Computação Gráfica e Realidade Virtual

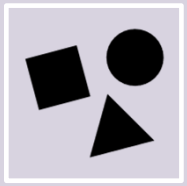
Rasterização de retas

Prof. Dr. Fernando Kakugawa

fernando.kakugawa@animaeducacao.com.br

Universidade Anhembi Morumbi

Primitivas Gráficas



São comandos e funções que manipulam e alteram os elementos gráficos de uma imagem

Elementos básicos que compõem uma imagem

Pontos, segmentos de reta e círculos

Pixel ou Ponto

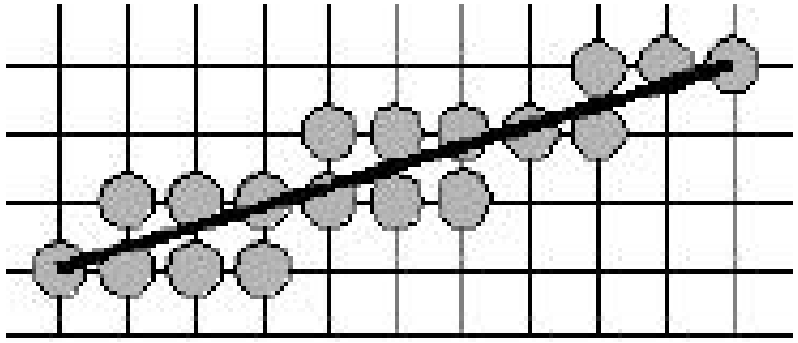
- É uma unidade gráfica fundamental
- **Propriedades:**
  - posição no plano gráfico
  - cor
- Um pixel é um objeto concreto e não abstrato
  - Pode ter uma superfície, que é dada pela construção física do dispositivo no qual está sendo desenhado

Rasterização de Retas

- O traçado de primitivas gráficas elementares requer a construção de algoritmos capazes de determinar **quais pixels serão alterados** para simular a aparência do elemento gráfico desejado
  - Denominados: Algoritmos de Conversão Matricial
- Um elemento comum a ser traçado é o segmento de reta
  - Diversas aplicações de computação gráfica têm como primitiva básica o segmento de reta
  - Tais algoritmos estão disponíveis em **hardware**
  - O hardware recebe as coordenadas dos pontos extremos e calcula os pixels intermediários que farão parte do desenho da reta na tela

Segmentos de reta

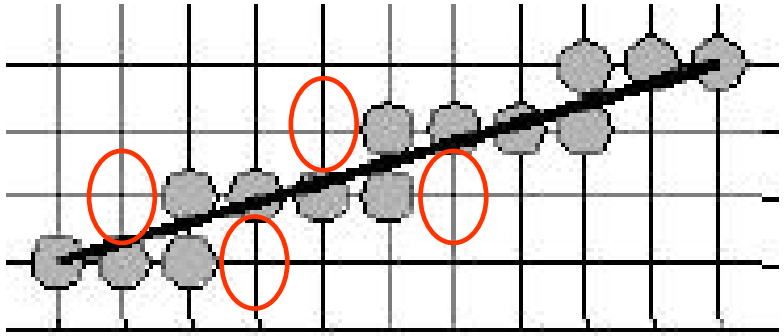
- Adotou-se o critério de selecionar os dois pixels imediatamente acima e abaixo do ponto de intersecção do segmento com cada vertical, menos quando o segmento passa por um pixel



- **Restrição:** dessa forma obtém-se linhas densas, como se o segmento fosse espesso

Segmentos de reta

- **Para melhorar:**
  - O critério foi selecionar todos os pixels cujas coordenadas são obtidas arredondando-se os valores das coordenadas de algum ponto do segmento (ceil, floor, trunc ou round)



  - **Restrição:** com segmentos a 45º o critério produz resultados semelhantes ao anterior

Algoritmo de DDA
Digital Differential Analyzer

- Estratégia simples: Uso da equação da reta

$$y = mx + B$$

  - Onde: `m = (y2 - y1) / (x2 - x1)` //inclinação (coeficiente angular)
  - Onde: `B = (y1 - m*X1)` //intersecção eixo y

$$y = y1 + m*(x - x1)$$

- Basta verificar qual é o inteiro mais próximo das coordenadas geradas pela equação da reta
- Inicia uma repetição de (x1,y1) até (x2,y2)
- Variando o **x** e calculando o valor de **y** no intervalo

Algoritmo de DDA

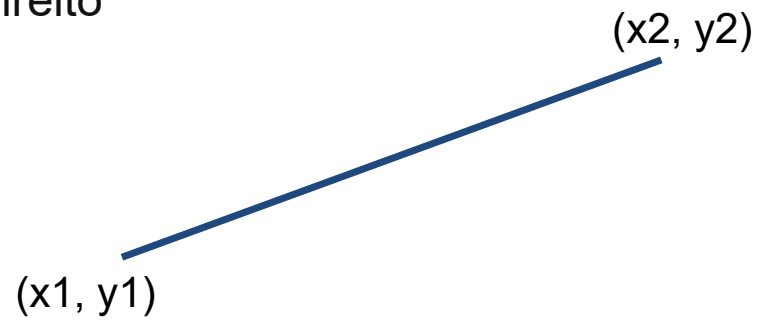
```
void linha(int x1, int x2, int y1, int y2, int cor){
    float y, m;
    int x;
    m = (y2 - y1) / (x2- x1); //coeficiente angular

    for(x = x1; x <= x2; x++){
        y = y1 + m * (x - x1);
        writePixel(x , round(y), cor);
    }
}
```

- **Problema:** trabalha com multiplicação e soma de ponto flutuante na determinação dos pontos que compõem a reta, além de arredondamentos
- Os cálculos com ponto flutuante são lentos quando comparado com inteiros

Algoritmo do Ponto Médio

- Um algoritmo equivalente ao da equação da reta, mas que dispensa as operações em ponto flutuante foi proposto por **Bresenham** (Década de 60)
- **Bresenham** desenvolveu um algoritmo clássico que usa apenas variáveis inteiras, e que permite que o cálculo seja feito de forma **incremental**
  - Assume que a inclinação da linha está entre 0 e 1 (i.e. 1º otctante: 0º - 45º)
  - O ponto (x1, y1) é o inferior esquerdo e o ponto (x2, y2) é o superior direito

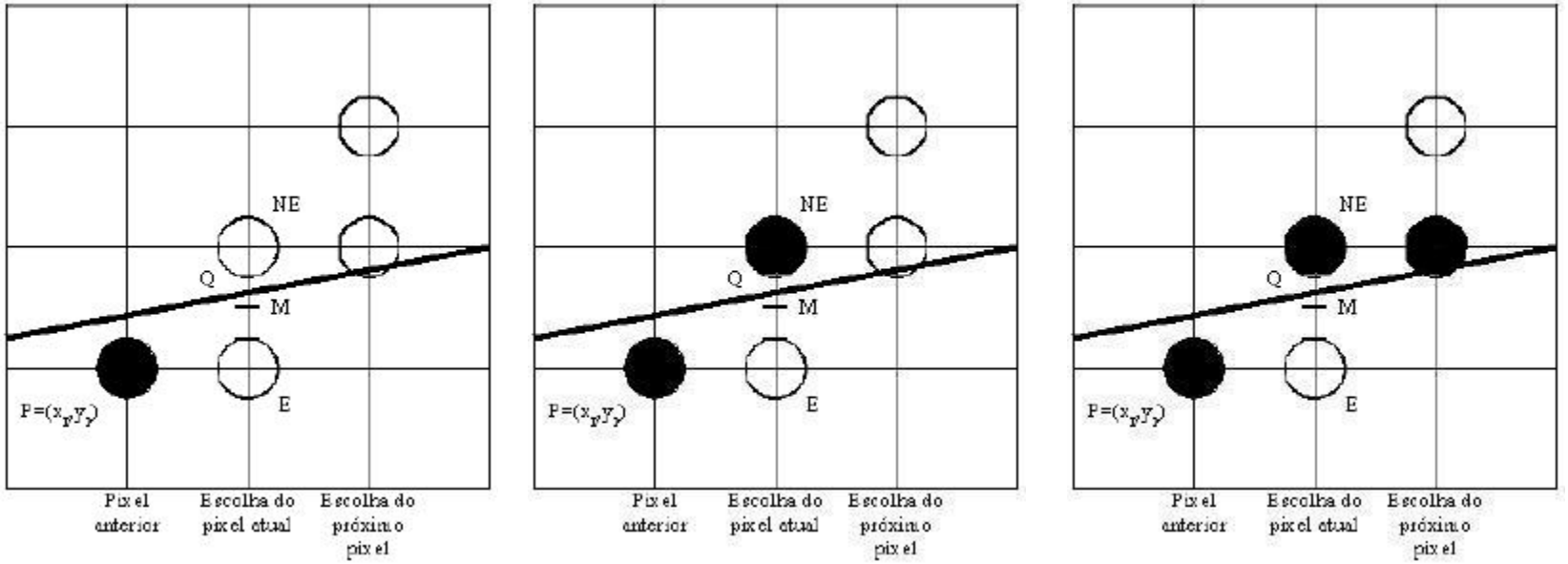


Algoritmo do Ponto Médio

- É um algoritmo eficiente já que apenas recorre a adições e subtrações inteiras e multiplicações por 2
- A partir do pixel inicial é calculado o ponto médio entre os próximos dois pixels a serem selecionados
- **Funcionamento do algoritmo**
  - Assumindo que o pixel que acabou de ser selecionado é **P**, em (xp, yp), e o próximo deve ser escolhido entre o pixel à direita (pixel E) e o pixel acima à direita (NE)
  - Seja:
    - **Q** o ponto de intersecção entre a reta e a coluna x = xp + 1 da malha, e
    - **M** o ponto intermediário entre os pixels E e NE

Algoritmo do Ponto Médio

- O que se faz é observar de que lado da reta está o ponto M
  - Se a reta estiver **acima de M**, próximo pixel é **NE**
  - Se a reta estiver **abaixo de M**, o próximo pixel é **E**
  - Dessa forma, o teste do ponto-médio permite a escolha do pixel mais próximo da reta



Algoritmo do Ponto Médio

```
dx := x2 - x1;
dy := y2 - y1;
d := (2 * dy – dx);
   { Valor inicial da variável de
   decisão d }
incE := 2 * dy;{Incremento de E }
incNE := 2 * (dy - dx);
{ Incremento de NE }
x := x1;
y := y1;
Desenha_Pixel(x, y);
```

```
while (x < x2) do
  begin
    if (d <= 0) then
    begin
      { Escolhe E – Q abaixo de M}
      d := d + incE;
      x := x + 1;
    end
  else begin
    { Escolhe NE – Q acima de M}
    d := d + incNE;
    x := x + 1;
    y := y + 1;
  end;
  Desenha_Pixel(x, y);
end;
```

Exercício

- Aplique o algoritmo para a reta:

  a) P1: (2,1) e P2: (11,3)

- Realize o teste de mesa e desenhe o traçado da reta no plano cartesiano

Material elaborado por:

Prof. Ms. Simone de Abreu

siabreu@gmail.com

Prof. Dr. Fernando Kakugawa

fernando.kakugawa@animaeducacao.com.br