desenhar a borda do círculo
- Devemos encontrar os pixels mais próximos do círculo contínuo
- Para manter a continuidade da linha e sua largura em 1 pixel
  - ter para cada pixel pintado exatos 2 pixels vizinhos também pintados
  - Considerando 8 vizinhos ao redor



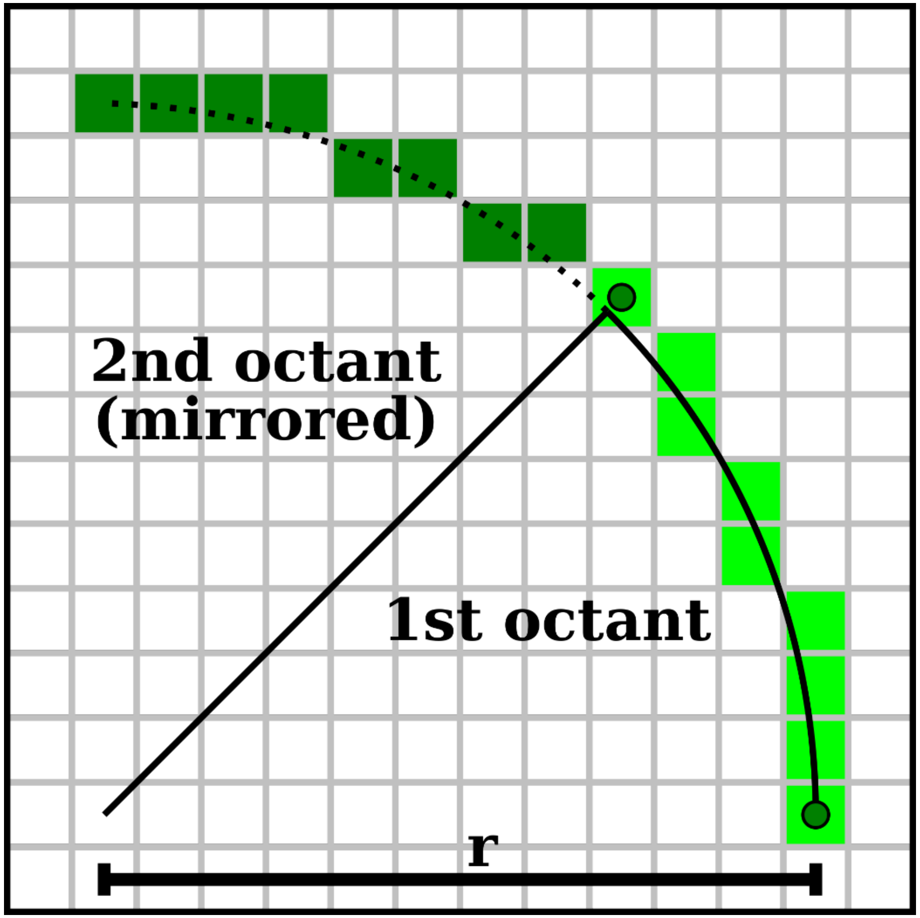
1ª Tentativa

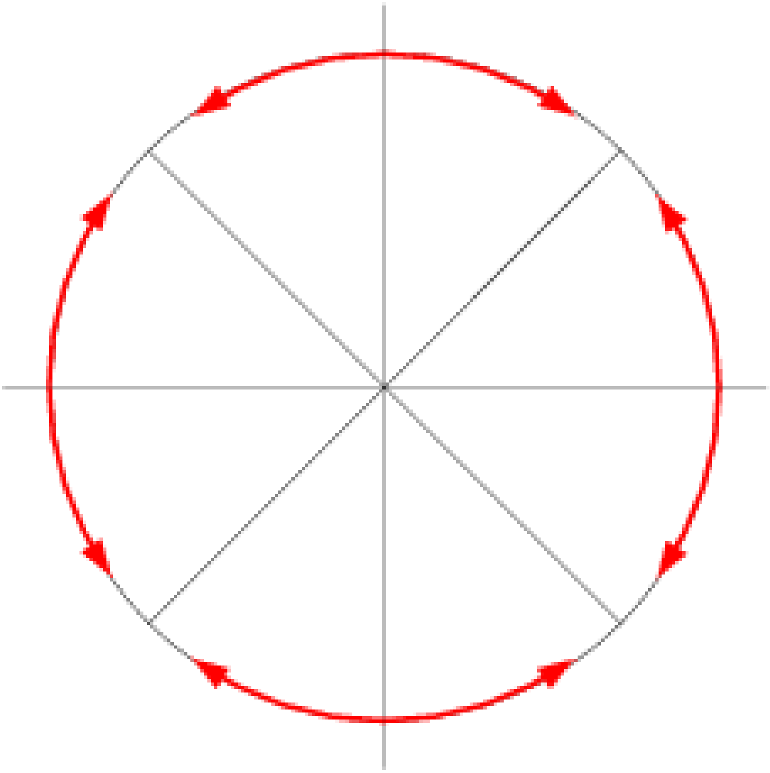


- Utilizando
  - $x_{n+1} = x_n + 1$
- E a fórmula
  - $y = y_c + \sqrt{r^2 - (x - x_c)^2}$
- Utilizando como ponto inicial
  - $(0, r)$

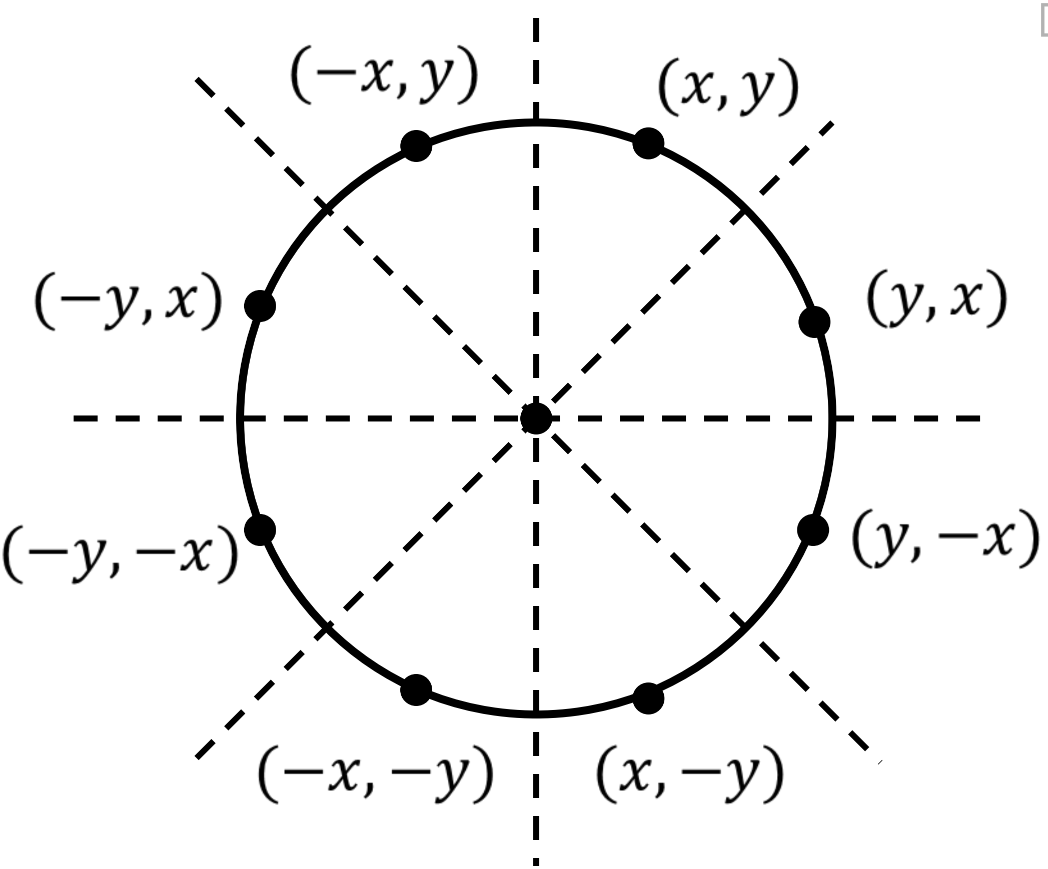
Problema do Passo

- Usando a simetria do círculo em octantes





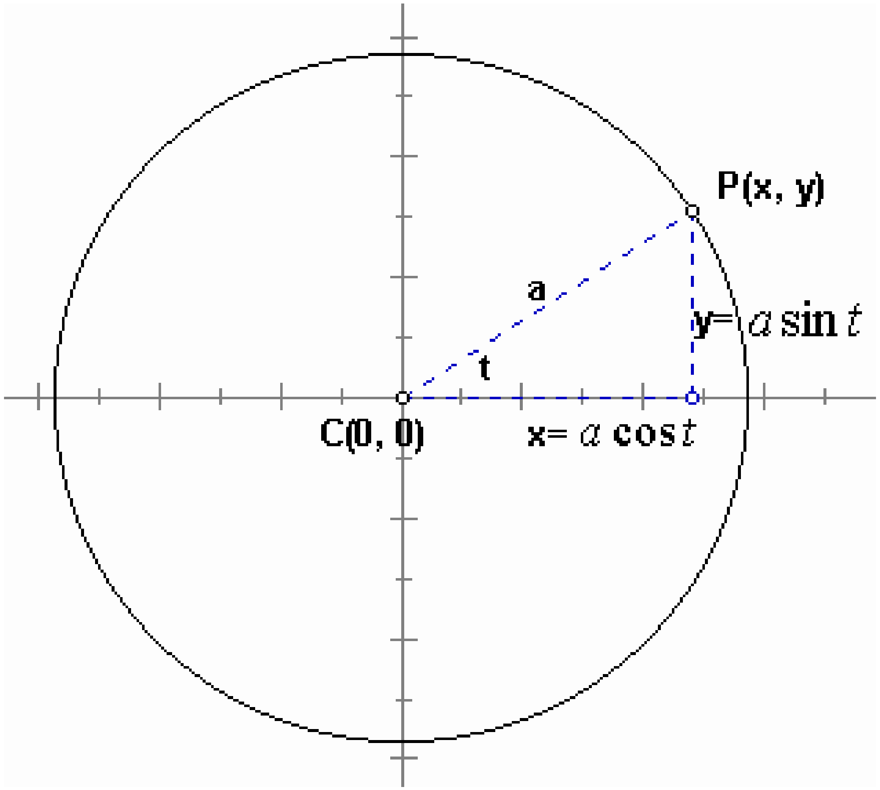
Simetria do Círculo



- Reflexão de pontos em octantes:

x	y
$-x$	y
x	$-y$
$-x$	$-y$
y	x
$-y$	x
y	$-x$
$-y$	$-x$

Usando Coordenadas Polares



- $x = r \times \cos(\sigma)$
- $y = r \times \sin(\sigma)$
- $\sigma \mathrel{+}= step$

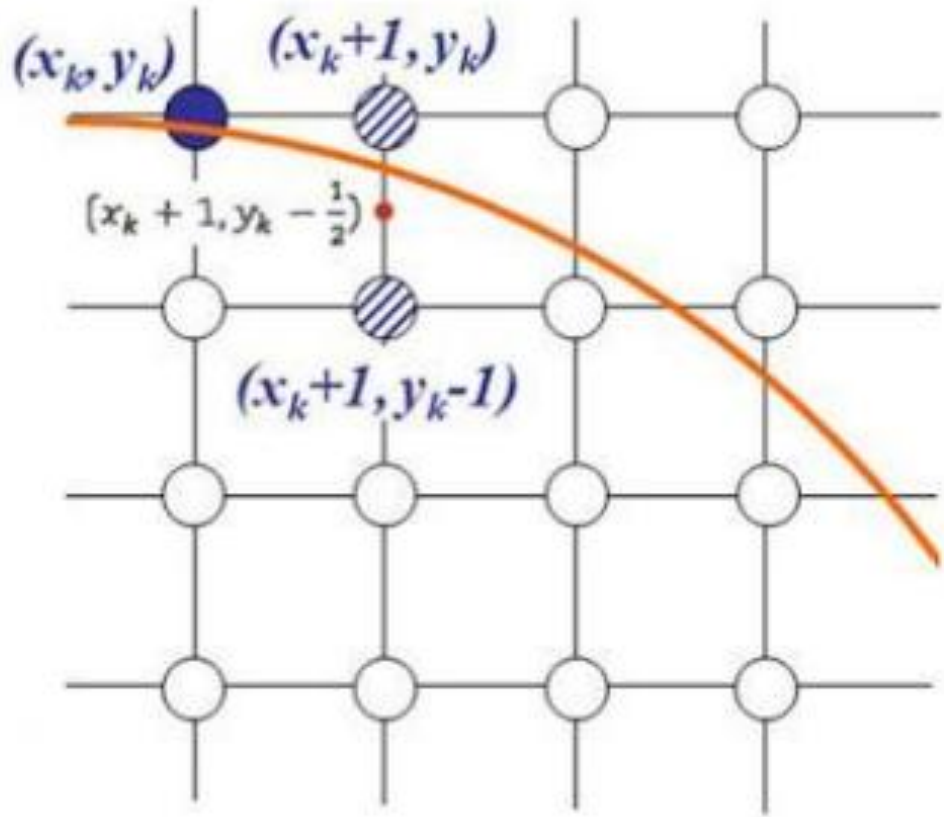
Problemas

- Usando a fórmula do círculo
  - Potenciação
  - Raiz Quadrada

- Usando Coordenadas Polares
  - Encontrar passo para o ângulo σ
  - Utilização de seno e cosseno na fórmula

Algoritmo do Ponto Médio

- Reflexão de octantes
- Pixel inicial $(0, r)$
- Calcula o erro recursivo em relação ao ponto médio
- Decide se
  - $y_{i+1} = y_i - 1$
- Ou
  - $y_{i+1} = y_i$



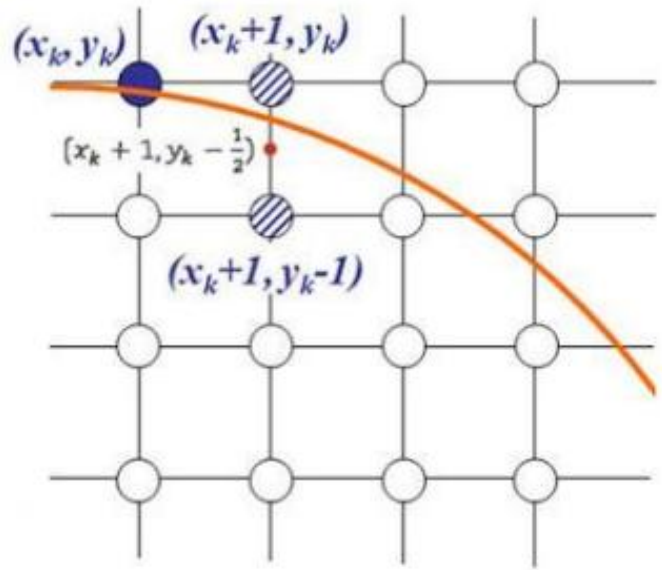
Algoritmo do Ponto Médio

- Para facilitar as contas, considera-se o centro em (0,0). Sendo assim:

- $x^2 + y^2 = r^2$

- Como x e y devem ser discretos, nem sempre a soma dos seus quadrados é igual ao raio ao quadrado, portanto ao subtrair os dois valores temos um erro.

- $e(x, y) = x^2 + y^2 - r^2$

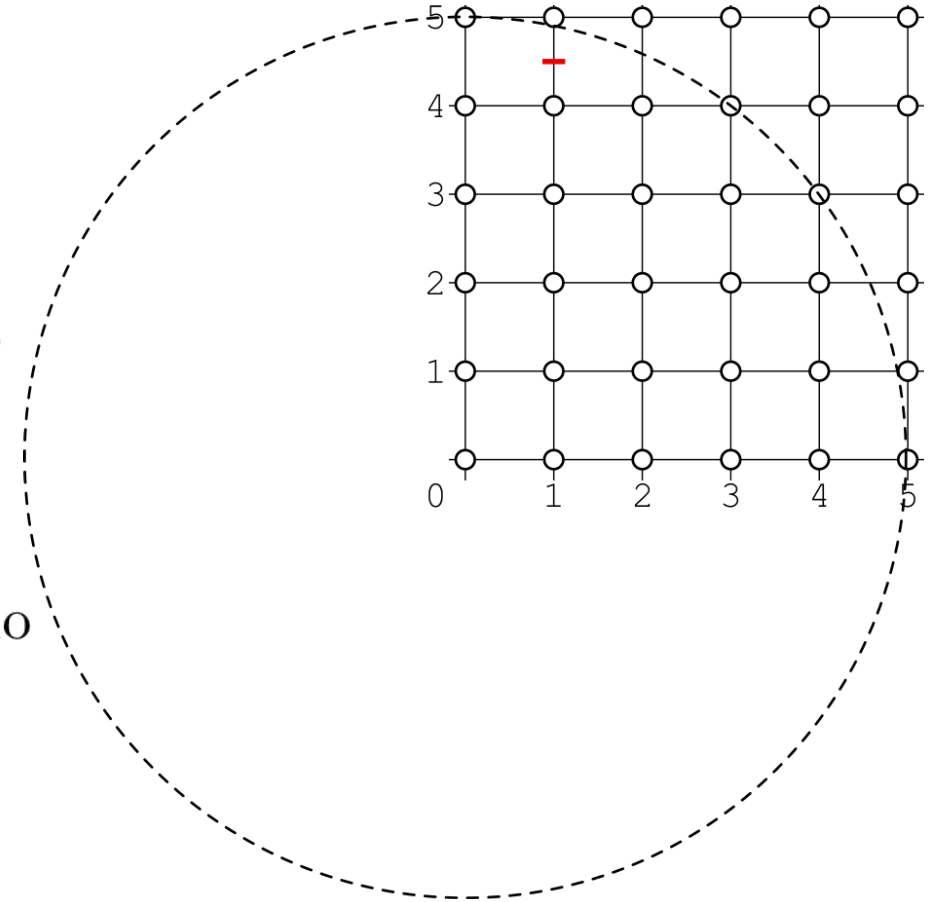


Algoritmo do Ponto Médio

- O objetivo é escolher para um determinado x qual o pixel que tem o y mais próximo da curva do círculo

- Para isso verificamos se a curva está acima ou abaixo do meio entre esses dois pixels.

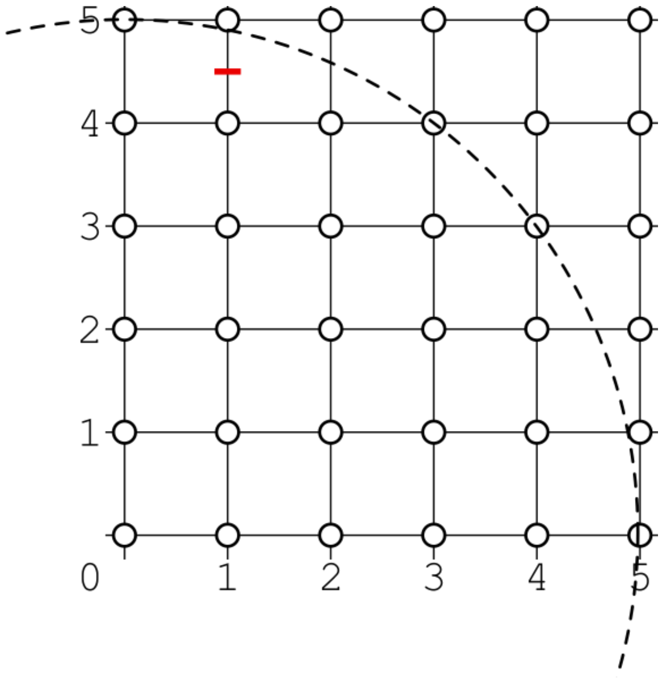
- Esse valor de meio é armazenado na variável auxiliar a cada iteração



Algoritmo do Ponto Médio

- Ao somar $+1$ no x, o próximo erro pode ser calculado assim:

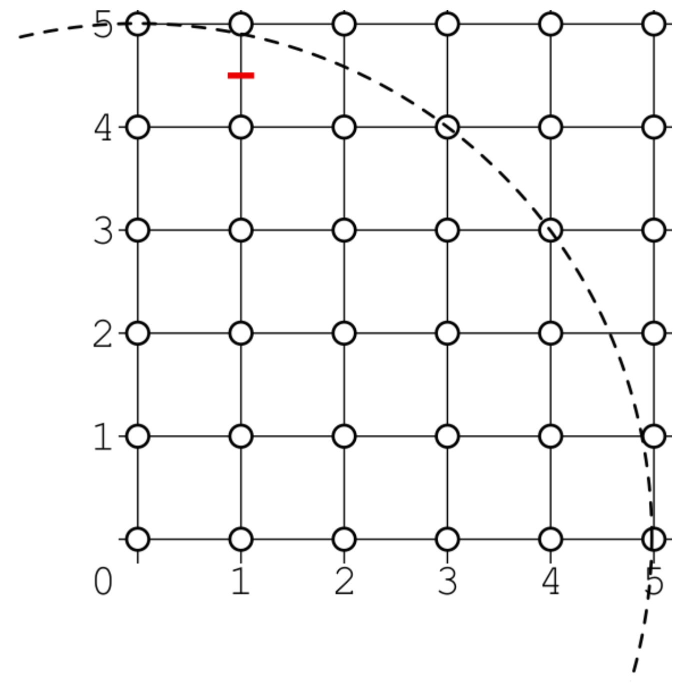
- $e(x+1, y) = (x+1)^2 + y^2 - r^2$
- $e(x+1, y) = x^2 + 2x + 1 + y^2 - r^2$
- $e(x+1, y) = x^2 + y^2 - r^2 + 2x + 1$
- $e(x+1, y) = e(x, y) + 2x + 1$



$$\boxed{e(x, y) = x^2 + y^2 - r^2}$$

Algoritmo do Ponto Médio

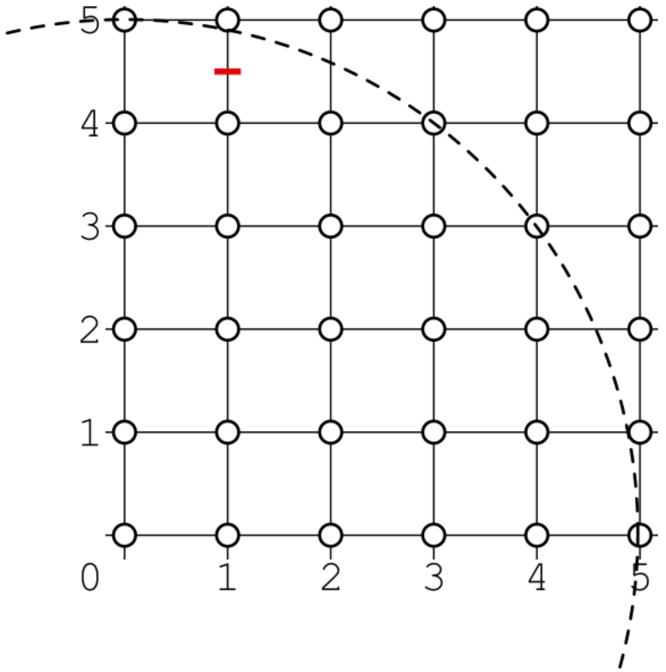
- Para decrementar y o erro deve ser calculado a partir do ponto médio entre os dois pixels
- Sendo assim, se $e < 0$ a curva está passando acima do ponto médio, portanto mais próximo do pixel de cima
- Caso contrário, está passando abaixo do ponto médio, portando mais próximo do pixel de baixo
- Para realizar esse procedimento, inicializa-se o erro assim:
- $e_0\left(0, r - \frac{1}{2}\right) = 0^2 + \left(r - \frac{1}{2}\right)^2 - r^2$



$$\boxed{\begin{gathered} e(x,y) = x^2 + y^2 - r^2 \\ e(x+1,y) = e(x,y) + 2x + 1 \end{gathered}}$$

Algoritmo do Ponto Médio

- $e_0\left(0, r - \frac{1}{2}\right) = 0^2 + \left(r - \frac{1}{2}\right)^2 - r^2$
- $e_0\left(0, r - \frac{1}{2}\right) = r^2 - r + \frac{1}{4} - r^2$
- $e_0\left(0, r - \frac{1}{2}\right) = -r + \frac{1}{4}$
- Este valor inicial pode ignorar o ¼ pois, como veremos a frente, o erro é sempre incrementado com valores discretos.



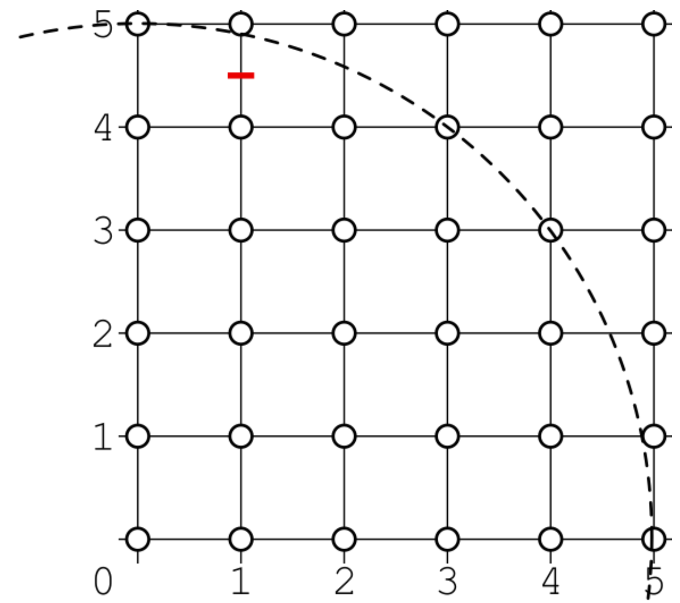
$$e(x, y) = x^2 + y^2 - r^2$$

$$e(x+1, y) = e(x, y) + 2x + 1$$

Algoritmo do Ponto Médio

- Deve-se levar em consideração que o erro é sempre calculado para o ponto médio, ou seja $y - 1/2$

- Sendo assim, para subtrair -1 no **y** utiliza-se $y - 1/2$ como ponto inicial e $y - 3/2$ como final



- $e_{ini}(x, y - \frac{1}{2}) = x^2 + \left(y - \frac{1}{2}\right)^2 - r^2$

- $e_{ini}\left(x, y - \frac{1}{2}\right) = x^2 + y^2 - y + \frac{1}{4} - r^2$

$$e(x,y) = x^2 + y^2 - r^2$$

$$e(x+1,y) = e(x,y) + 2x + 1$$

$$e_0\left(0, r - \frac{1}{2}\right) = -r + \frac{1}{4}$$

Algoritmo do Ponto Médio

- $e_{ini}\left(x, y - \frac{1}{2}\right) = x^2 + y^2 - y + \frac{1}{4} - r^2$
- Calculando o erro final:
- $e_{fin}\left(x, y - \frac{3}{2}\right) = x^2 + \left(y - \frac{3}{2}\right)^2 - r^2$
- $e_{fin}\left(x, y - \frac{3}{2}\right) = x^2 + y^2 - 3y + \frac{9}{4} - r^2$
- $e_{fin}\left(x, y - \frac{3}{2}\right) = x^2 + y^2 + (-y - 2y) + \left(\frac{1}{4} + 2\right) - r^2$
- $e_{fin}\left(x, y - \frac{3}{2}\right) = e_{ini}\left(x, y - \frac{1}{2}\right) - 2y + 2$

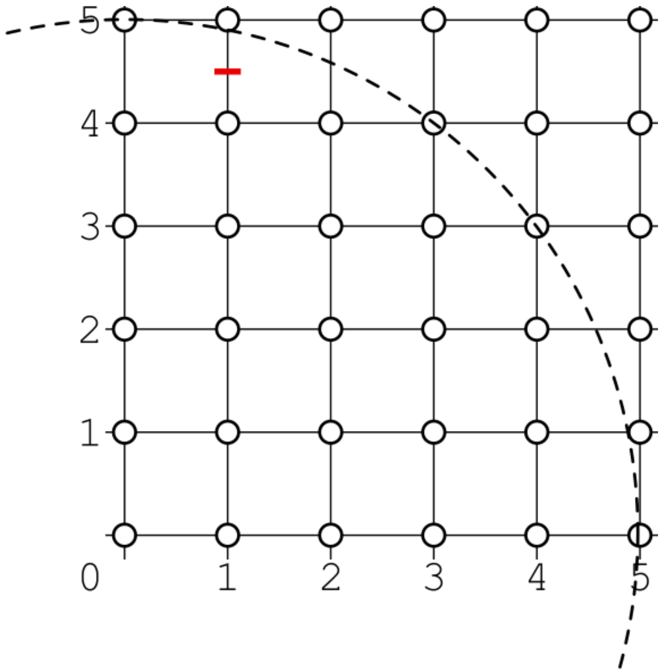
Algoritmo do Ponto Médio

$$e(x, y) = x^2 + y^2 - r^2$$

$$e(x+1, y) = e(x, y) + 2x + 1$$

$$e_0\left(0, r - \frac{1}{2}\right) = -r + \frac{1}{4}$$

$$e_{fin}\left(x, y - \frac{3}{2}\right) = e_{ini}\left(x, y - \frac{1}{2}\right) - 2y + 2$$



Algoritmo do Ponto Médio

- Entrada:
- r //raio
- (x_c, y_c) //centro

1. $x = 0$
2. $y = r$
3. $e = -r$

- desenha8(x, y, x_c, y_c)

2. Enquanto$(x \leq y)$
   1. $e += 2x + 1$
   2. $x++$
   3. Se$(e \geq 0)$
      1. $e += 2 - 2y$
      2. $y--$
   4. Fim-Se
   5. desenha8(x, y, x_c, y_c)
3. Fim-Enquanto

Algoritmo do Ponto Médio

- desenha8(x, y, x_c, y_c)

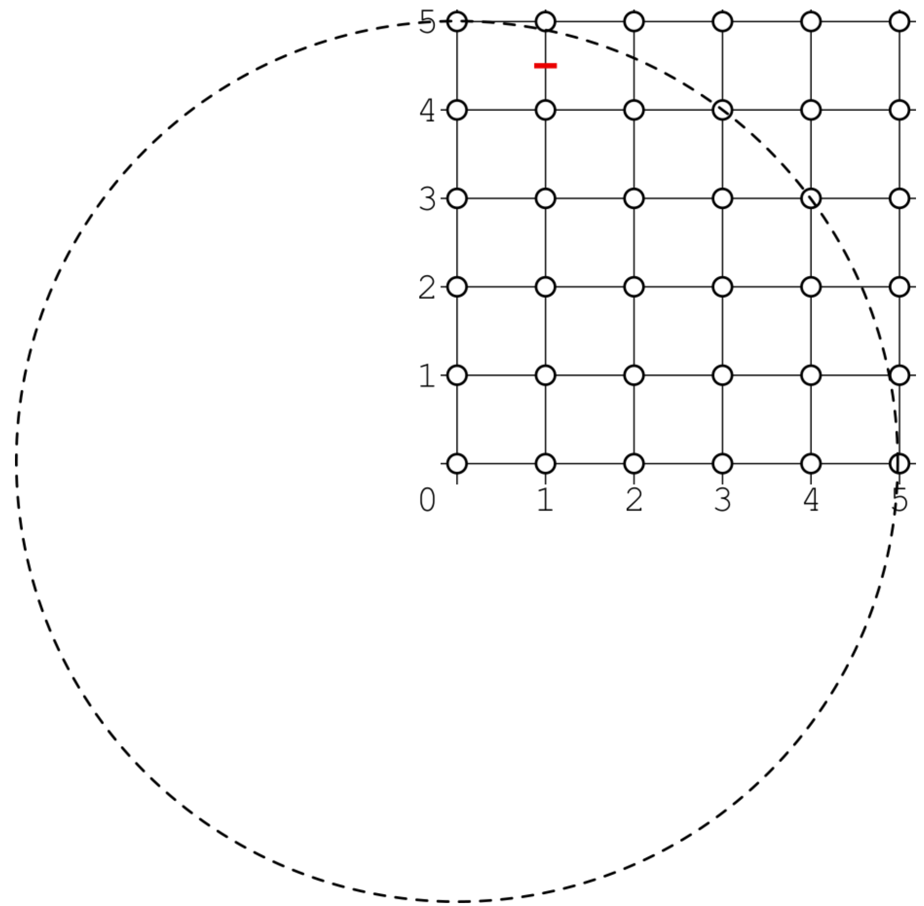
1. setPixel($x + x_c, y + y_c$)
2. setPixel($y + x_c, x + y_c$)
3. setPixel($y + x_c, -x + y_c$)
4. setPixel($x + x_c, -y + y_c$)
5. setPixel($-x + x_c, -y + y_c$)
6. setPixel($-y + x_c, -x + y_c$)
7. setPixel($-y + x_c, x + y_c$)
8. setPixel($-x + x_c, y + y_c$)

Exercício

- Desenhar a borda do círculo de $r = 5$ e centro em $(6,3)$

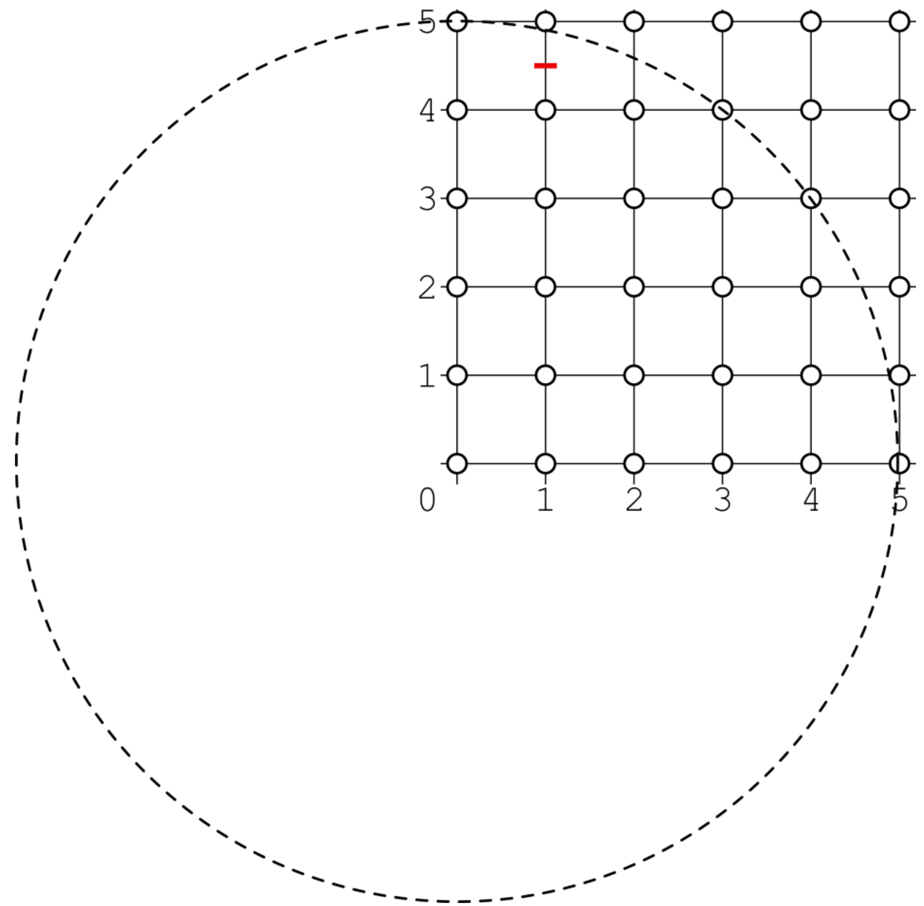
- Inicialmente considera-se o desenho do círculo com mesmo raio e centro em $(0,0)$

- O erro inicial fica $e = -5$ e o primeiro pixel já pode ser pintado em $(0, r)$



Exercício

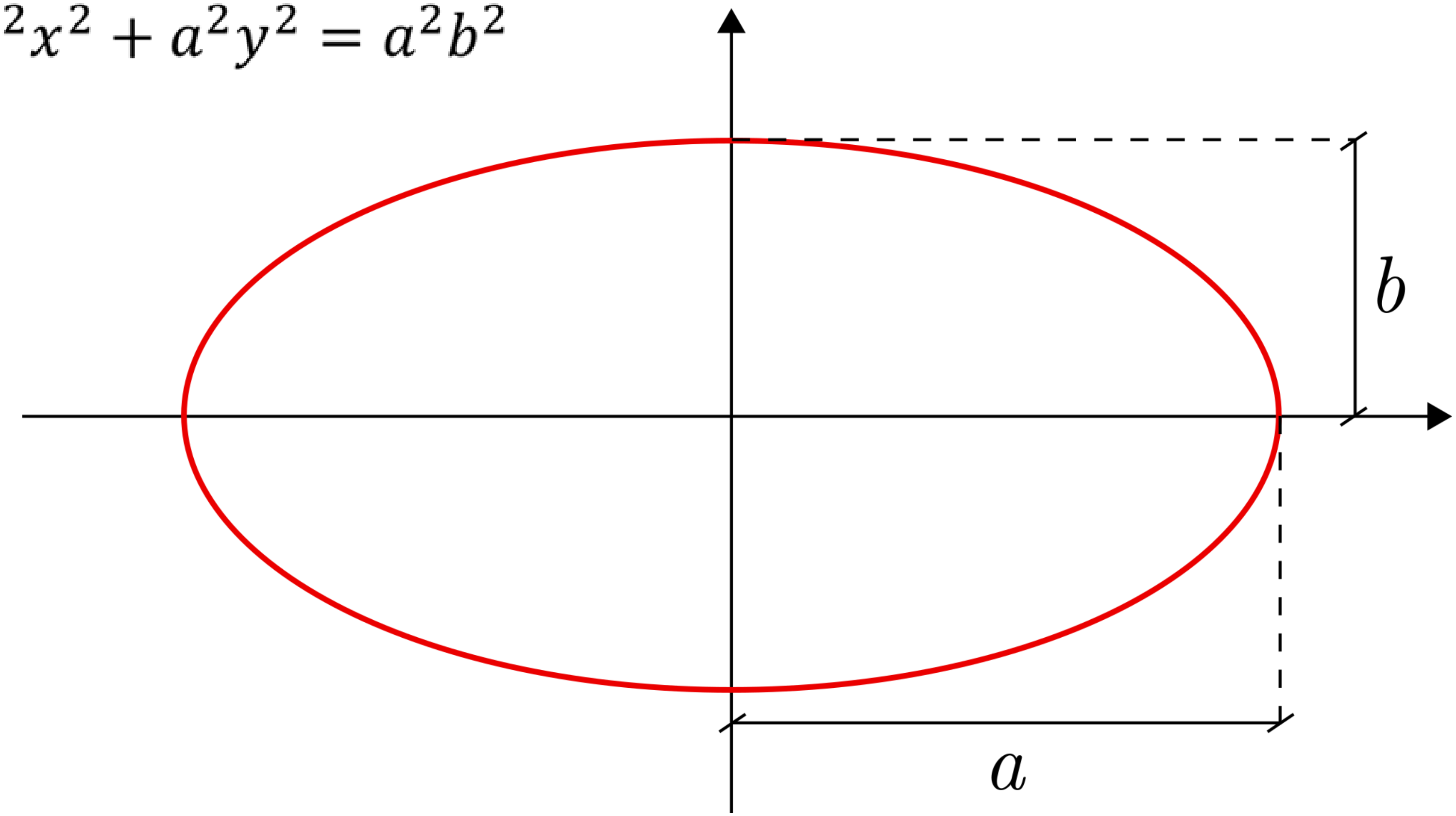
x	e	y
0	$\mathbf{-5}$	5
1	$-5 + 2 \times 0 + 1 = \mathbf{-4}$	5
2	$-4 + 2 \times 1 + 1 = \mathbf{-1}$	5
3	$-1 + 2 \times 2 + 1 = 4$ $4 - 2 \times 5 + 2 = -4$	4
4	$-4 + 2 \times 3 + 1 = 3$ $3 - 2 \times 4 + 2 = -2$	3



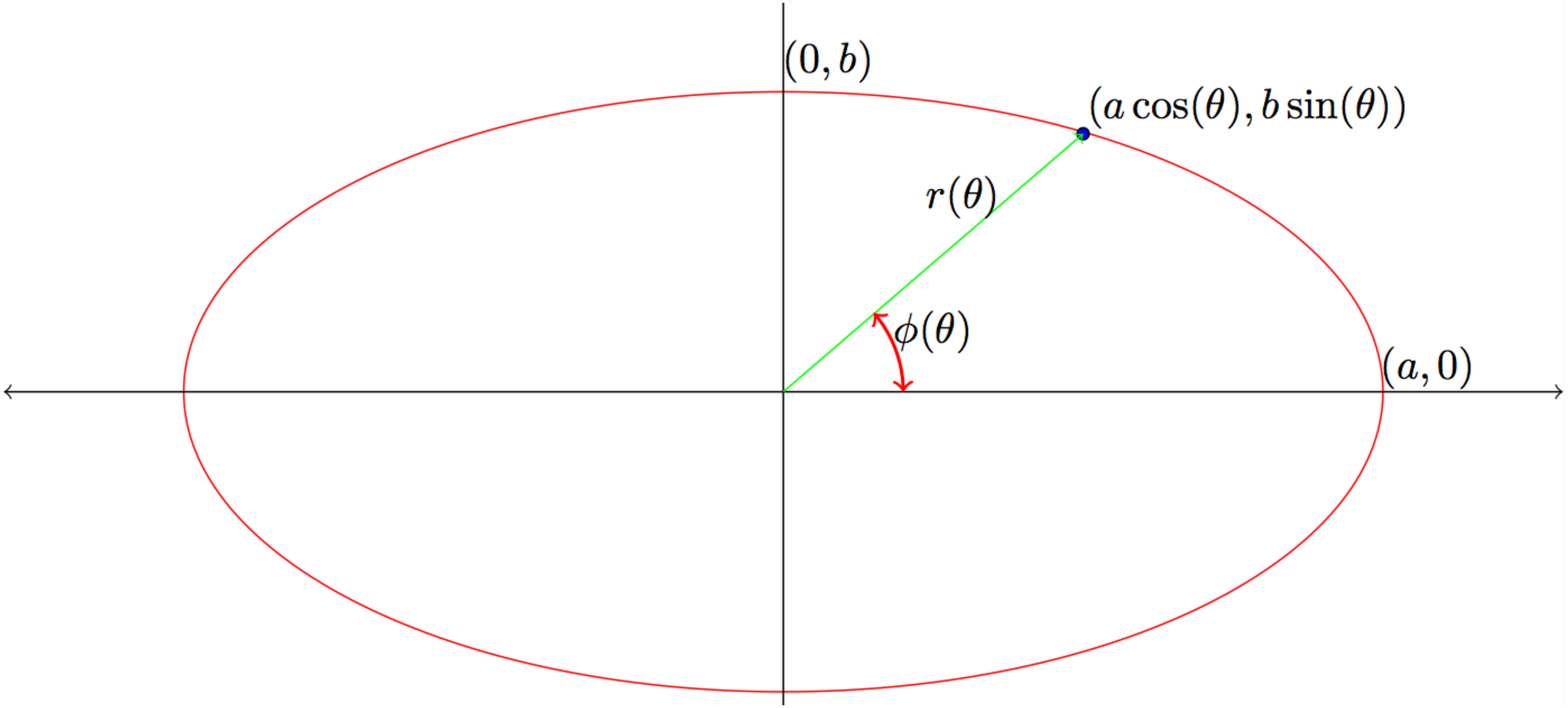
Elipses

Elipse

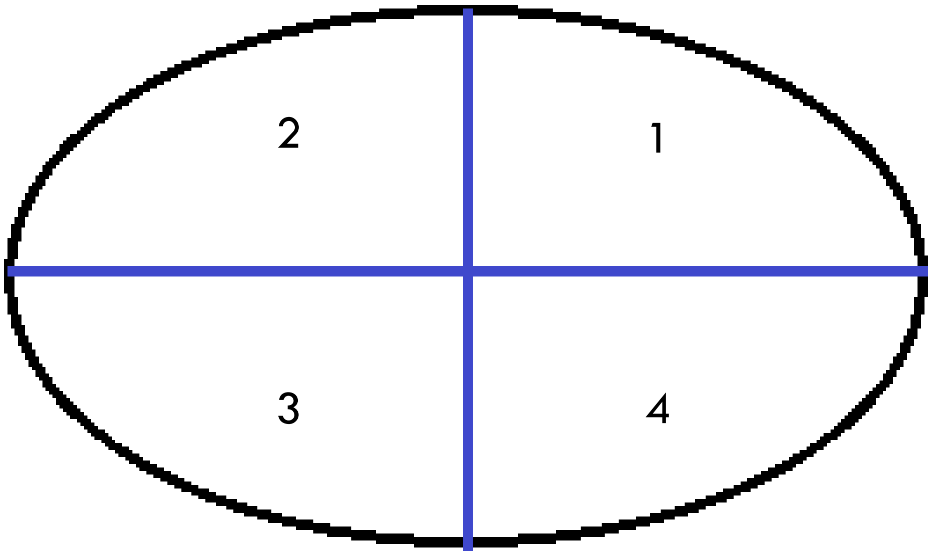
- Matematicamente:
- $b^2x^2 + a^2y^2 = a^2b^2$



Em coordenadas Polares



Simetria na Elipse



Q	x_n	y_n
1	x	y
2	$-x$	y
3	x	$-y$
4	$-x$	$-y$

Problemas

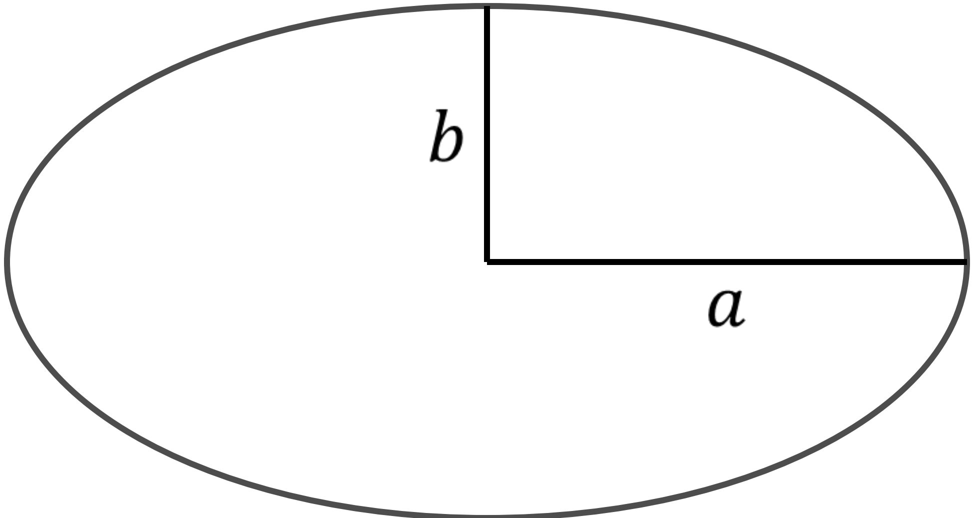
- Temos os mesmos problemas relacionados ao desenho do Círculo

- Usando a fórmula da elipse
  - Potenciação
  - Raiz Quadrada

- Usando Coordenadas Polares
  - Encontrar passo para o ângulo σ
  - Utilização de seno e cosseno na fórmula

Algoritmo do Ponto Médio

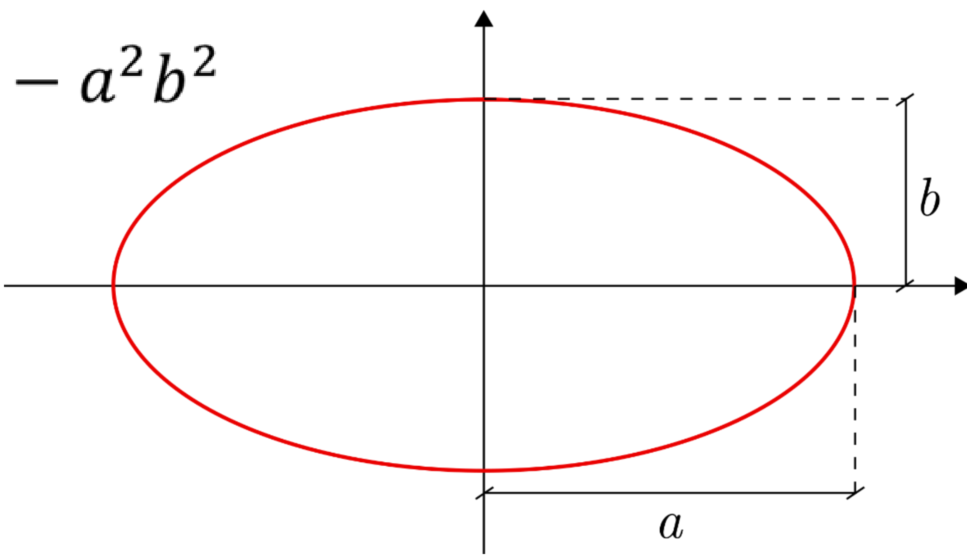
- Outra variação do algoritmo de Bresenham
- Armazena a^2 e b^2 antes no loop
- Utiliza apenas soma de inteiros e multiplicação por 2 dentro do loop



Algoritmo do Ponto Médio

- Da mesma forma que nos alg. anteriores, utiliza-se um erro calculado recursivamente

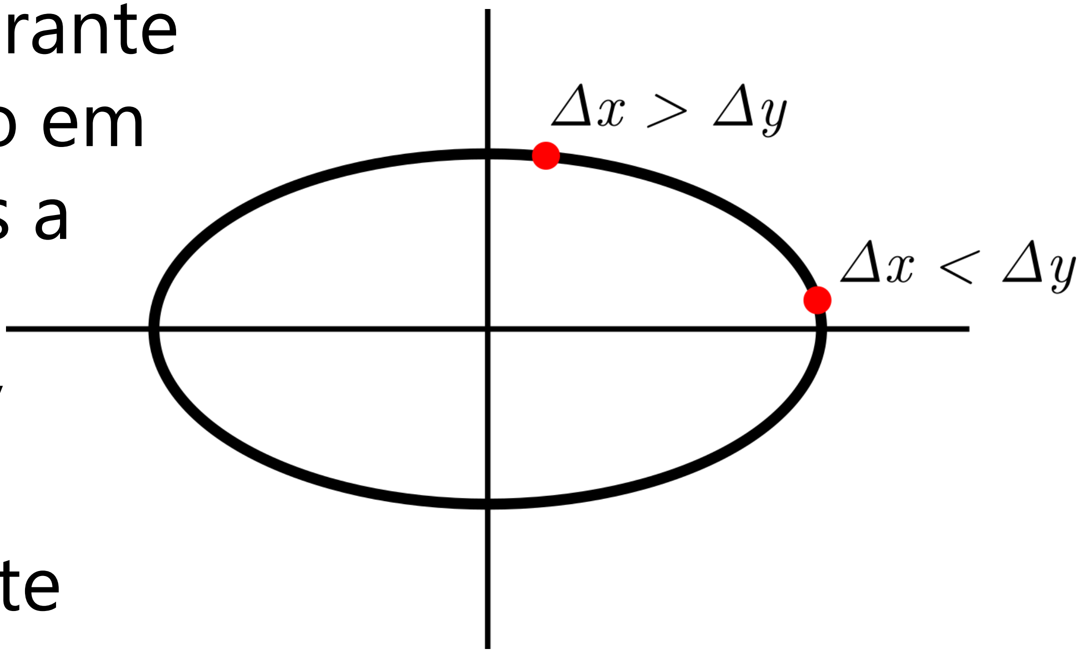
- Para isso define-se a função do erro da seguinte forma:
  - $f(x, y) = b^2x^2 + a^2y^2 - a^2b^2$



Algoritmo do Ponto Médio

- O algoritmo será executado no primeiro quadrante e espelhado para os demais

- O primeiro quadrante deve ser dividido em duas partes, pois a 'velocidade' da variação de x e y muda dentro do mesmo quadrante

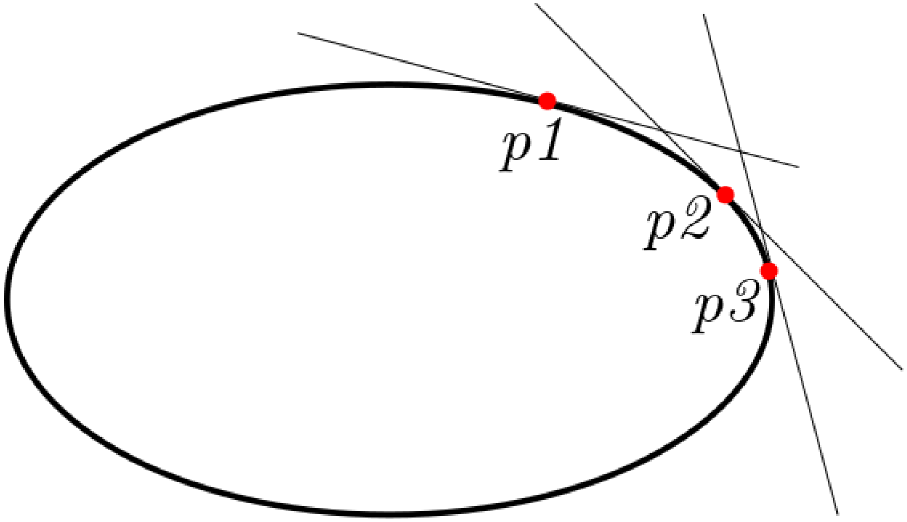


Algoritmo do Ponto Médio

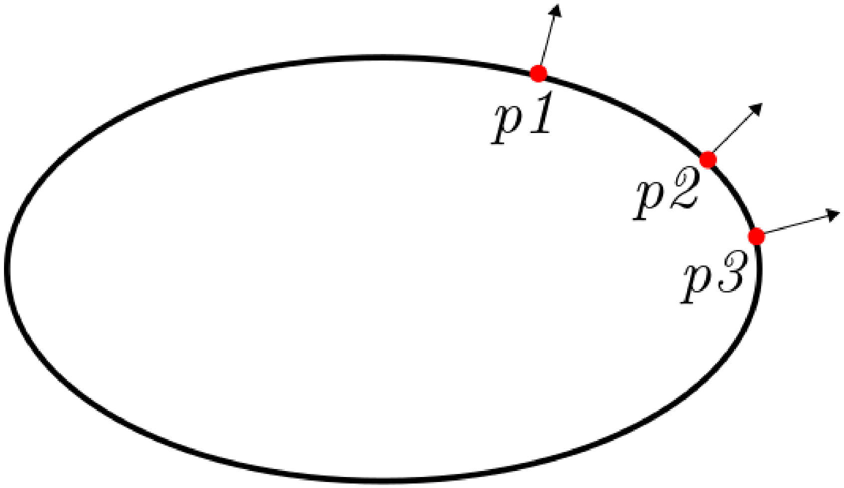
- Para utilizar uma estratégia semelhante aos algoritmos anteriores deve-se dividir a elipse em duas partes
  - A 1ª parte com x variando mais que y
  - E a 2ª parte com y variando mais que x
- Como encontrar o meio entre essas partes?

Algoritmo do Ponto Médio

- Para resolver, podemos usar a noção da reta tangente

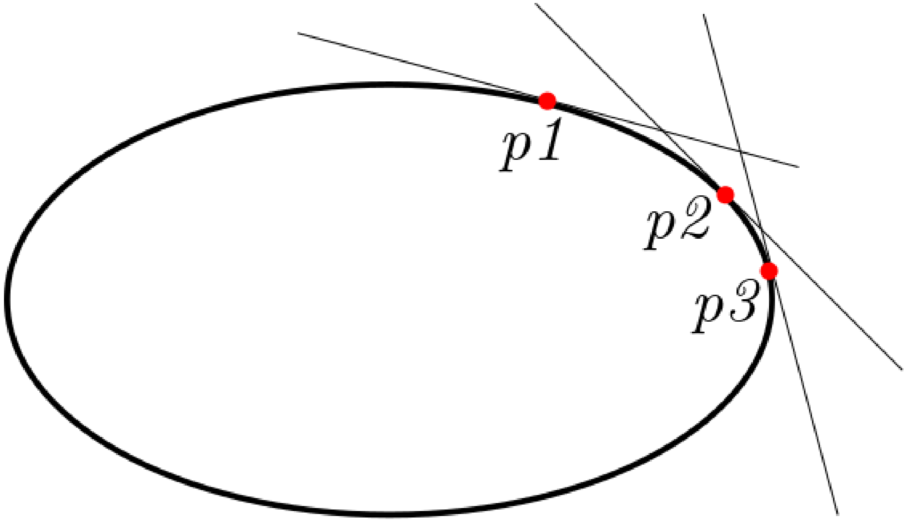


- Ou o vetor gradiente, uma vez que este é perpendicular à reta tangente

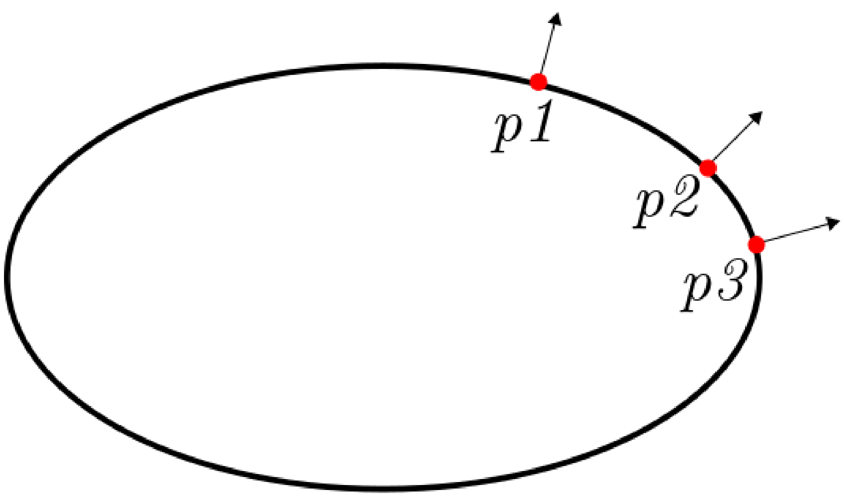


Algoritmo do Ponto Médio

- O ponto de separação das regiões é $\sim p2$, onde o coeficiente angular da reta tangente é $m = -1$



- De forma análoga, quando o coeficiente angular do vetor gradiente é igual a 1



Algoritmo do Ponto Médio

- O algoritmo do ponto médio utiliza o vetor gradiente para essa decisão

- O vetor gradiente pode ser calculado da seguinte forma:
  - $\nabla f(x,y) = \frac{\partial f(x,y)}{\partial x} i + \frac{\partial f(x,y)}{\partial y} j$
  - $\nabla f(x,y) = 2b^2 xi + 2a^2 yj$

$$\boxed{f(x,y) = b^2x^2 + a^2y^2 - a^2b^2}$$

Algoritmo do Ponto Médio

- O vetor gradiente precisa ser atualizado a cada ponto, portanto deve-se obter sua forma recursiva

$$\frac{\partial f(x,y)}{\partial x} = 2b^2x$$
$$\frac{\partial f(x,y)}{\partial y} = 2a^2y$$
$$f(x,y) = b^2x^2 + a^2y^2 - a^2b^2$$

- $\frac{\partial f(x+1,y)}{\partial x+1} = 2b^2(x+1)$
- $\frac{\partial f(x+1,y)}{\partial x+1} = 2b^2x + 2b^2$
- $\frac{\partial f(x+1,y)}{\partial x+1} = \frac{\partial f(x,y)}{\partial x} + 2b^2$

- $\frac{\partial f(x,y-1)}{\partial y-1} = 2a^2(y-1)$
- $\frac{\partial f(x,y-1)}{\partial y-1} = 2a^2y - 2a^2$
- $\frac{\partial f(x,y-1)}{\partial y-1} = \frac{\partial f(x,y)}{\partial y} - 2a^2$

Algoritmo do Ponto Médio

- Para simplificar, chamaremos as derivadas parciais de x e y de d_x e d_y

- Lembre-se que as derivadas são as coordenadas do vetor gradiente, portando queremos encontrar o momento que o coeficiente angular desse vetor passa a ser ≤ 1

- Portanto, basta verificar quando $d_x \geq d_y$

$$\boxed{\begin{gathered} d_{x+1} = d_x + 2b^2 \\ d_{y-1} = d_y - 2a^2 \\ f(x,y) = b^2x^2 + a^2y^2 - a^2b^2 \end{gathered}}$$

Algoritmo do Ponto Médio

- Na primeira região da elipse decide-se entre os pixels $(x+1, y)$ e $(x+1, y-1)$

- Portanto precisamos encontrar a forma recursiva da função de erro para $f(x+1, y)$ e $f(x, y-1)$

$$d_{x+1} = d_x + 2b^2$$
$$d_{y-1} = d_y - 2a^2$$
$$f(x, y) = b^2x^2 + a^2y^2 - a^2b^2$$

Algoritmo do Ponto Médio

- $f(x+1,y) = b^2(x+1)^2 + a^2y^2 - a^2b^2$
- $f(x+1,y) = b^2(x^2+2x+1) + a^2y^2 - a^2b^2$
- $f(x+1,y) = b^2x^2 + 2b^2x + b^2 + a^2y^2 - a^2b^2$
- $f(x+1,y) = f(x,y) \ + 2b^2x + b^2$
- $f(x+1,y) = f(x,y) \ + d_x + b^2$

$$d_x = 2b^2x \ \mid \ d_{x+1} = d_x + 2b^2$$
$$d_y = 2a^2y \ \mid \ d_{y-1} = d_y - 2a^2$$
$$f(x,y) = b^2x^2 + a^2y^2 - a^2b^2$$

Algoritmo do Ponto Médio

- $f(x, y-1) = b^2x^2 + a^2(y-1)^2 - a^2b^2$
- $f(x, y-1) = b^2x^2 + a^2(y^2 - 2y + 1) - a^2b^2$
- $f(x, y-1) = b^2x^2 + a^2y^2 - 2a^2y + a^2 - a^2b^2$
- $f(x, y-1) = f(x, y) - 2a^2y + a^2$
- $f(x, y-1) = f(x, y) - d_y + a^2$

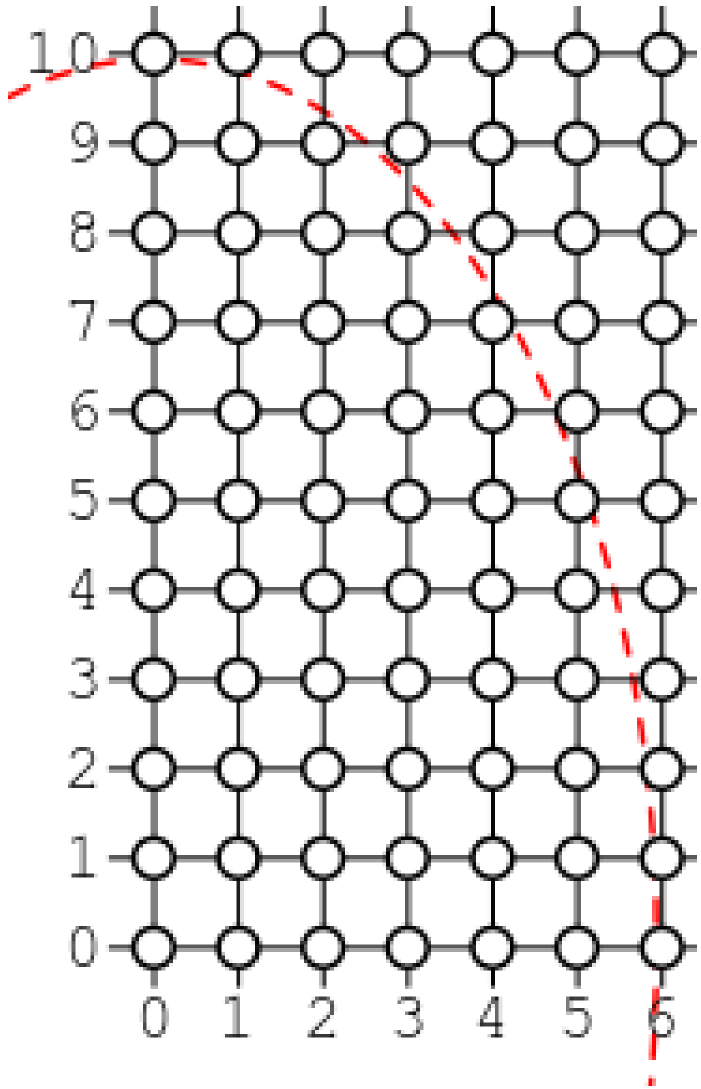
$$d_x = 2b^2x \quad | \quad d_{x+1} = d_x + 2b^2$$
$$d_y = 2a^2y \quad | \quad d_{y-1} = d_y - 2a^2$$
$$f(x, y) = b^2x^2 + a^2y^2 - a^2b^2$$
$$f(x+1, y) = f(x, y) + 2b^2x + b^2$$

Algoritmo do Ponto Médio

- O erro deve ser inicializado para (x,y-1/2) para a 1ª região
- E (x+1/2, y) para a segunda região

- Isso porque na 1ª região sempre se incrementa x e utiliza-se o ponto médio para decidir se decrementa ou não y

- Já na 2ª região sempre se decrementa y e utiliza-se o ponto médio para decidir se incrementa ou não x



Algoritmo do Ponto Médio

- Para 1ª região podemos simplificar o erro inicial pois sabemos os valores iniciais de x e y

$$d_{x+1} = d_x + 2b^2$$
$$d_{y-1} = d_y - 2a^2$$
$$f(x,y) = b^2x^2 + a^2y^2 - a^2b^2$$
$$f(x+1,y) = f(x,y) + 2b^2x + b^2$$
$$f(x,y-1) = f(x,y) - 2a^2y + a^2$$

- $f\left(0, b-\frac{1}{2}\right) = b^2 0^2 + a^2\left(b-\frac{1}{2}\right)^2 - a^2b^2$
- $f\left(0, b-\frac{1}{2}\right) = +a^2\left(b^2 - b + \frac{1}{4}\right) - a^2b^2$
- $f\left(0, b-\frac{1}{2}\right) = +a^2b^2 - a^2b + \frac{a^2}{4} - a^2b^2$
- $f\left(0, b-\frac{1}{2}\right) = -a^2b + \frac{a^2}{4}$

- Mas não sabemos qual (x,y) se inicia a 2ª região, então utilizamos a fórmula diretamente

Algoritmo do Ponto Médio

```
ellipse(float a, float b, float xc, float yc){

    //a e b ao quadrado
    a_sq = a*a
    b_sq = b*b

    x = 0
    y = b

    //derivadas parciais de x e y
    dx = 2*b_sq*x
    dy = 2*a_sq*y

```

$$\boxed{\begin{aligned} d_x &= 2b^2x \\ d_y &= 2a^2y \end{aligned}}$$

Algoritmo do Ponto Médio

```
    //erro para f(0, b-0.5)
    e = - b*a_sq + a_sq*0.25

    //Região 1
    while(dx < dy){

        print4(x,y,xc,yc)

        x++
        e += dx + b_sq
        dx += 2*b_sq

        if(e > 0){
            y--
            e += a_sq - dy
            dy- = 2*a_sq
        }

    }

```

$$d_{x+1} = d_x + 2b^2$$
$$d_{y-1} = d_y - 2a^2$$
$$f(x,y) = b^2x^2 + a^2y^2 - a^2b^2$$
$$f(x+1,y) = f(x,y) + 2b^2x + b^2$$
$$f(x,y-1) = f(x,y) - 2a^2y + a^2$$
$$f\left(0, b - \frac{1}{2}\right) = -a^2b + \frac{a^2}{4}$$

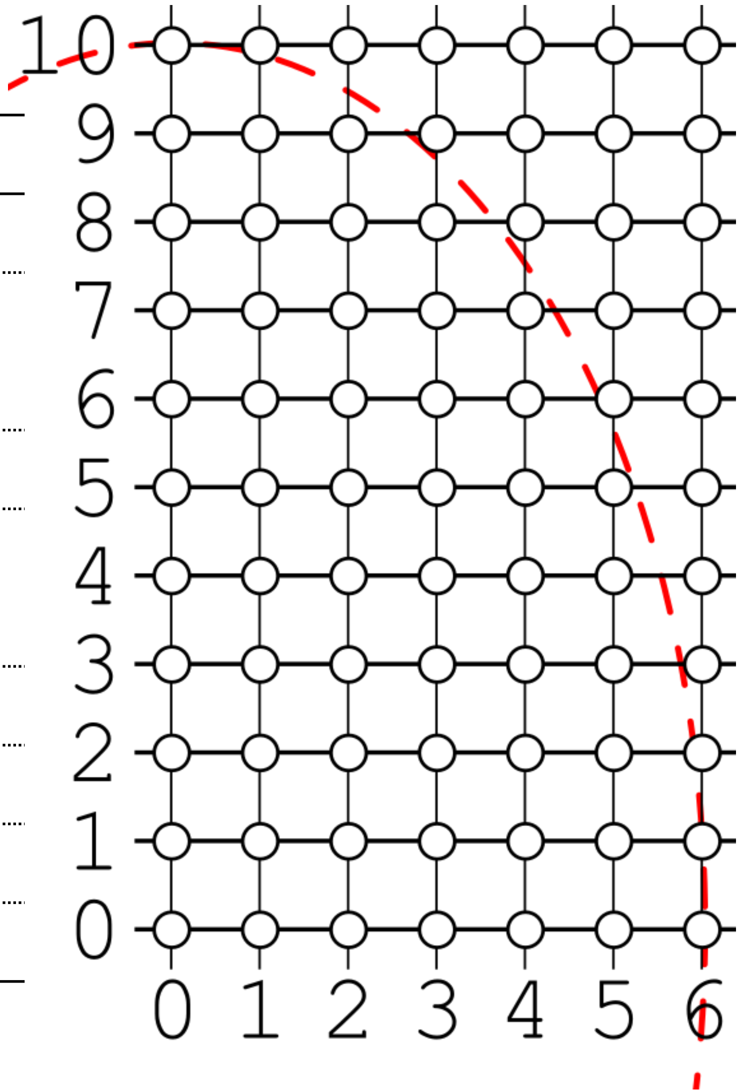
Algoritmo do Ponto Médio

```
    //Erro da região 2
    e = b_sq*((x+0.5)*(x+0.5)) + a_sq*y*y - a_sq*b_sq

    //Região 2
    while(y>=0){
        print4(x,y,xc,yc)

        y--
        e += a_sq - dy
        dy-= 2*a_sq

        if(e < 0){
            x++
            e += dx + b_sq
            dx += 2*b_sq
        }

    }

}
```

$$d_{x+1} = d_x + 2b^2$$
$$d_{y-1} = d_y - 2a^2$$
$$f(x,y) = b^2x^2 + a^2y^2 - a^2b^2$$
$$f(x+1,y) = f(x,y) + 2b^2x + b^2$$
$$f(x,y-1) = f(x,y) - 2a^2y + a^2$$
$$f\left(0, b - \frac{1}{2}\right) = -a^2b + \frac{a^2}{4}$$

Exemplo

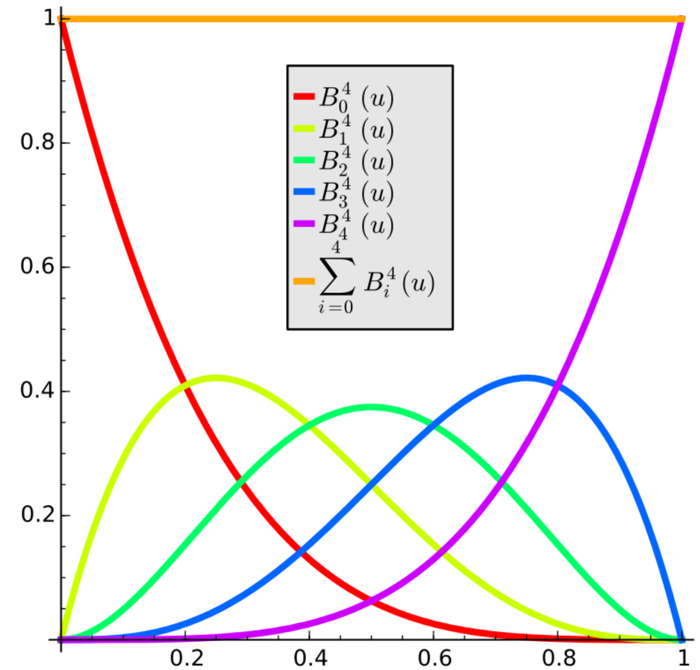
x	dx	dy	e	y
4	800	504	189	**7**
4	800	432	-279	6
5	1000	432	621	**6**
5	1000	360	225	**5**
5	1000	288	-99	4
6	1200	288	1001	**4**
6	1200	216	749	**3**
6	1200	144	569	**2**
6	1200	72	461	**1**
6	1200	0	425	**0**



Curvas

Polinomial de Bernstein

- Curvas arbitrárias podem ser representadas pela polinomial de Bernstein
  - São polinômios bases combinados linearmente que quando somados dentro do escopo entre [0,1] também tem saída entre



- A fórmula do polinomial de Bernstein foi aplicada pela 1ª vez para computação gráfica por dois estudiosos para desenhar curvas em carros na Renault e Citröen

Polinomial de Bernstein

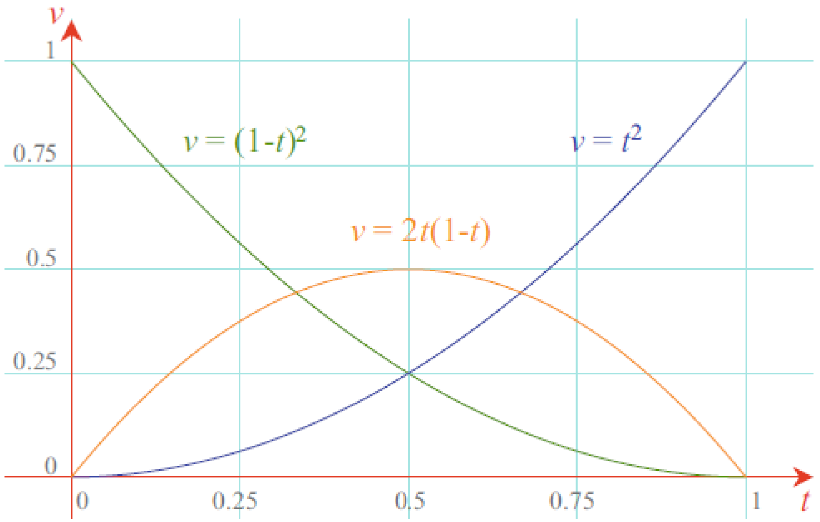
- Cada polinômio é descrito pela seguinte fórmula

$$B_i^n(t) = \binom{n}{i} t^i (1-t)^{n-i}$$

- Onde:
  - n é o grau da polinomial e i determina o i-ésimo polinômio base
  - $\binom{n}{i}$ é uma combinação simples dado pela fórmula:
    - $\binom{n}{i} = \frac{n!}{(n-i)!i!}$

Polinomial de Bernstein

- A ideia básica é combinar valores diferente a cada polinômio base, sendo assim, a soma ponderada em um determinado x combina linearmente os valores.

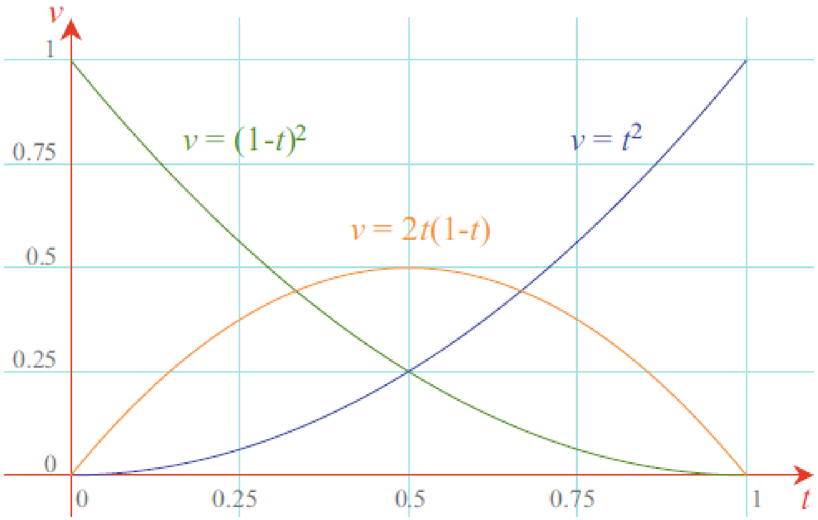


$$1 = (1-t)^2 + 2t(1-t) + t^2$$

- Por exemplo, usando a polinomial quadrática de Bernstein temos:

Polinomial de Bernstein

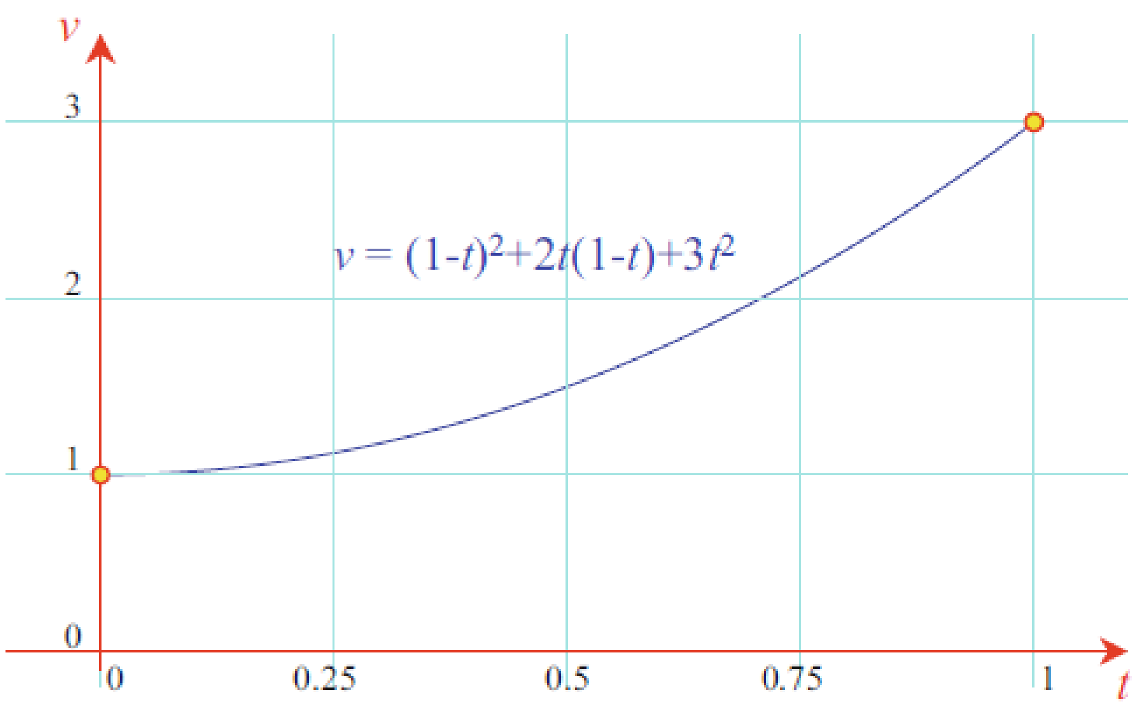
- Então podemos interpolar dois valores v_1 e v_2 ao multiplicar esses valores aos polinômios base para gerar um valor interpolado v.



$$v = v_1(1-t)^2 + 2t(1-t) + v_2 t^2$$

Polinomial de Bernstein

- $$v = v_1(1-t)^2 + 2t(1-t) + v_2 t^2$$
- Quando consideramos $v_1 = 1$ e $v_2 = 3$ geramos os seguintes valores

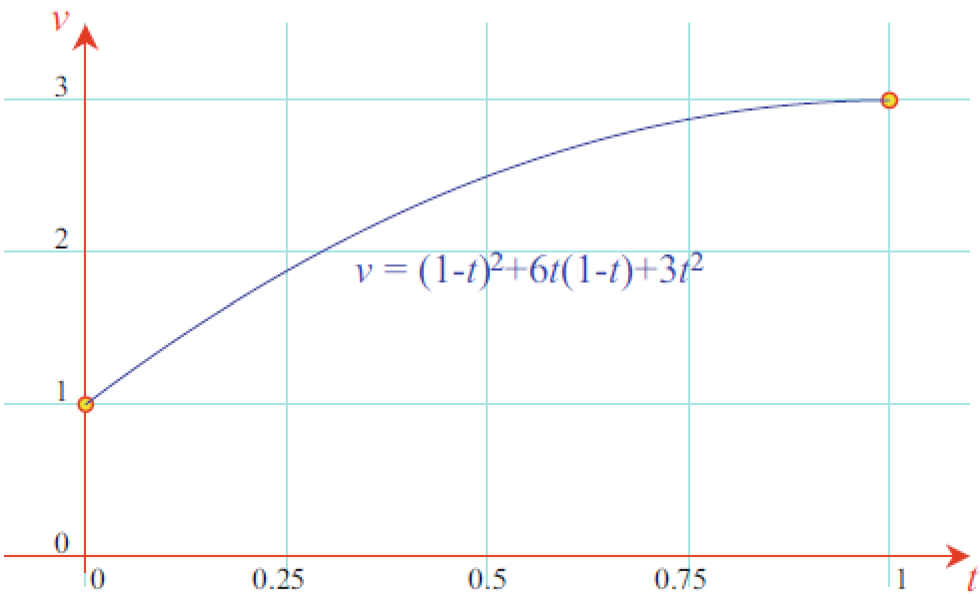


Polinomial de Bernstein

- Na polinomial de grau 2 ainda temos um terceiro termo que pode ser utilizado ao multiplicá-lo por um outro valor v_c.

$$v = v_1(1-t)^2 + v_c 2t(1-t) + v_2 t^2$$

Para:
$v_c = 3$
$v_1 = 1$
$v_2 = 3$



Curvas Bézier

- Baseia-se na interpolação de pontos de controle utilizando a Polinomial de Bernstein.

- As interpolações são parametrizadas por um único parâmetro t.

- Basta considerar que v ao invés de um escalar é um vetor. Por exemplo um ponto 2D. Então a fórmula abaixo representa uma curva Bézier de grau 1:

$$\mathbf{P}(t) = (1-t)\,\mathbf{p}_0 + t\,\mathbf{p}_1, \quad t \in [0,1],$$

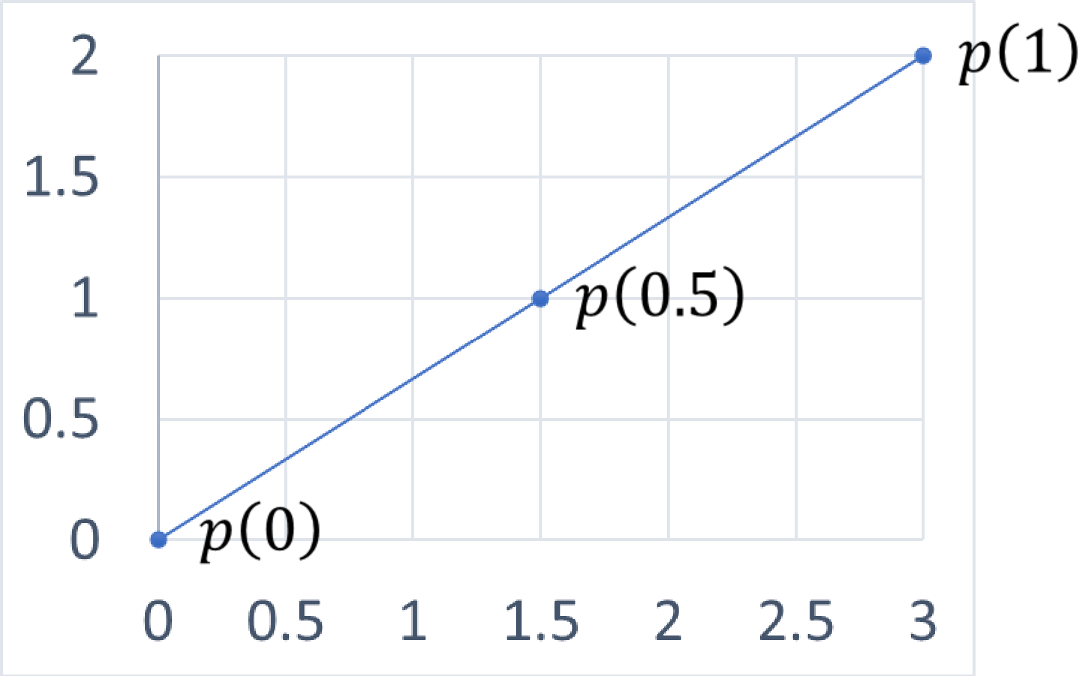
Curvas Bézier

$$\mathbf{P}(t) = (1-t)\,\mathbf{p}_0 + t\,\mathbf{p}_1, \quad t \in [\mathbf{0}, \mathbf{1}],$$

- Por exemplo, para $p_0 = [0,0]$ e $p_1 = [3,2]$ temos:
- $P(0) = (1-0)[0,0] + 0[3,2] = [0,0]$
- $P(0.5) = (1-0.5)[0,0] + 0.5[3,2] = [1.5,1]$
- $P(1) = (1-1)[0,0] + 1[3,2] = [3,2]$

Curvas Bézier

- $p(0) = (1-0)[0,0] + 0[3,2] = [0,0]$
- $p(0.5) = (1-0.5)[0,0] + 0.5[3,2] = [1.5,1]$
- $p(1) = (1-1)[0,0] + 1[3,2] = [3,2]$



Curvas Bézier

- A polinomial de Bernstein pode ser desmembrada de forma recursiva, por exemplo a curva Bézier quadrática (grau 2) pode ser vista da seguinte forma:

$$P_0^2(t) = (1-t)^2 p_0 + 2t(1-t)p_1 + t^2 p_2$$

$$P_0^2(t) = (1-t)^2 p_0 + t(1-t)p_1 + t(1-t)p_1 + t^2 p_2$$

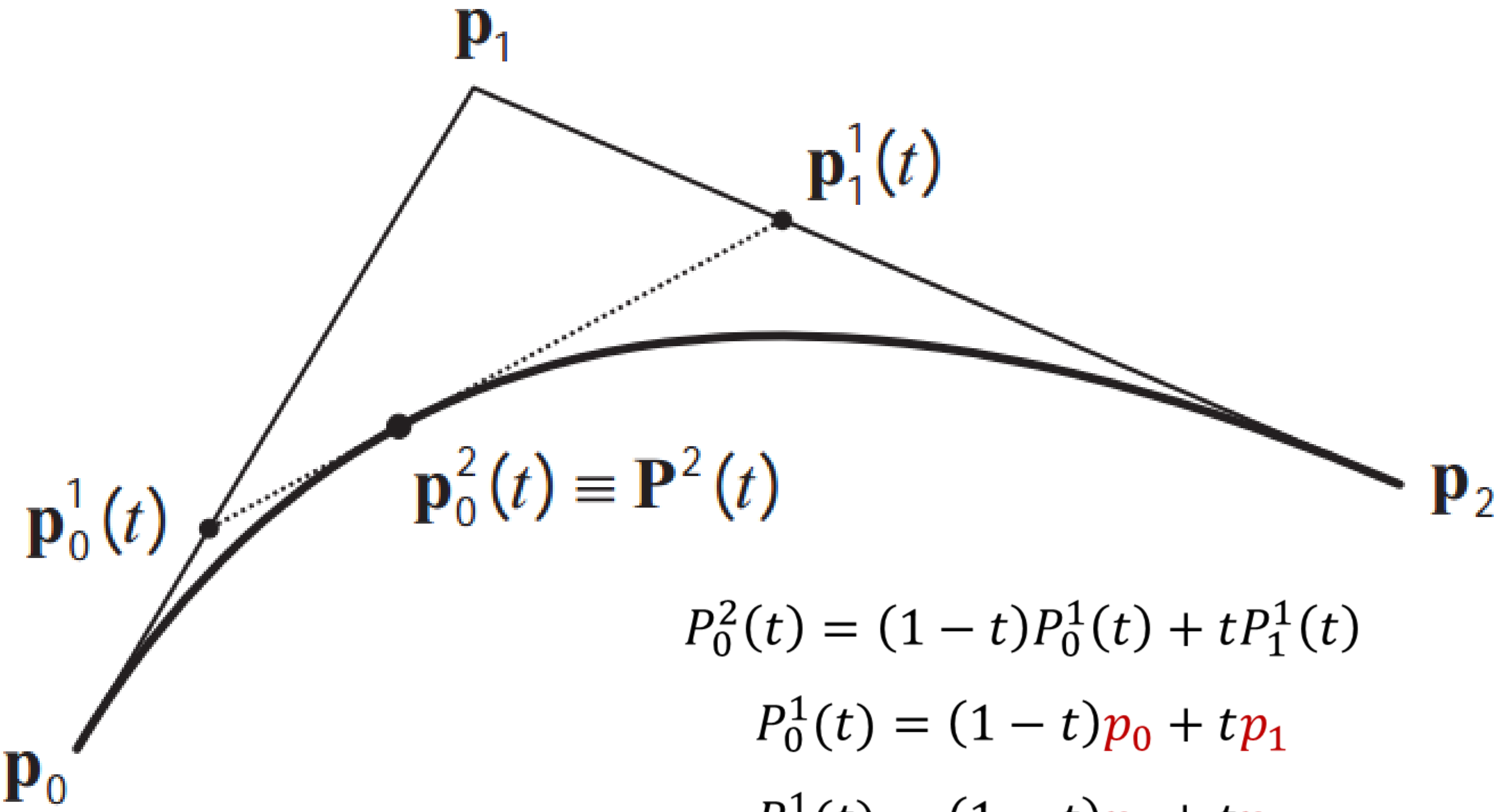
$$P_0^2(t) = (1-t)\big((1-t)p_0 + tp_1\big) + t\big((1-t)p_1 + tp_2\big)$$

$$P_0^2(t) = (1-t)P_0^1(t) + tP_1^1(t)$$

$$P_0^1(t) = (1-t)p_0 + tp_1$$

$$P_1^1(t) = (1-t)p_1 + tp_2$$

Curvas Bézier

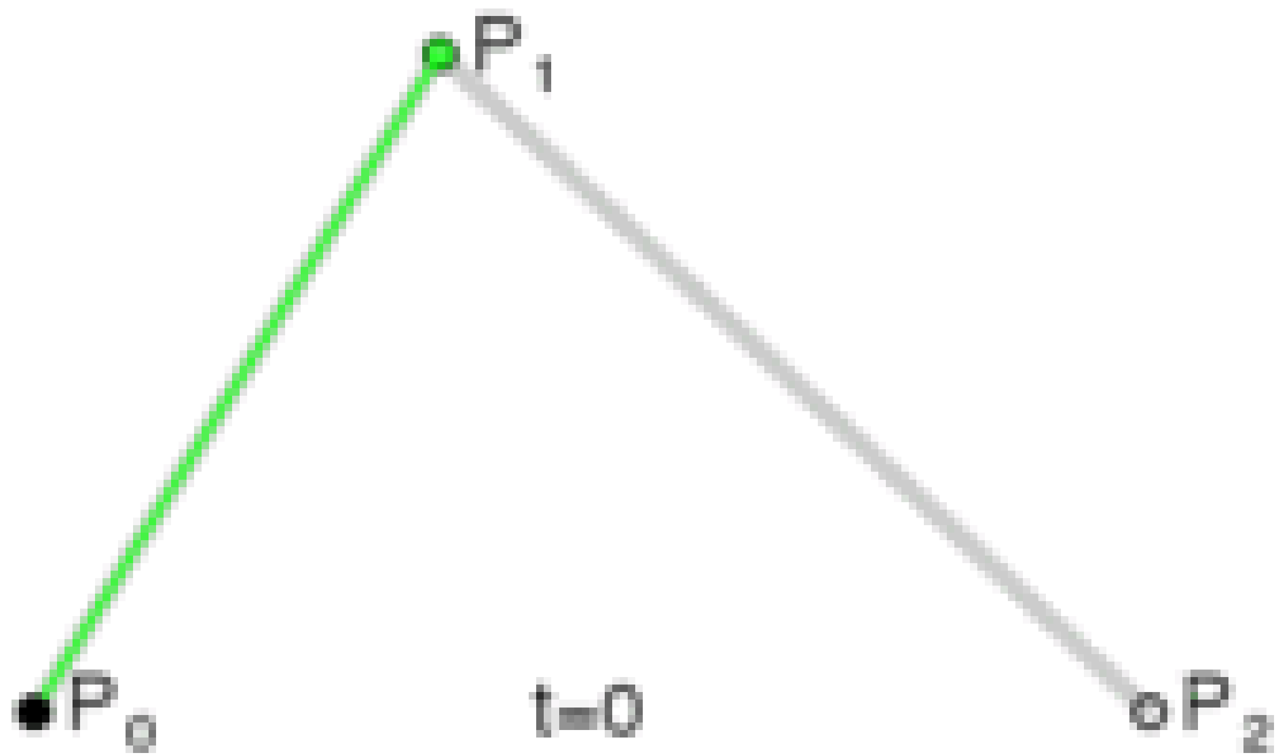


$$P_0^2(t) = (1 - t)P_0^1(t) + tP_1^1(t)$$

$$P_0^1(t) = (1 - t)p_0 + tp_1$$

$$P_1^1(t) = (1 - t)p_1 + tp_2$$

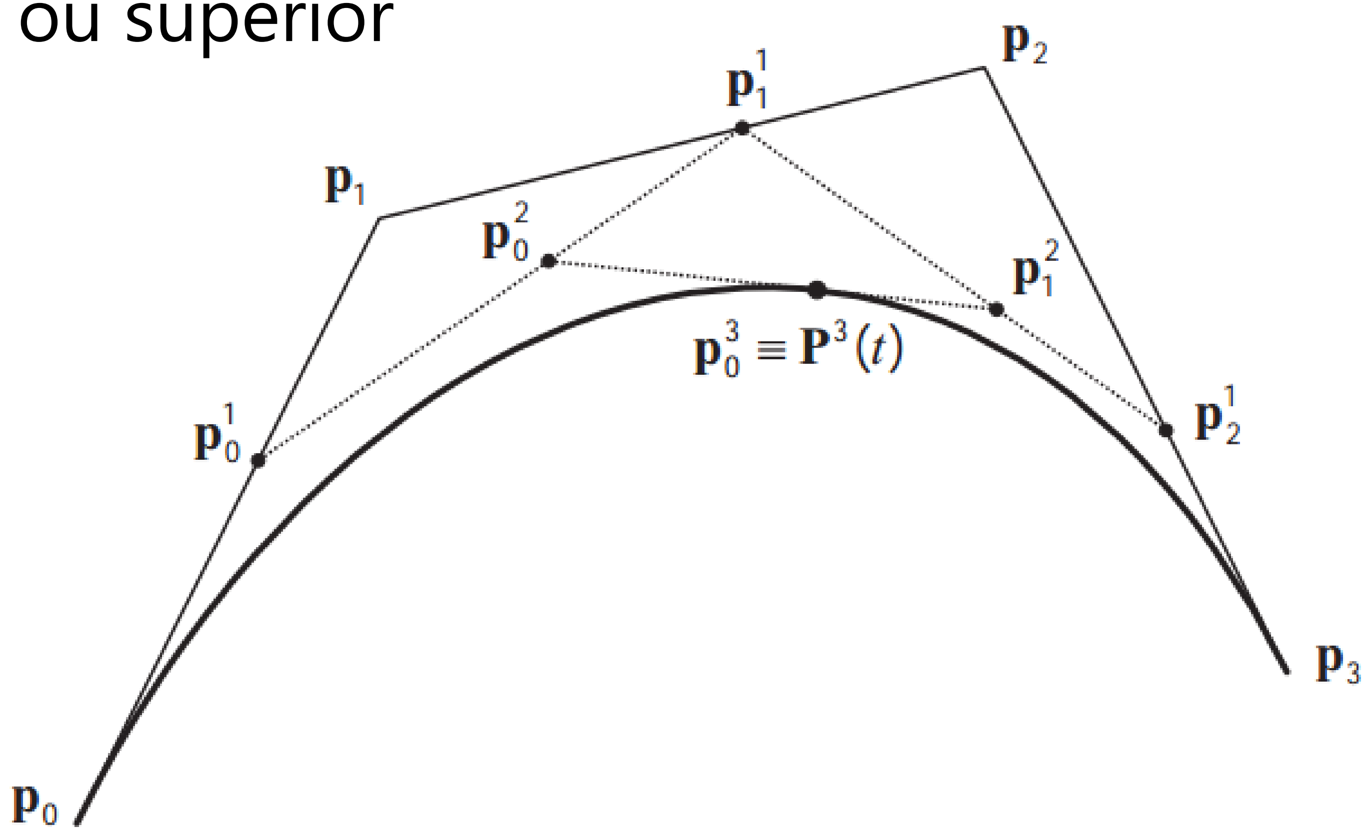
Variando t temos



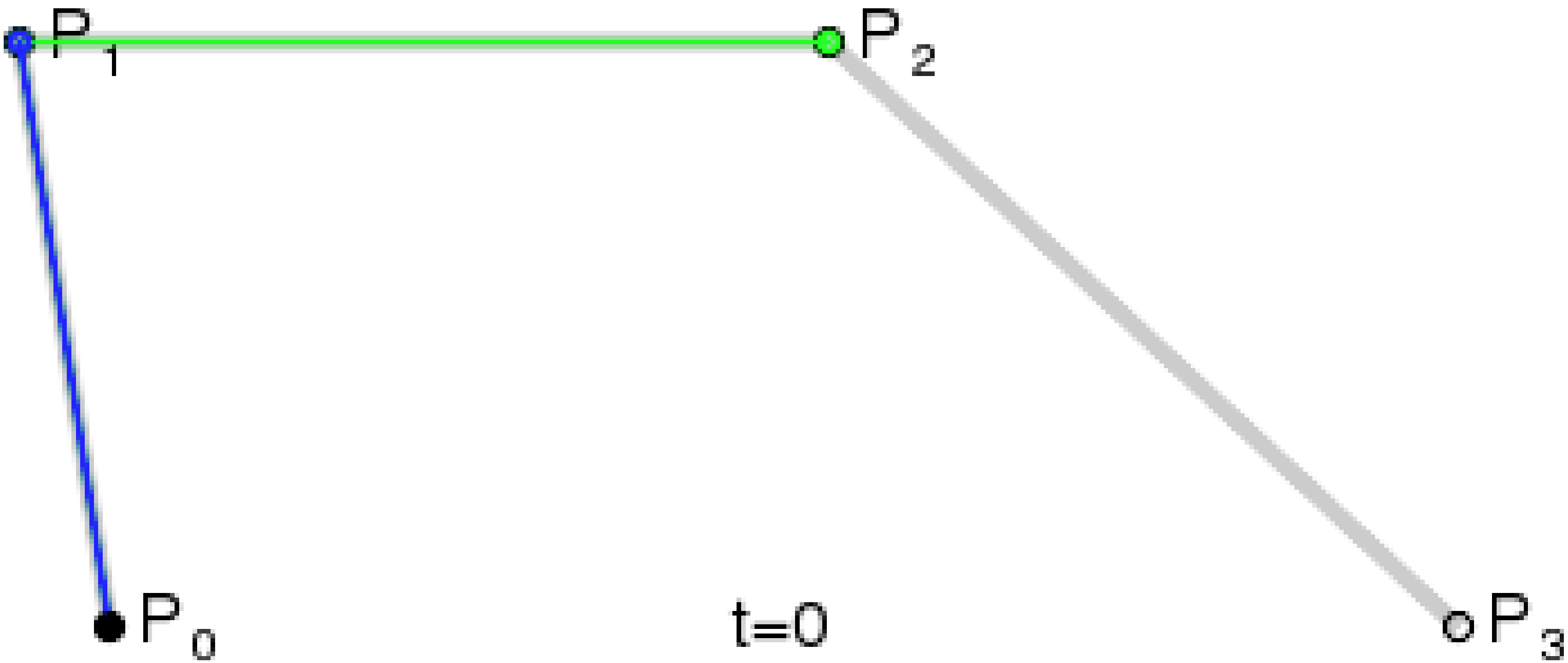
https://pt.wikipedia.org/wiki/Curva_de_B%C3%A9zier#/media/File:B%C3%A9zier_2_big.gif

Bézier Cúbico

- Naturalmente o mesmo desmembramento recursivo se aplica aos polinômios de Bernstain de grau 3 ou superior



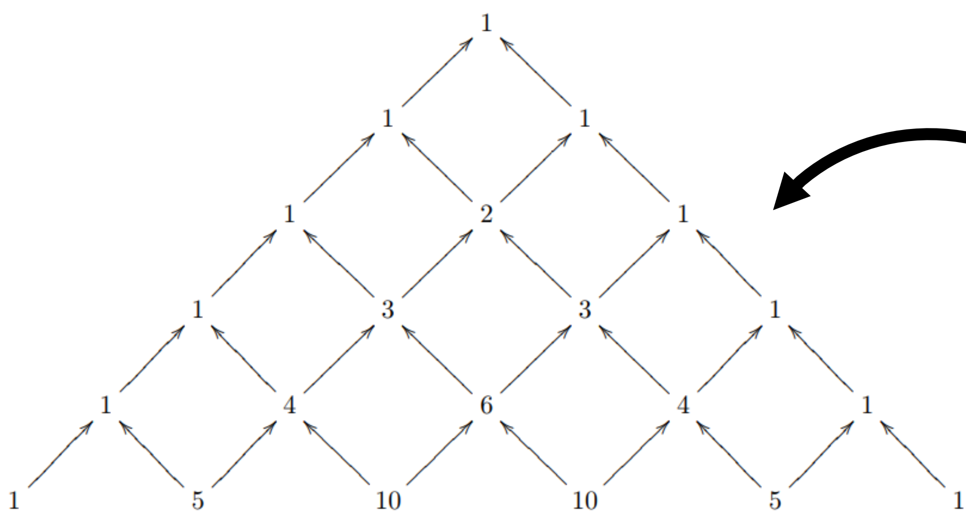
Bézier Cúbico



https://pt.wikipedia.org/wiki/Curva_de_B%C3%A9zier#/media/File:B%C3%A9zier_3_big.gif

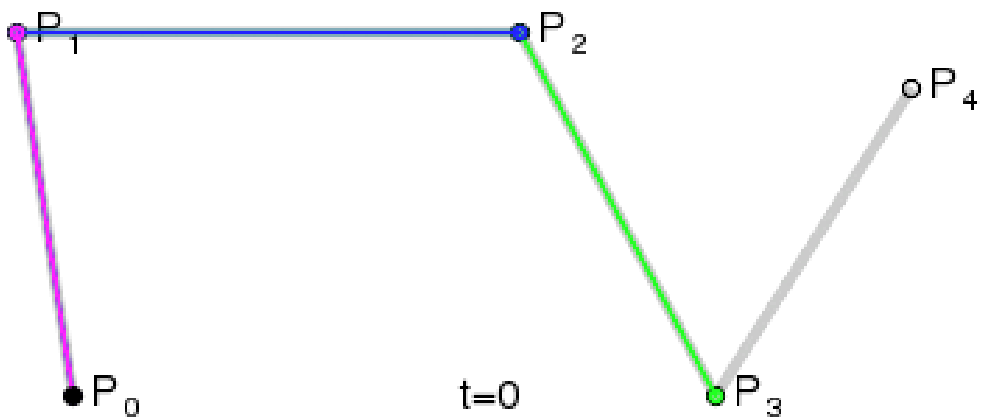
Bézier de Grau n

$$\mathbf{P}^n(t) = \sum_{i=0}^{n} \binom{n}{i} t^i (1-t)^{n-i} \mathbf{p}_i, \quad t \in [0,1].$$

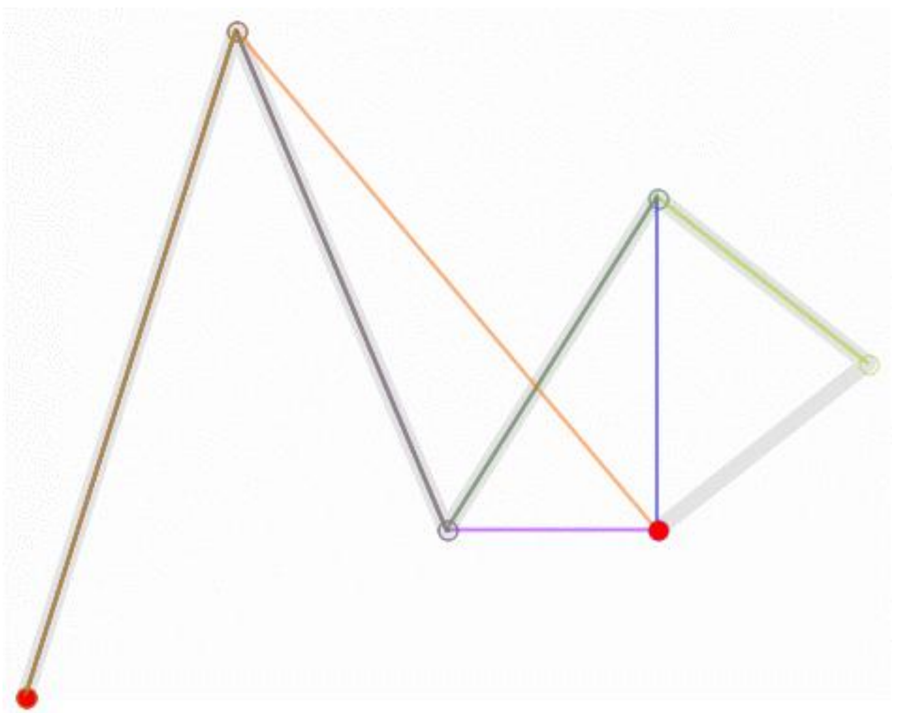


$$\binom{n}{i} = \frac{n!}{i!\,(n-i)!}$$

Bézier de Grau n



https://upload.wikimedia.org/wikipedia/commons/a/a4/B%C3%A9zier_4_big.gif



https://upload.wikimedia.org/wikipedia/commons/0/0b/BezierCurve.gif

Algoritmo de De Casteljau

- 1. Para um determinado valor de t, atribua para cada valor de i

$$p_i^0(t) = pi, \qquad i = 0,1, \dots, n$$

- 2. Realize as seguintes interpolações lineares

$$p_i^r(t) = (1-t)p_i^{r-1}(t) + tp_{i+1}^{r-1}(t), \qquad \begin{matrix} r = 1,2, \dots, n \\ i = 0,1, \dots, n-r \end{matrix}$$

- 3. O valor do ponto correspondente ao valor de t será

$$p^n(t) = p_0^n(t)$$

Algoritmo de De Casteljau

$$
\begin{tikzcd}
\mathbf{p}_0 = \mathbf{p}_0^0 \arrow[r, "1-t"] & \mathbf{p}_0^1 \arrow[r] & \mathbf{p}_0^2 \arrow[r] & \mathbf{p}_0^3 = \mathbf{P}^3(t) \\
\mathbf{p}_1 = \mathbf{p}_1^0 \arrow[r] \arrow[ur, "t"] & \mathbf{p}_1^1 \arrow[r] \arrow[ur] & \mathbf{p}_1^2 \arrow[ur] & \\
\mathbf{p}_2 = \mathbf{p}_2^0 \arrow[r] \arrow[ur] & \mathbf{p}_2^1 \arrow[ur] & & \\
\mathbf{p}_3 = \mathbf{p}_3^0 \arrow[ur] & & &
\end{tikzcd}
$$

Algoritmo de De Casteljau

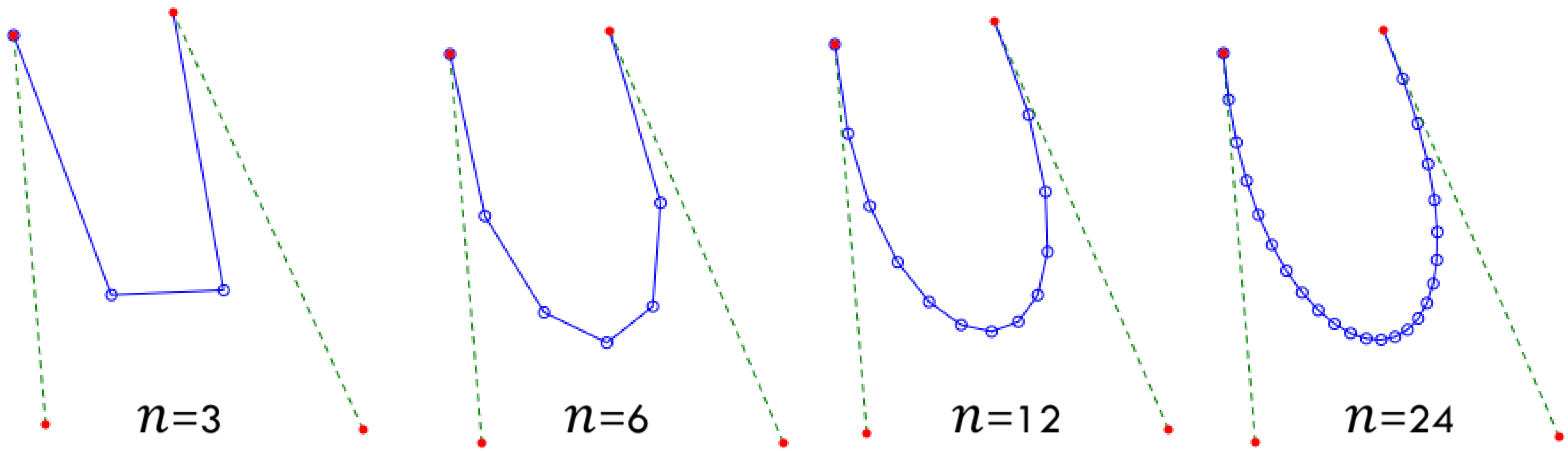
```
Point bezierPoint(int n, Point[] ctrlPts, float
t){
      Point pts[n+1];
      for (i=0; i <= n; i++)
            pts[i] = ctrlPts[i];

      for (r=1; r <= n; r++) {
            for (i=0; i <= n-r; i++) {
                  pts [i] = (1-t)*pts[i] +
t*pts[i+1];
            }
      }
      return pts[0];
}
```

Bézier com Cordas

- Consistem em desenhar uma curva Bézier com várias linhas retas (cordas)
- Dividi-se t em n partes então tem-se $n + 1$ pontos
- Então se desenha n retas com o algoritmo de polilinhas para entre esses $n + 1$ pontos

Bezier com Cordas

- https://observablehq.com/@gustavoresque/desenhando-bezier-com-cordas



Caminho (Path)

- Semelhante à polilinha reune diversos segmentos de curvas, onde o ponto inicial da i-ésima curva é igual a ponto final da (i-1)-ésima curva

- Geralmente os softwares de desenho gráfico utilizam esse princípio para definir curvas alta complexidade, por exemplo, várias curvas Bézier cúbicas

Caminho (Path)

- Exemplo no software de desenho Inkscape utilizando uma imagem SVG

- Utiliza uma sequência curvas Bézier Cúbicas