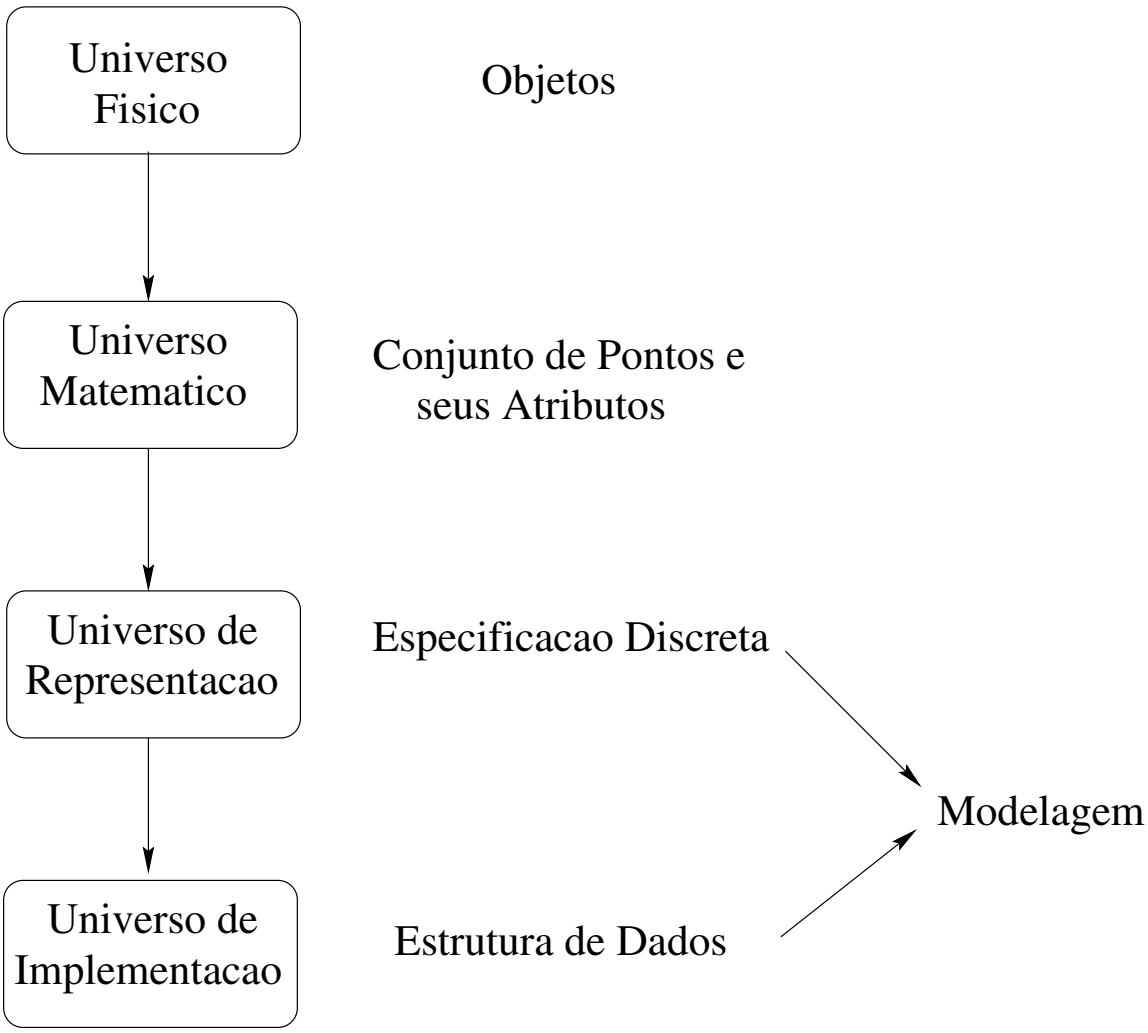
# Objetos Planares
# Introdução à Computação Gráfica

Antonio Elias Fabris

IME-USP
www.ime.usp.br/ aef
aef@ime.usp.br

# Objetos Gráficos



# Caracterização Matemática

- Objeto Gráfico = $(S, f)$
  - $S \subset R^m$
  - $f : S \subset R^m \longrightarrow R^n$
- $S$ é o suporte geométrico do objeto

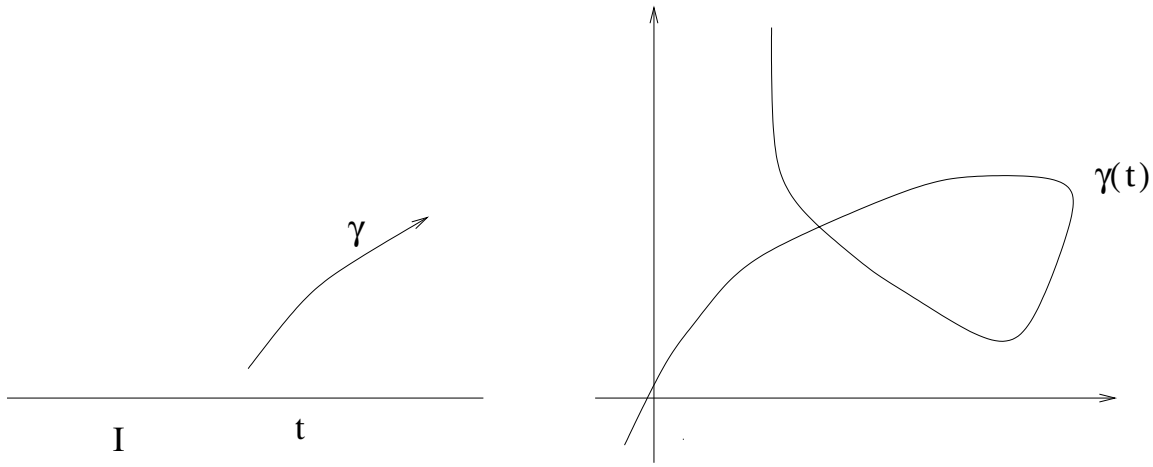
  $f$ é a função de atributos
- dimensão do objeto = $\dim S$

# Curvas

- Representação Paramétrica
- Representação Implícita
- Exemplos
- Prós e Contras das Duas Representações

# Objetos Parametrizados

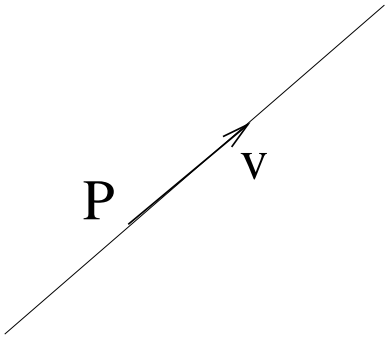
- $\gamma : I \subset R \longrightarrow R^2$ $\qquad\qquad$ $\gamma(t) = (x(t), y(t))$



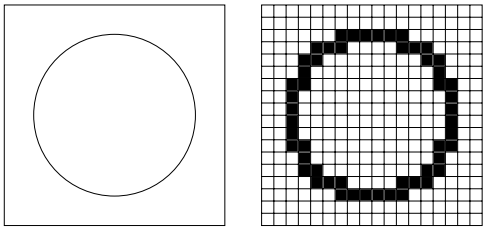

- $\gamma(I)$ é o traço da curva

# Exemplos: Repr. Paramétrica

- $\gamma(t) = p + tv \qquad t \in R$



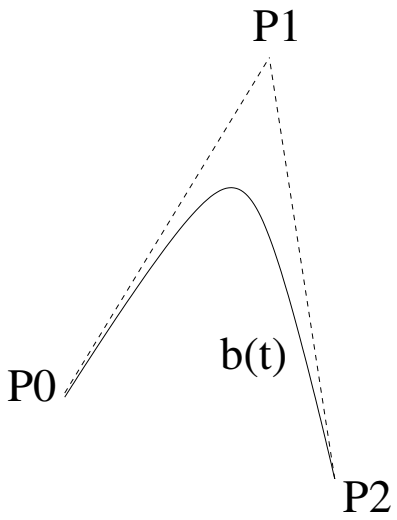

- $\gamma(t) = (\cos(t), \sin(t)) \qquad t \in [0, 2\pi]$



# Exemplos: Repr. Paramétrica

- Curvas de Bézier

$$b(t) = \sum_{i=0}^{n} \binom{n}{i} t^i (1-t)^{n-i} P_i$$




- Gráfico de Função

$$\gamma(t) = (t, f(t))$$

# Objetos Implícitos

- Objeto é especificado como conjunto das raízes de uma equação nas variáveis $x$ e $y$.

- $F : U \subset R^2 \longrightarrow R$

$$S = F^{-1}(0) = \{(x, y) \in R^2 : F(x, y) = 0\}$$

- Se $F(x, y)$ é um polinômio, a curva $F^{-1}(0)$ é dita algébrica.

# Exemplos: Repr. Implícita

- Retas

$$F(x, y) = ax + by + c \qquad ab \neq 0$$

- Círculos com centro na origem $(0, 0)$

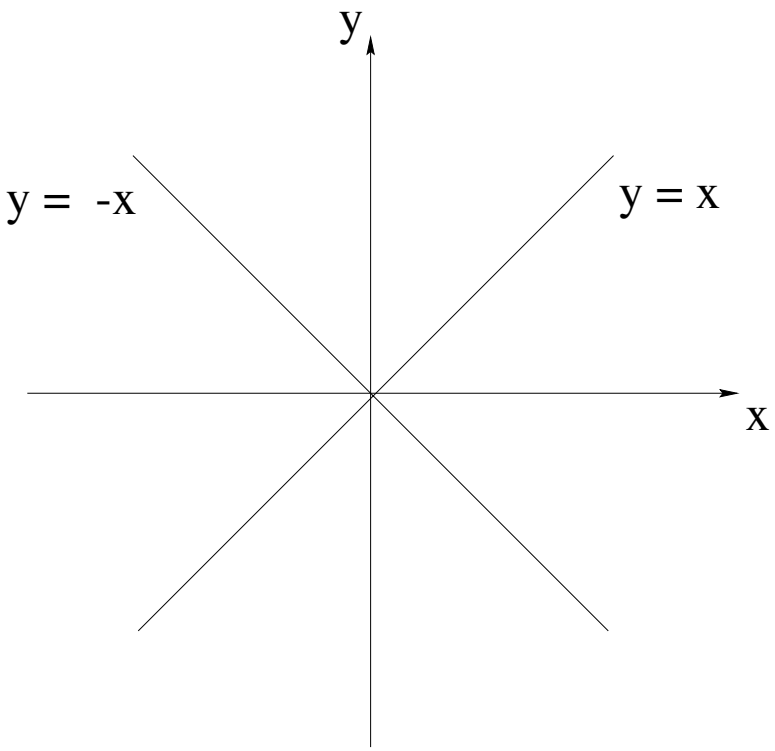
$$F(x, y) = x^2 + y^2 - r^2 \qquad r > 0$$

- Cônicas (círculo, elipse, parábola, hipérbole)

$$F(x, y) = ax^2 + by^2 + cxy + dx + ey + f \qquad abc \neq 0$$

# Valor Não- Regular

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




- $\frac{\delta F}{\delta x}(0, 0) = 0 \qquad \frac{\delta F}{\delta y}(0, 0) = 0$

- $\mathsf{grad}(F)(0, 0) = 0$

# Valor Regular

- $F^{-1}(0)$ representa uma curva topológica se e somente se $0$ é valor regular de $F$, i.e.,

$$\forall (x,y) \in F^{-1}(0)$$

$$\frac{\delta F}{\delta x}(x,y) \neq 0 \qquad \text{ou} \qquad \frac{\delta F}{\delta y}(x,y) \neq 0$$

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

$$\frac{\delta F}{\delta x}(x,y) = 2x \qquad \frac{\delta F}{\delta y}(x,y) = 2y$$

$$\frac{\delta F}{\delta x} = 0 \text{ e } \frac{\delta F}{\delta y} = 0 \Leftrightarrow (x,y) = (0,0) \notin F^{-1}(0)$$

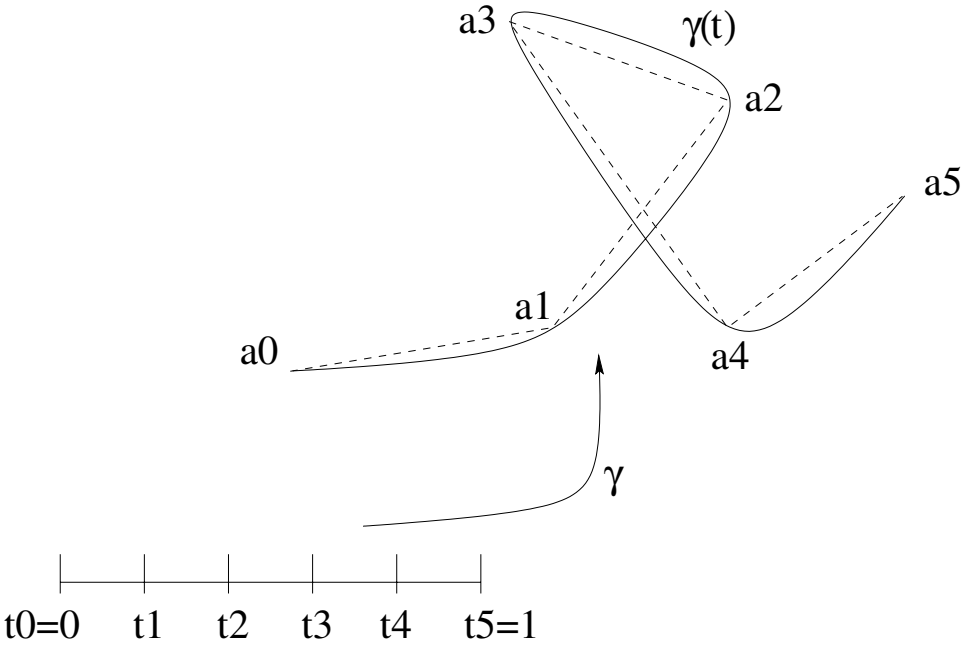
# Implícito ou Paramétrico ?

- Depende da aplicação

- Análise dos prós e contras será feita com relação a dois problemas básicos:
  - Amostragem Pontual
  - Classificação Ponto-Conjunto

# Amostragem Pontual

- $\mathcal{O}_1$ definido por $\gamma : I \subset R \rightarrow R^2$
  - Cálculo de $\gamma$ em $n + 1$ pontos do intervalo $[0, 1]$ : usualmente trivial



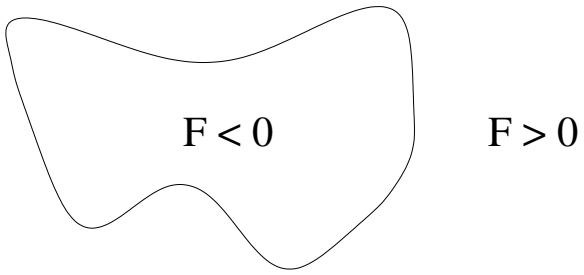

- $\mathcal{O}_2 = F^{-1}(0)$ definido por $F : R^2 \rightarrow R$
  - Admitindo-se que a representação implícita é possível, devemos calcular $n+1$ raízes da equação $F(x, y) = 0$ : quase sempre muito difícil

# Classificação Ponto-Conjunto

- $p = (p_1, p_2)$ "query point"

- $\mathcal{O}_2 = F^{-1}(0)$ definido por $F : R^2 \rightarrow R$

  - Admitindo-se que a representação implícita é possível, para determinar se $p \in \mathcal{O}_2$ basta verificar se $F(p_1, p_2) = 0$




- $\mathcal{O}_1$ definido por $\gamma : I \subset R \rightarrow R^2$

  - $p \in \mathcal{O}_1$ se o sistema de equações em $t$

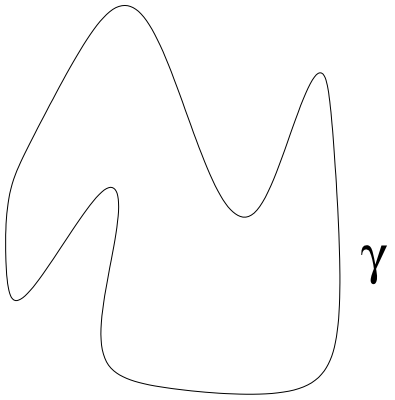
$$\begin{cases} x(t) = p_1 \\ y(t) = p_2 \end{cases}$$

    tiver soluções

# Objetos Planares Bidimensionais

- Sólido Bidimensional: região fechada e limitada do plano.

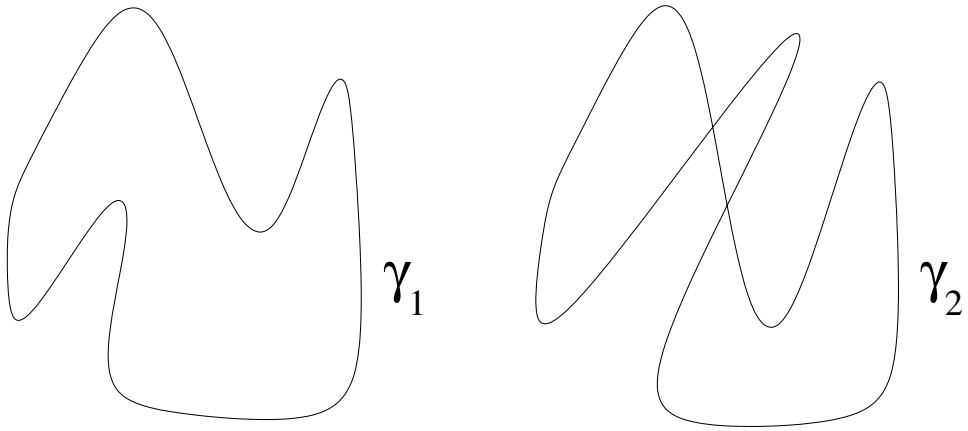
- Teorema da curva de Jordan: uma curva topológica $\gamma$ fechada divide o plano em duas regiões abertas, sendo uma limitada e a outra ilimitada.




- Representação da região limitada:
  - representar curva de fronteira
  - determinar algoritmo para resolver o problema de classificação ponto-conjunto

# Representação de regiões

- Devemos:
  - representar curva de fronteira
  - determinar algoritmo para resolver o problema de classificação ponto-conjunto




Curva Simples

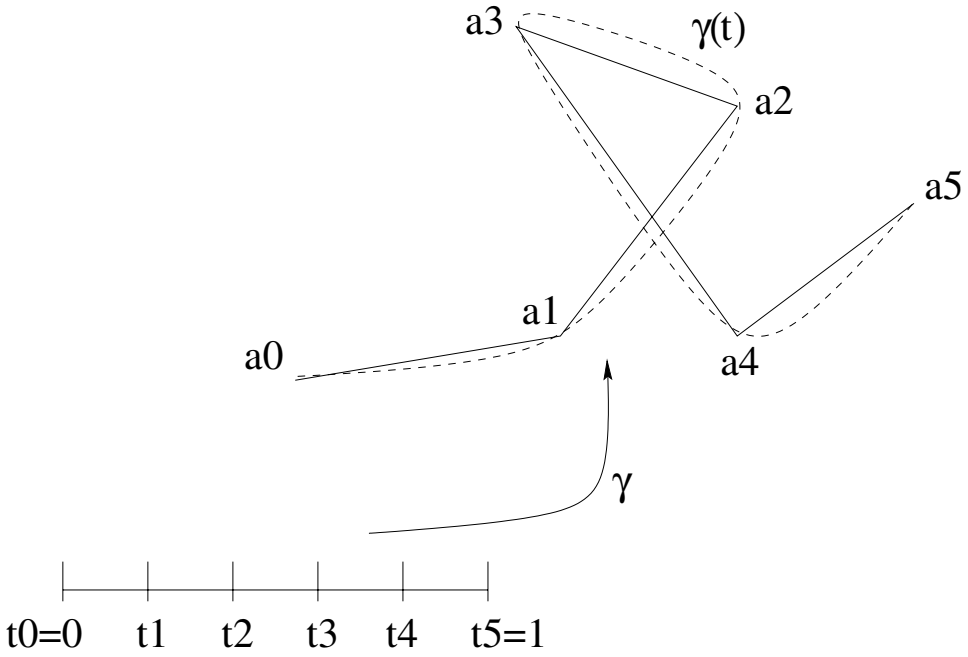
Curva Nao-Simples

# Representação de curvas

- Duas abordagens:
  - representação por aproximação poligonal
  - representação por decomposição espacial

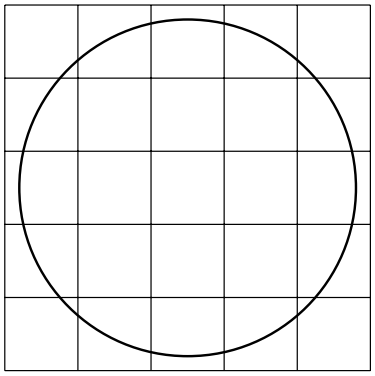
# Aproximação poligonal

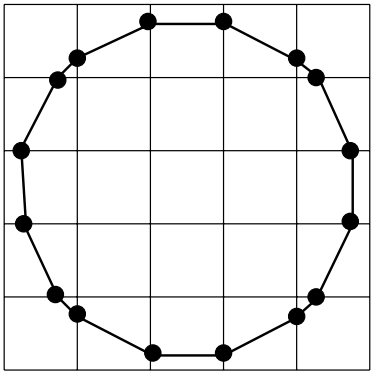
- Curvas Paramétricas

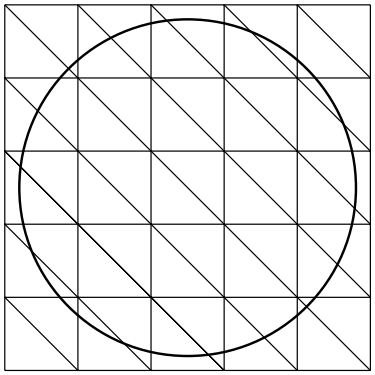


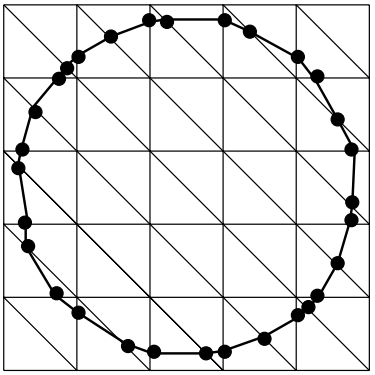
# Aproximação poligonal

- Curvas Implícitas



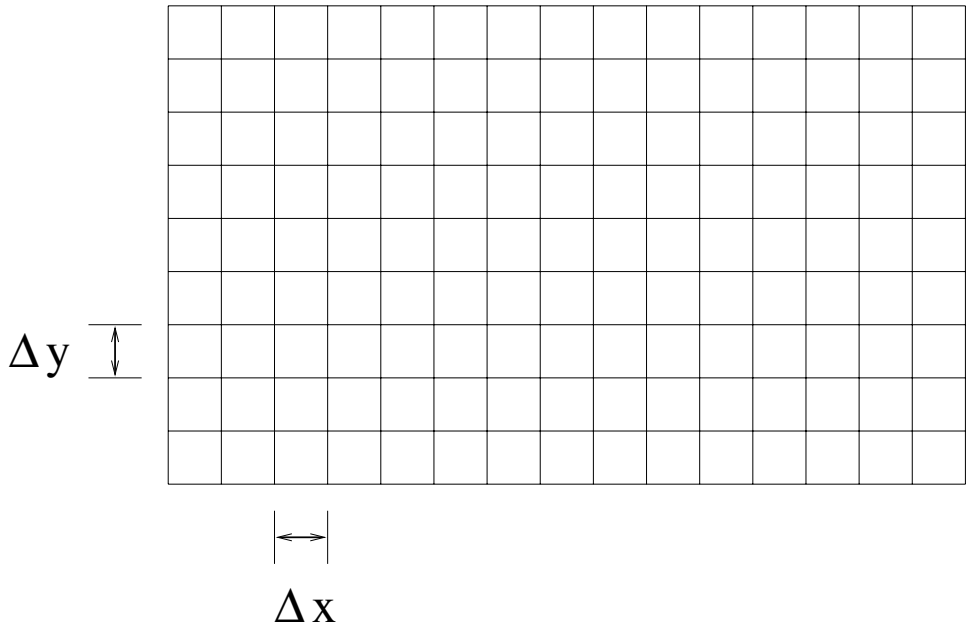






# Representação Matricial

- É um caso particular de *decomposição espacial*:
  - Discretiza o objeto gráfico como uma união de retângulos de mesma dimensão
  - definidos num reticulado uniforme do plano:



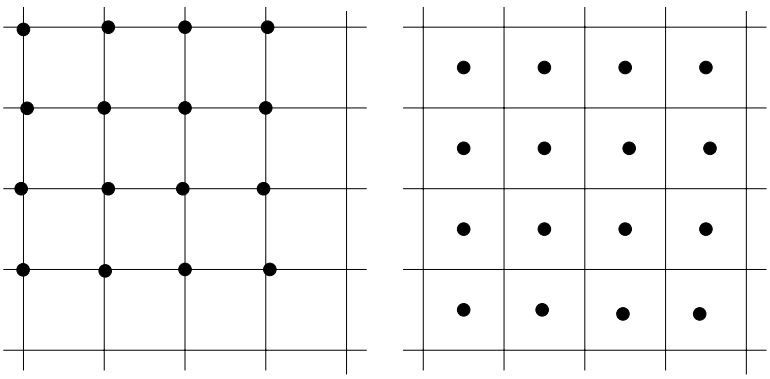

$$\Delta_{(\Delta x,\Delta y)} = \{(m\Delta x, n\Delta y) : m, n \in Z\}$$

# Representação das Células

$$\Delta_{(\Delta x, \Delta y)} = \{(m\Delta x, n\Delta y) : m, n \in Z\}$$

- Os retângulos determinados pelos pontos do reticulado $\Delta_{(\Delta x, \Delta y)}$ são chamados de *células*

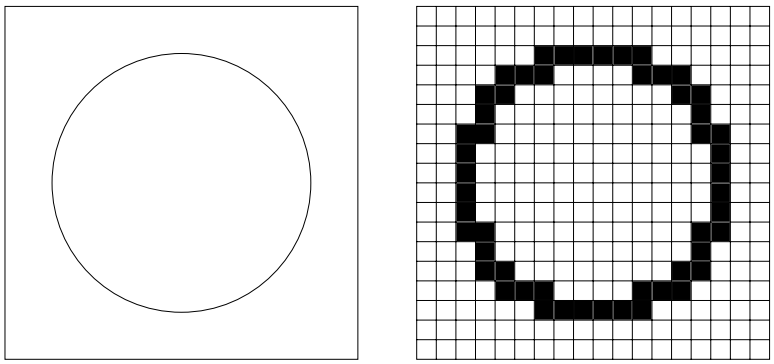
- As células podem ser representadas
  - tomando-se as coordenadas de um dos seus vértices,
  - tomando-se as coordenadas dos centróides de cada célula, etc.



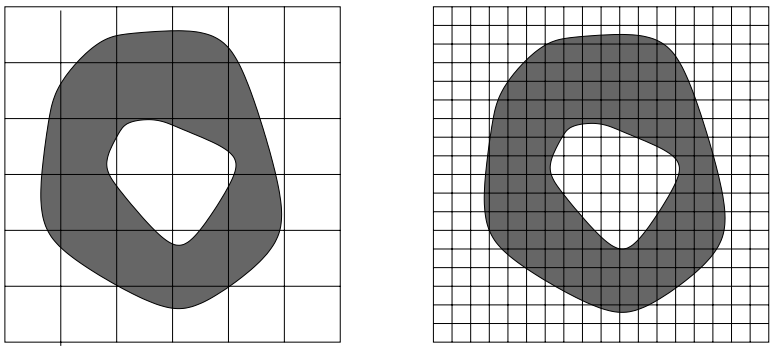
# Mais Representação Matricial ...

$$\Delta_{(\Delta x, \Delta y)} = \{(m\Delta x, n\Delta y) : m, n \in Z\}$$

- Círculo



- Preservando a topologia

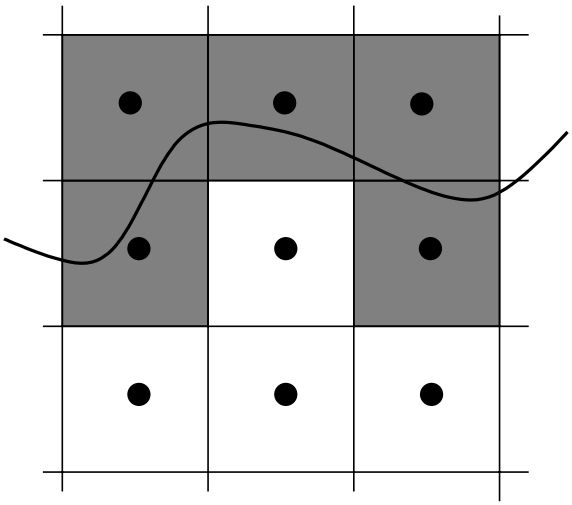


# Rasterização

- É o processo que determina uma representação matricial de um objeto $\mathcal{O} \subset R^2$ num reticulado do plano

- Noutras palavras:
  - dado um reticulado $\Delta_{(\Delta x, \Delta y)}$ do plano com $m \times n$ células,
  - o processo de rasterização consiste em obter uma enumeração $C_1, C_2, \cdots, C_k$ de células do reticulado que constituem uma representação do objeto gráfico $\mathcal{O} \subset R^2$

# Escolha das células de rasterização



- Um critério:
  - $C_i$ é célula da representação do objeto $\mathcal{O}$ $\Leftrightarrow \quad C_i \wedge \mathcal{O} \neq \emptyset$
- Critério mais simples:
  - $C_i$ é célula da representação do objeto $\mathcal{O}$ $\Leftrightarrow$ centróide $P_i$ é tal que $P_i \in \mathcal{O}$
  - Faz sentido ?

# Tipos de rasterização

- Rasterização Incremental: células que representam o objeto são obtidas por um procedimento iterativo
  - Intrínseca: percorre-se o objeto
  - Espacial: percorre-se o reticulado
- Rasterização por Subdivisão:
  - Intrínseca: o suporte geométrico do objeto é subdividido até que cada subconjunto obtido esteja contido numa célula do reticulado.
  - Espacial: uma região do plano que contém o objeto é subdividida até que
    - não existem pontos do objeto na sub-região
    - sub-região está contida no objeto
    - sub-região contida numa célula do reticulado

# Rasterização Incremental Intrínseca

- Um exemplo - algoritmo ingênuo para retas: tome passos incrementais unitários em $x$; aumente $Coef\_Ang$ em $y$; arredonde $y$ para o pixel mais próximo (usar simetria se $Coef\_Ang > 1$)

  Segmento_Reta (int $x_0$, int $y_0$, int $x_1$, int $y_1$)

  float $y = y_0$

  float Coef_Ang = ($y_1$-$y_0$)/(float)($x_1$-$x_0$)

  int $x$

  for ($x = x_0$; $x <= x_1$; $x++$)

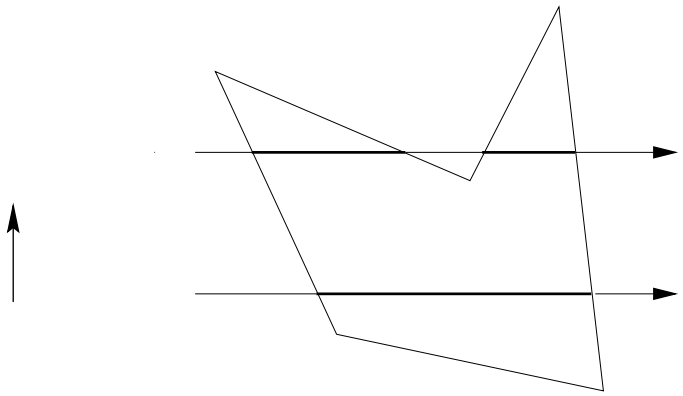
  DesenhaPixel ($x$,Round($y$))

  $y$ += Coef_Ang;

- Função de arredondamento e adição em ponto flutuante são caras.

# Rasterização Incremental Espacial

- Exemplo: preenchimento de regiões

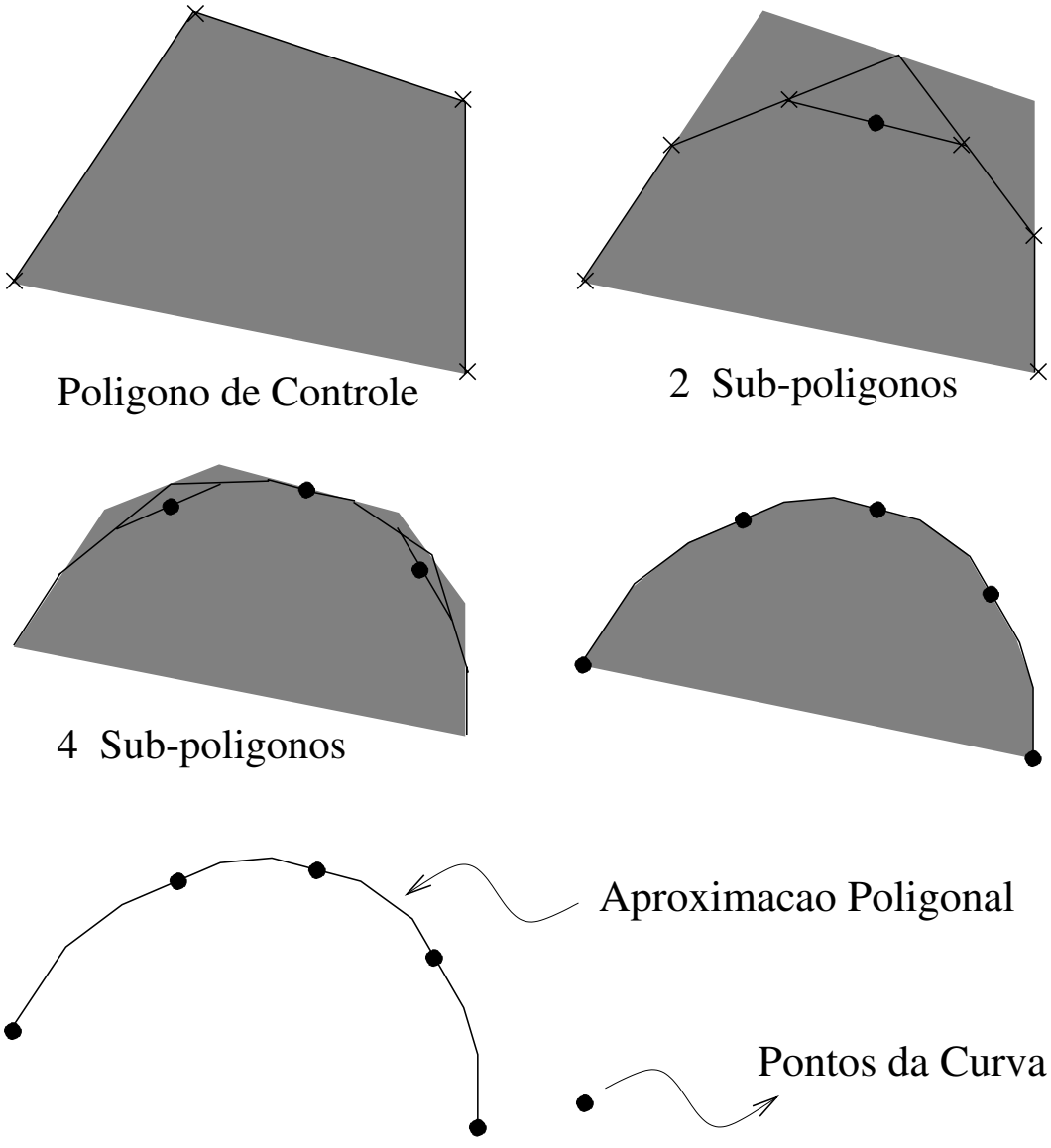


- Para cada linha do reticulado
  - Determine todas as intersecções linha/polígono
  - Ordene da esquerda para a direita
  - Preencha segmentos interiores entre interseções

Régra da Paridade $\left\{ \begin{array}{l} \text{ímpares são interiores} \\ \text{pares são exteriores} \end{array} \right.$

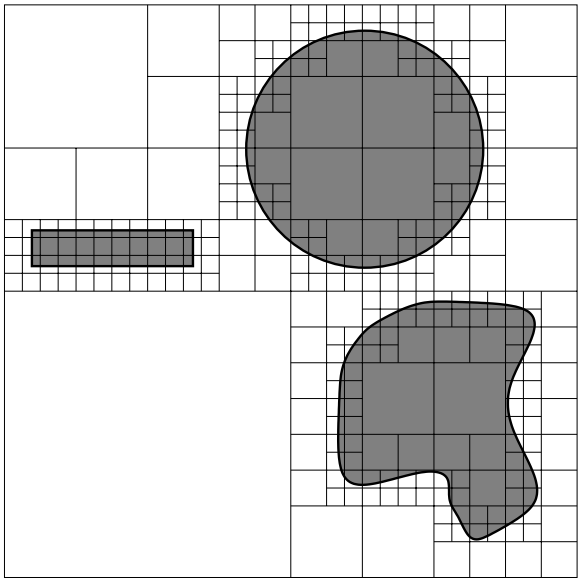
# Subdivisão Intrínseca

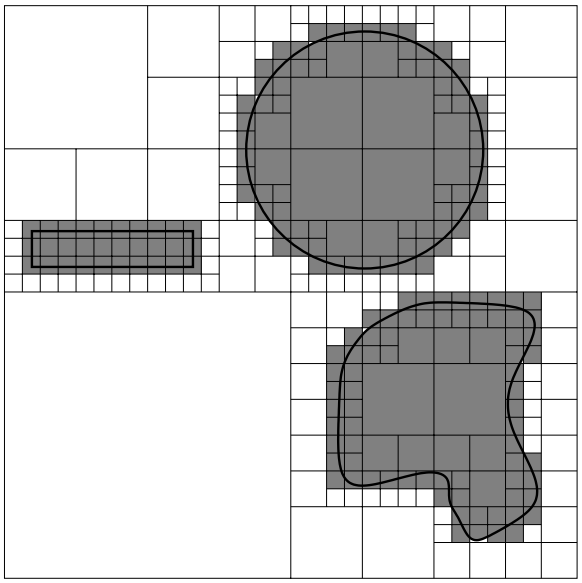
- Exemplo: curvas de Bézier



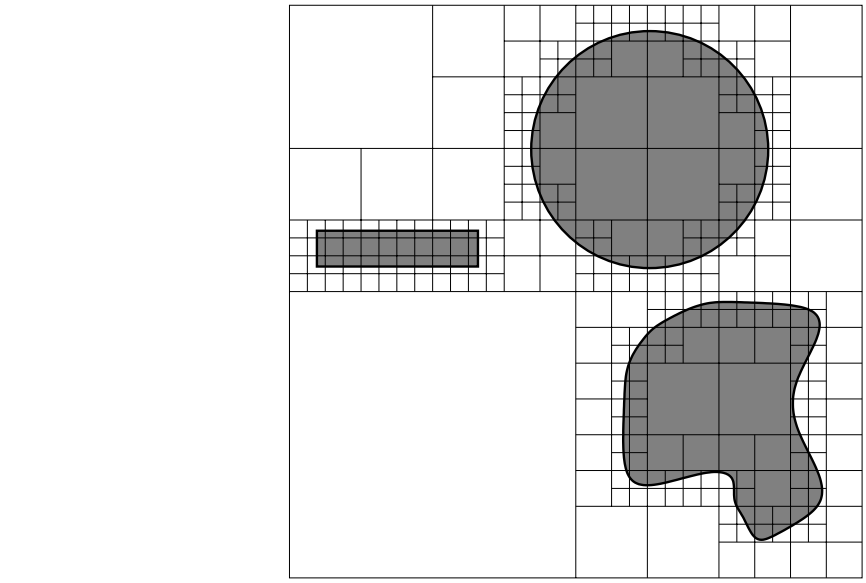
# Subdivisão Espacial

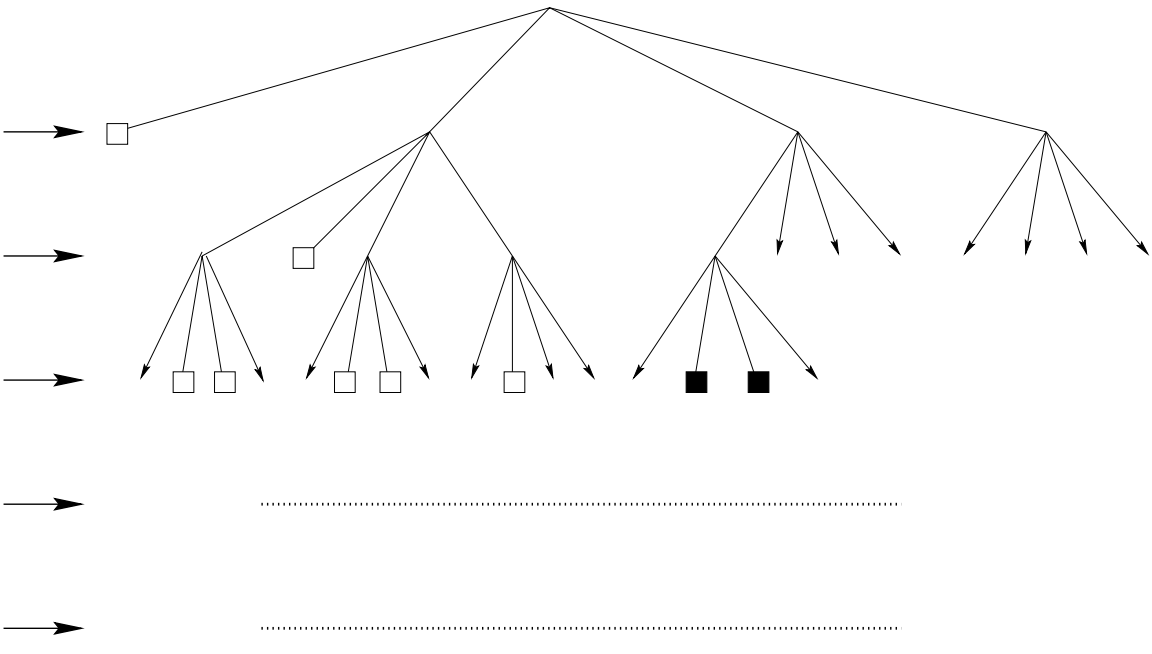
- Exemplo: preenchimento de regiões





# Rasterização de Regiões





# Dispositivos de Saída Gráfica

- **Dispositivos Vetoriais**: desenham polígonos e segmentos de reta diretamente
  - Eram os únicos até os anos 70
  - Exemplos atuais:
    - Plotters
    - Sistemas de Projeção a Laser
- **Dispositivos Matriciais**: representam uma imagem como um reticulado regular de amostras
  - CRTs, LCDs, Plasma, Impressoras, etc.
  - Cada amostra é chamada pixel
  - Exige algoritmos de rasterização: para determinação dos pixels que representam as primitivas geométricas

# Coordenadas e Dispositivos

- Inicie com uma cena 2D
  - Pode ser a projeção de uma cena 3D, que efetivamente estamos querendo visualizar
- Cena 2D está descrita num plano de modelagem.
  - que pode ser infinito
- O dispositivo de saída tem uma retângulo finito visível.
- O que devemos fazer ?

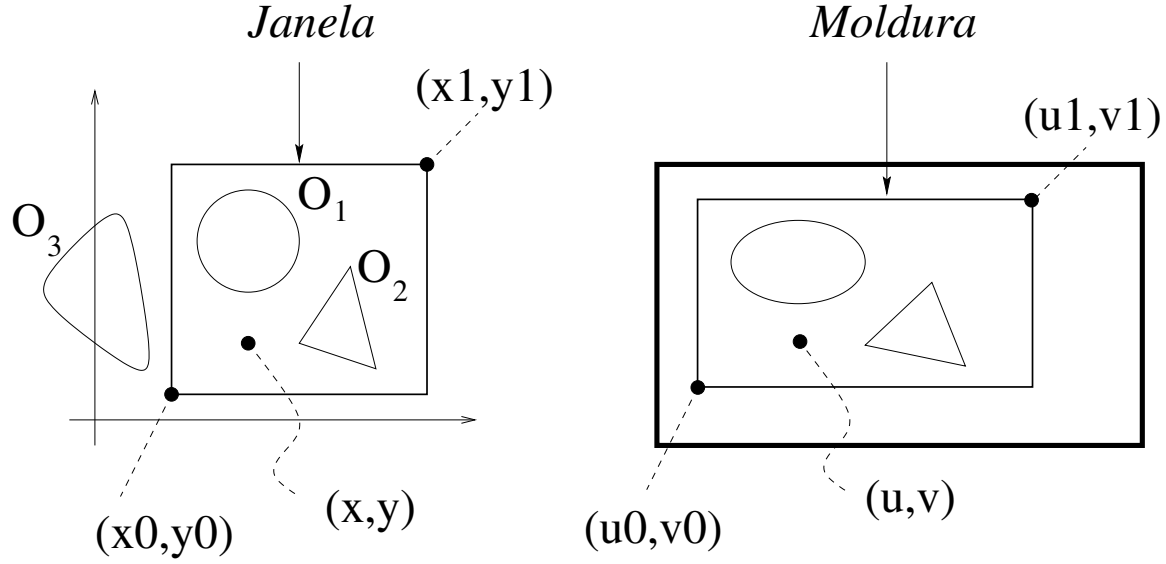
# Coordenadas e Dispositivos

- Resposta: Mapear a região de interêsse da cena para exibição nas coordenadas do dispositivo

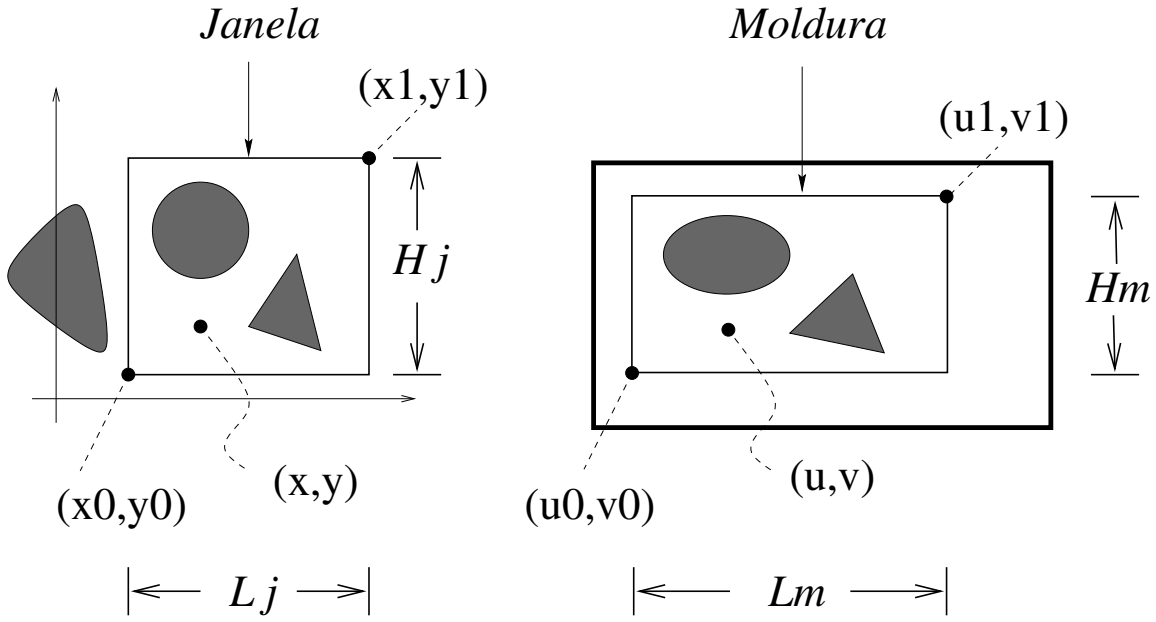
  – Janela (Window): região retangular de interêsse na cena

  – Moldura (Viewport): região retangular no dispositivo

    Usualmente ambos retângulos são alinhados com os eixos coordenados



# Coordenadas e Dispositivos



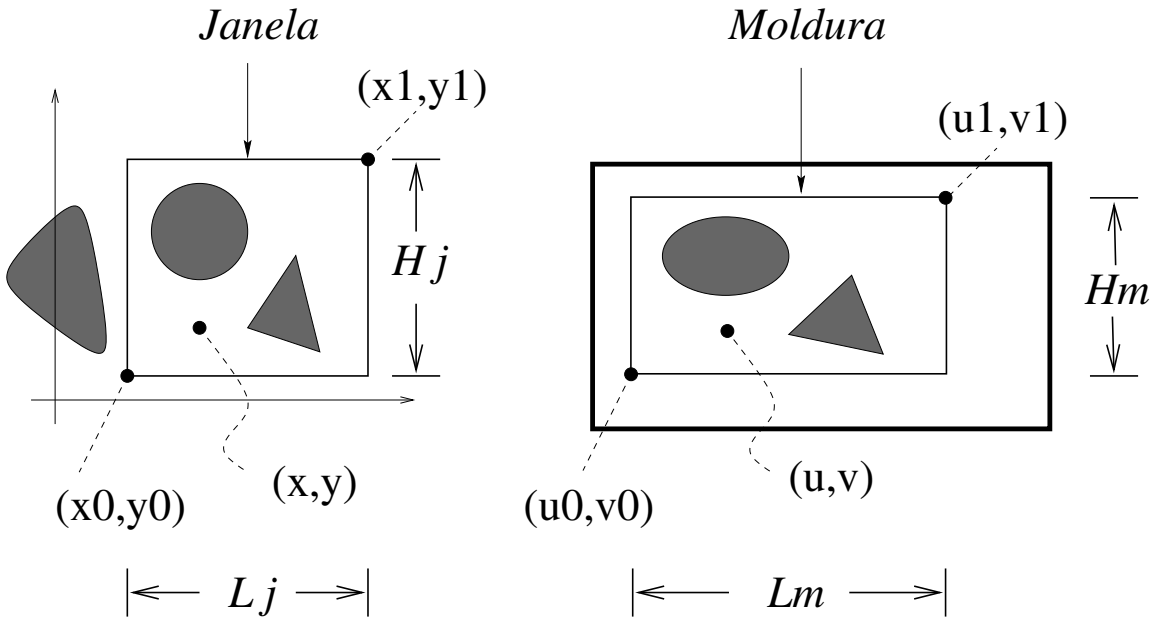

- $\Delta x = x - x_0 \qquad \Delta y = y - y_0$
  $\Delta u = u - u_0 \qquad \Delta v = v - v_0$

- Mapeie proporcionalmente cada coordenada:

$$\frac{\Delta x}{L_j} = \frac{\Delta u}{L_m} \qquad \frac{\Delta y}{H_j} = \frac{\Delta v}{H_m}$$

$$\frac{x - x_0}{L_j} = \frac{u - u_0}{L_m} \Rightarrow u = \frac{L_m}{L_j}(x - x_0) + u_0$$

# Coordenadas e Dispositivos




$$\frac{x - x_0}{L_j} = \frac{u - u_0}{L_m} \Rightarrow u = \frac{L_m}{L_j}(x - x_0) + u_0$$

$$\frac{y - y_0}{H_j} = \frac{v - v_0}{H_m} \Rightarrow v = \frac{H_m}{H_j}(y - y_0) + v_0$$

- Se $H_j/L_j \neq H_m/L_m$ a imagem será distorcida
  - Razões de semelhança (*aspect ratios)*

# Coordenadas e Dispositivos

- Onde devemos especificar a moldura ?

- Pode ser nas coordenadas do dispositivo . . .

- MAS, suponha que rodaremos o programa em diversas plataformas de hardware e/ou em distintos dispositivos de saída

  – origem no canto inferior esquerdo, no canto superior esquerdo

  – workstations: $1280 \times 1024$ frame buffers

  – página PostScript: $2550 \times 3300$ pixels em $300$dpi

  – etc, etc.

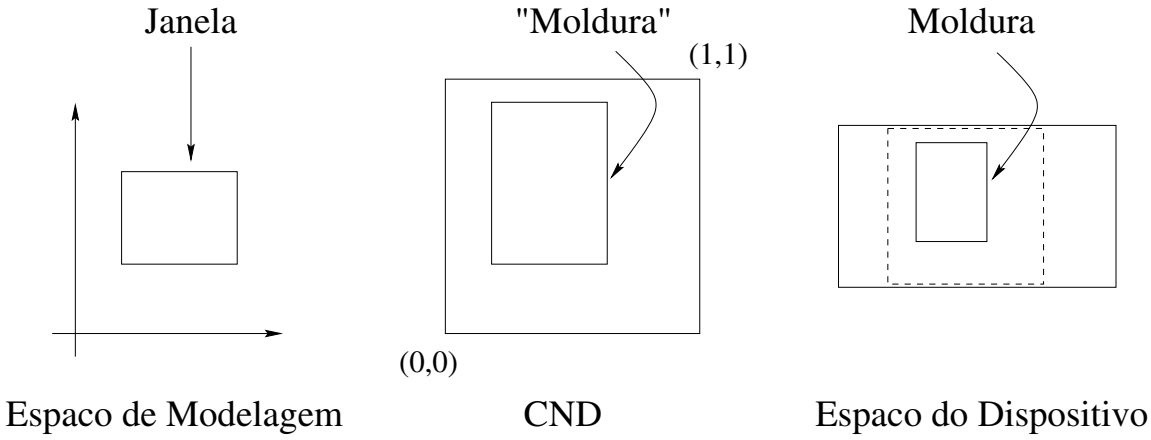
# Coordenadas e Dispositivos

- Se mapearmos diretamente as CMO para as CD, teremos de reescrever tal transformação para cada particular dispositivo de saída

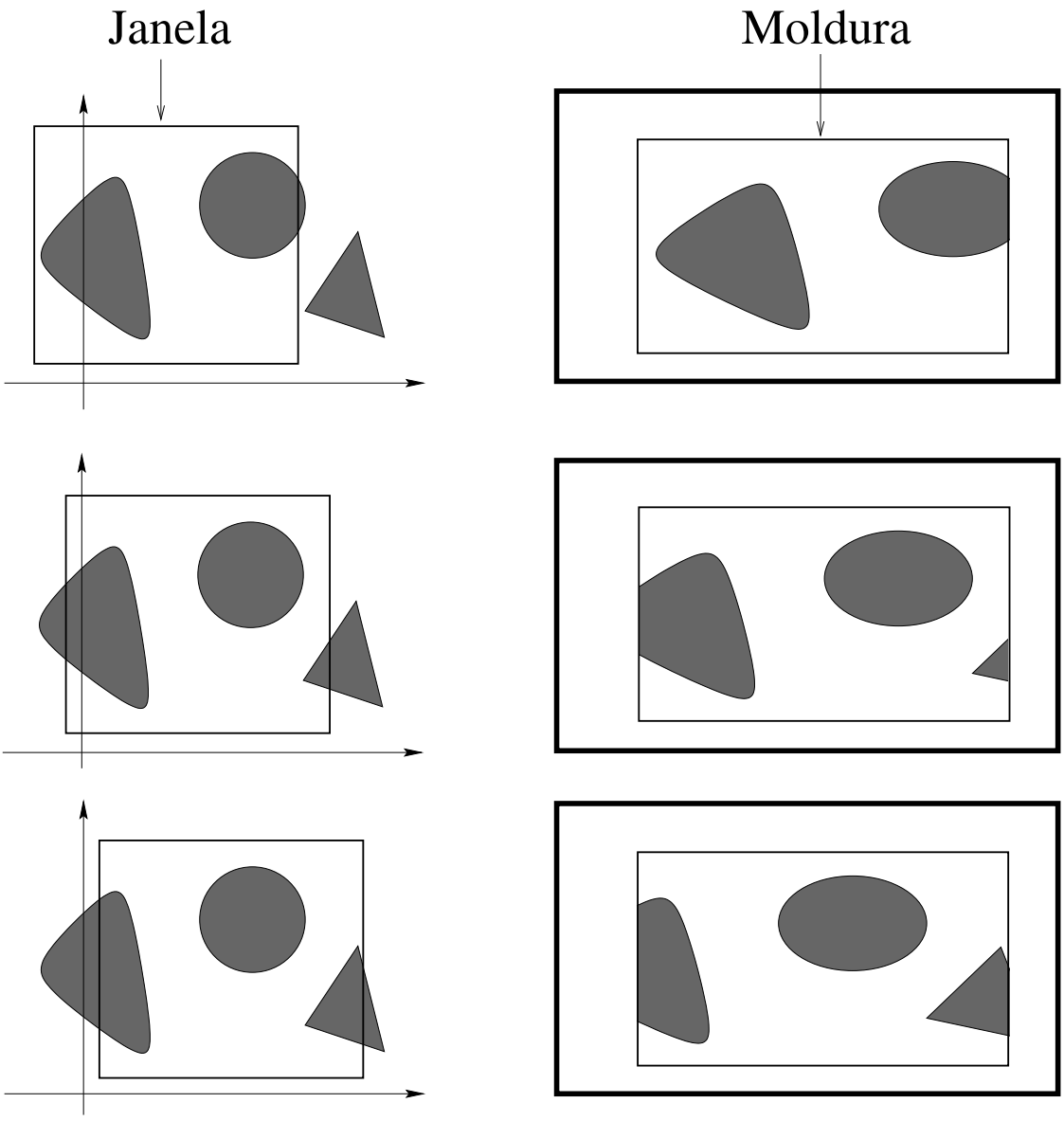
- Ao invéz, usamos Coordenadas Normalizadas do Dispositivo (**CND**) como um sistema intermediario
  - Então basta escala simples

- Considere regiões quadradas do dispositivo:

janelas $\longrightarrow$ “molduras” $\longrightarrow$ molduras



# Recorte Planar



# Porque recortar ?

- Eficiência computacional: realiza-se
  - rasterização
  - transformações de coordenadas

  sómente nos objetos que serão exibidos

- Crucial em aplicações onde os modelos são compostos por um número grande de primitivas

- Exemplos
  - Sistemas Gráficos de Visualização 3D
  - Animação 2D e 3D
  - Visão computacional e robótica, etc.

- Heurística: faça recorte sempre . . .

# Abordagens básicas para recorte

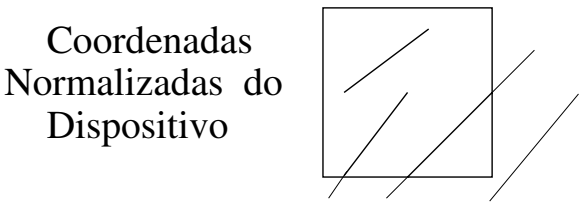
- Antes da rasterização: recorte “contínuo”
  - feito analíticamente em algum sistema de coordenadas que antecede o sistema de coordenadas do dispositivo
  - dificuldade: cálculo de muitas interseções nas cenas complexas
- Depois: recorte “discreto”
  - para cada pixel da cena total rasterizada

    testar pertinência à moldura
  - dificuldade: pode fazer transformações e rasterização em objetos que não serão visualizados

# Recorte: força bruta

- Recorte analítico de segmentos de reta em relação a regiões retangulares horizontais

- testar predicado interior/exterior para extremidades com relação aos semi-planos
  - ambos interiores: aceitação trivial
  - um interior: determine interseção e recorte
  - ambos exteriores: ou recorte ou rejeite




- Dificuldade: muitos cálculos de intersecções envolvendo divisão em ponto flutuante.

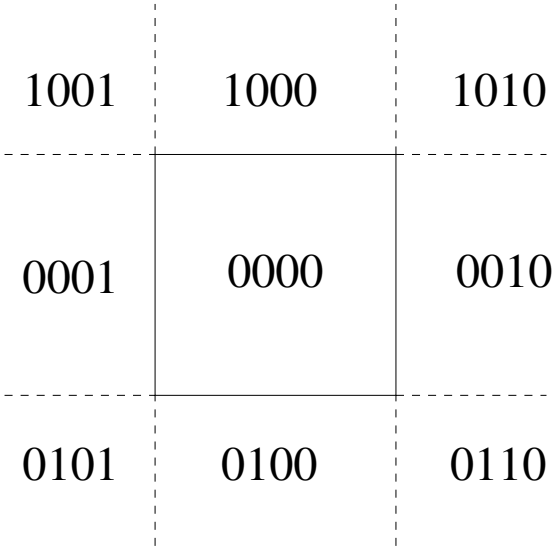
# Recorte: Cohen-Sutherland

bit 1 : y > ymax

bit 2 : y < ymin

bit 3 : x > xmax

bit 4 : x < xmin




- outcode($P_1$) e outcode($P_2$) nulos : aceitação trivial
  outcode($P_1$) &&outcode($P_2$) $\neq 0$ : rejeição trivial
  caso contrário : subdivide (como ?)

- Teste dos outcodes feito com operações booleanas
  Calcula intersecções quando necessário
  Pode ser extendido para 3D

- Muito bom quando o número de segmentos visíveis é pequeno proporcionalmente ao número total de segmentos