Modelação de Formas Geométricas

Computação Gráfica

Inverno 2011/2012

Parcialmente adaptado de Hanspeter Pfister, Harvard / MERL

http://hof.povray.org/images/Villarceau_Circles-CSG.jpg

Sumário

- Formas 2D
  - Representação de rectas e curvas
  - Polígonos e Triângulos
    - Coordenadas baricêntricas
    - Rasterização de triângulos
  - Partição do espaço (plano): *Quadtrees* e BSP *trees*

- Formas 3D
  - Poliedros
  - *Meshes*
  - Partição do espaço: *Octrees* (e BSP *trees*)
  - Represenção do espaço: *Voxels*
  - Modelação de Sólidos: CSG

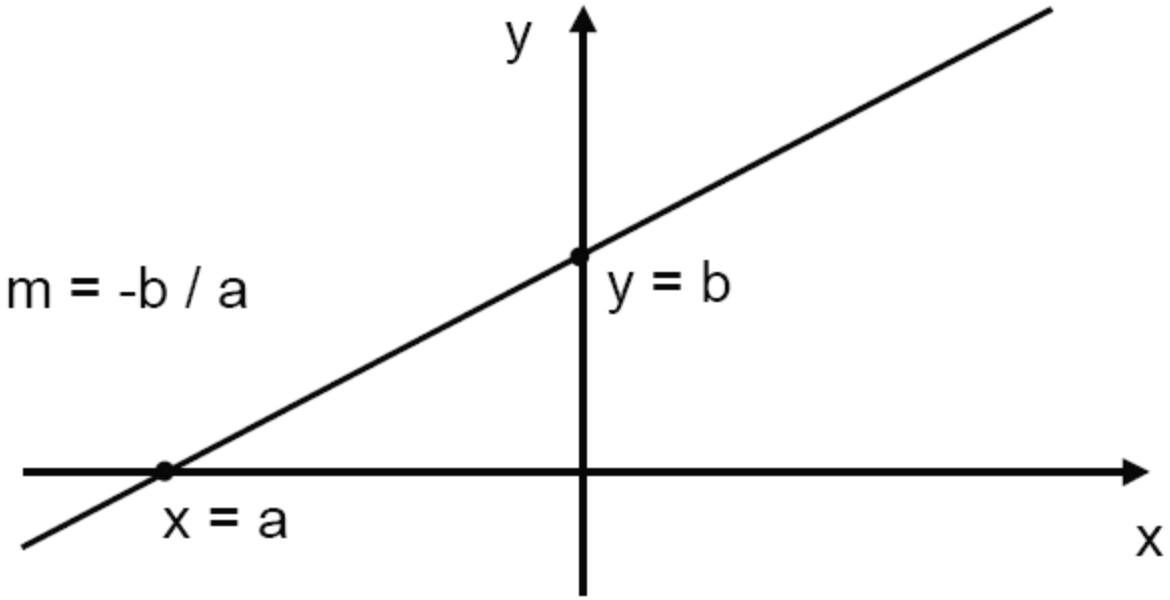
- Normais: ponte entre forma e luz

Representação de Rectas e Curvas

- **Forma explícita**

$$y = f(x)$$

$$y = mx + b$$

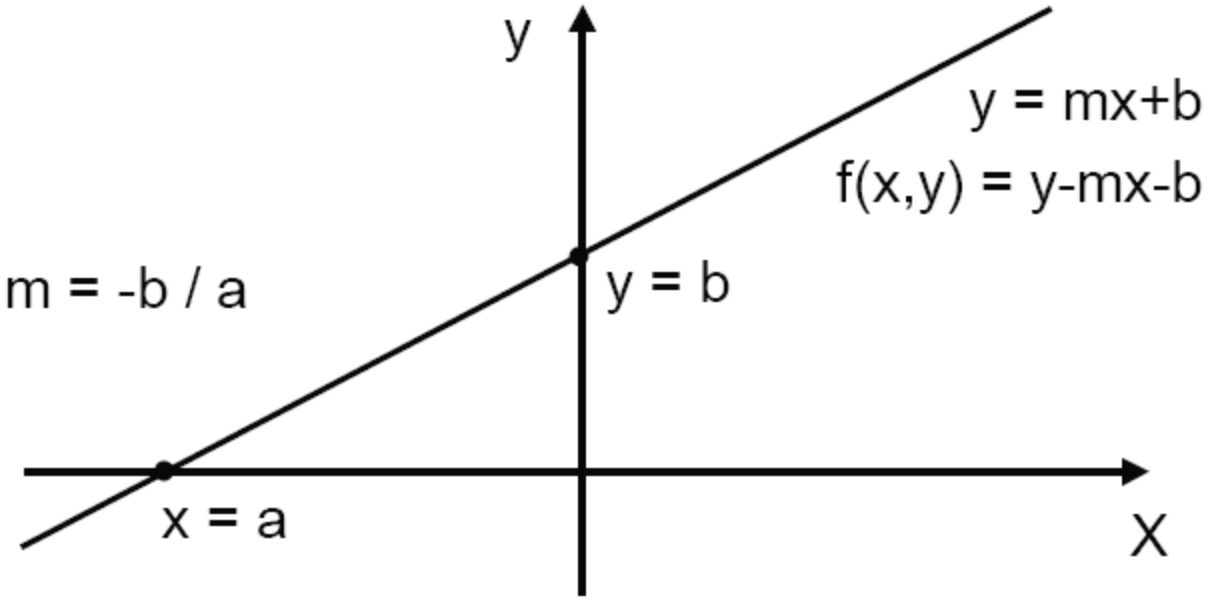


Representação de Rectas e Curvas

- **Forma implícita**

$$f(x,y) = 0$$

$$y - mx - b = 0$$



Problema: linhas verticais

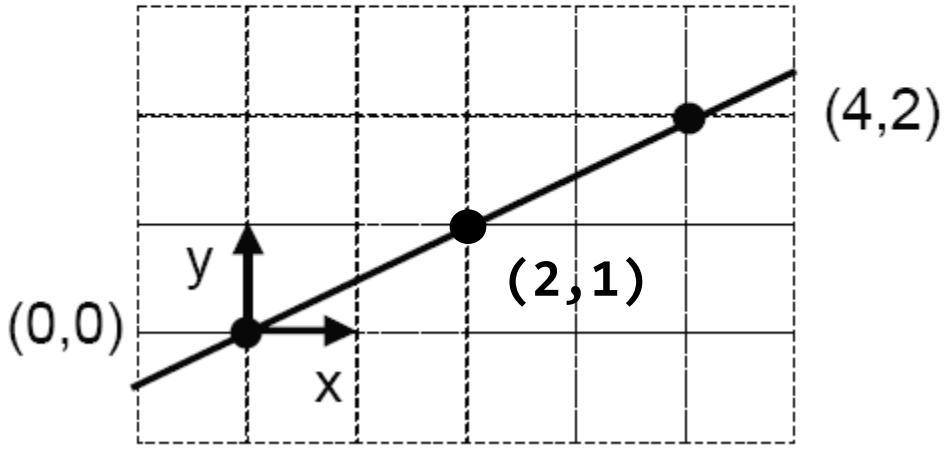
Representação de Rectas e Curvas

- **Forma implícita**

$$f(x,y)=0$$

$$(y_0-y_1)x+(x_1-x_0)y+x_0y_1-x_1y_0=0$$

$$-2x+4y+0=0$$

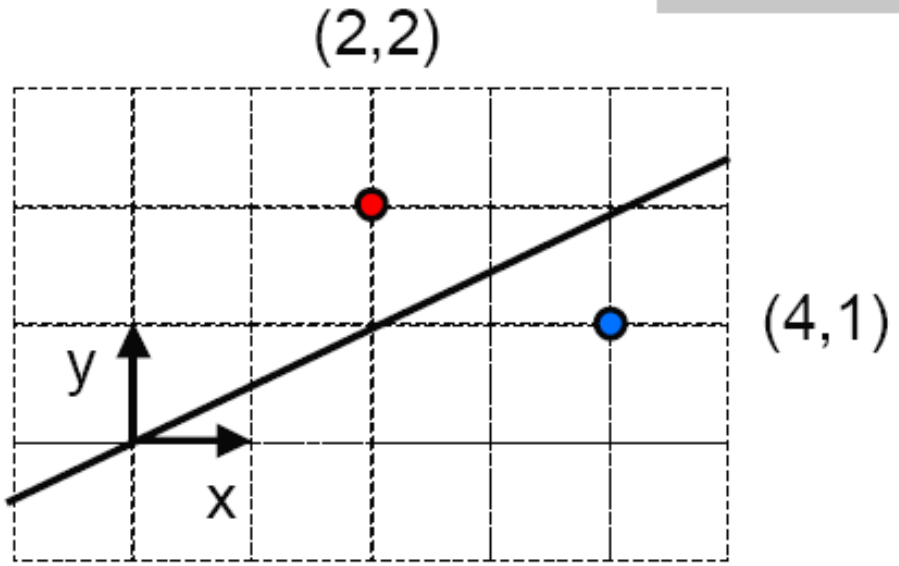


$$Ax+By+C=0$$

Representação de Rectas e Curvas

- **Vantagem da forma implícita:**
  - é possível determinar se um ponto está **acima**/**abaixo** da linha…

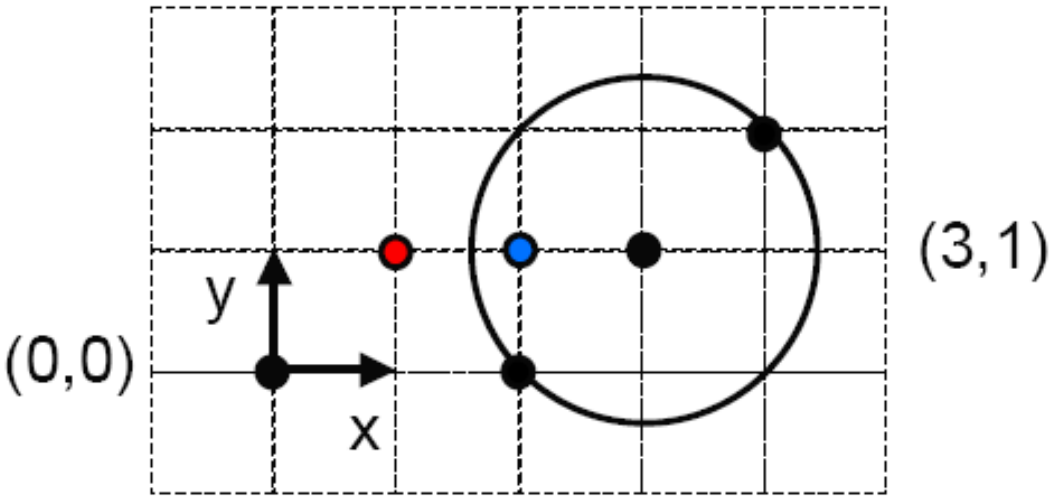
$$f(2,2) = +4 \quad (+ = \text{above})$$
$$f(4,1) = -4 \quad (- = \text{below})$$



Representação de Rectas e Curvas

- ...ou então, verificar se está **fora**/**dentro** de um círculo

$$f(x,y) = (x - x_c)^2 + (y - y_c)^2 - r^2$$



$$(x_c, y_c) = (3,1)$$

$$r = \sqrt{2}$$

$$f(2,0) = 1 + 1 - 2 = 0$$

$$f(4,2) = 1 + 1 - 2 = 0$$

$$f(1,1) = 4 + 0 - 2 = 2$$

$$f(2,1) = 1 + 0 - 2 = -1$$

Representação de Rectas e Curvas

- Forma paramétrica
  - Uma curva paramétrica é governada por um único parâmetro, **t**
  - Uma linha recta é um caso particular de uma curva
  - Qual é a forma paramétrica de um círculo?

$$x = g(t)$$
$$y = h(t)$$

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} g(t) \\ h(t) \end{bmatrix}$$

- Forma paramétrica de um círculo
  - O parâmetro t representa o o ângulo α com o eixo dos xx

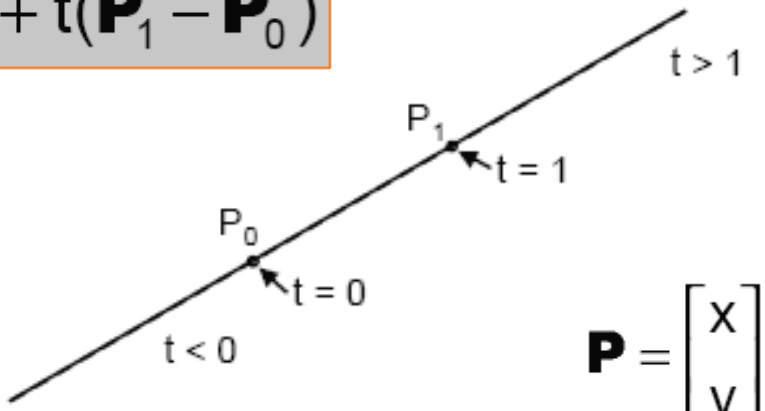
$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \cos(\alpha) \\ \sin(\alpha) \end{bmatrix}$$

Representação de Rectas e Curvas

- Forma paramétrica da recta

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} x_0 + t(x_1 - x_0) \\ y_0 + t(y_1 - y_0) \end{bmatrix}$$

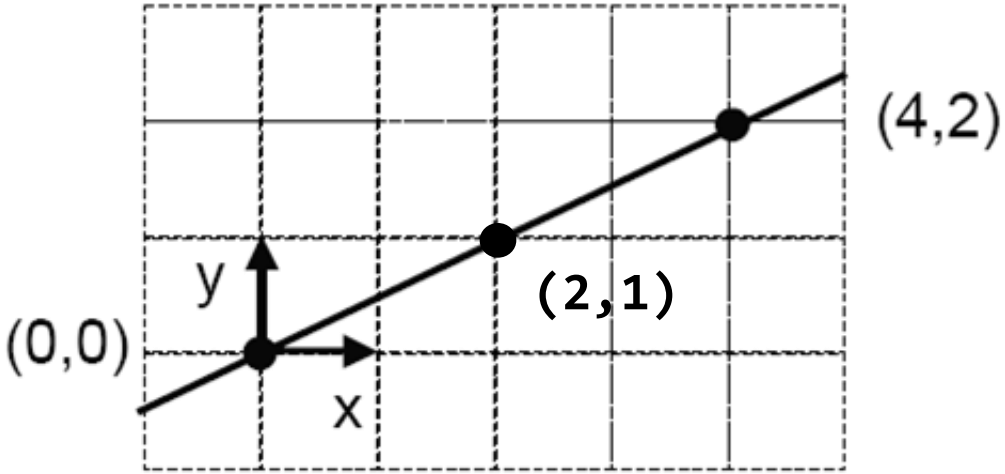
$$\mathbf{P}(t) = \mathbf{P}_0 + t(\mathbf{P}_1 - \mathbf{P}_0)$$



$$\mathbf{P} = \begin{bmatrix} x \\ y \end{bmatrix}$$

Representação de Rectas e Curvas

- Qual é a equação paramétrica desta recta?



$$\mathbf{P}(t) = \mathbf{P}_0 + t(\mathbf{P}_1 - \mathbf{P}_0)$$

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} + t \begin{bmatrix} 4 \\ 2 \end{bmatrix}$$

Pontos **(0,0)** e **(4,2)**

Ou

$$\mathbf{P}(t) = \mathbf{P}_0 + t(\mathbf{P}_1 - \mathbf{P}_0)$$

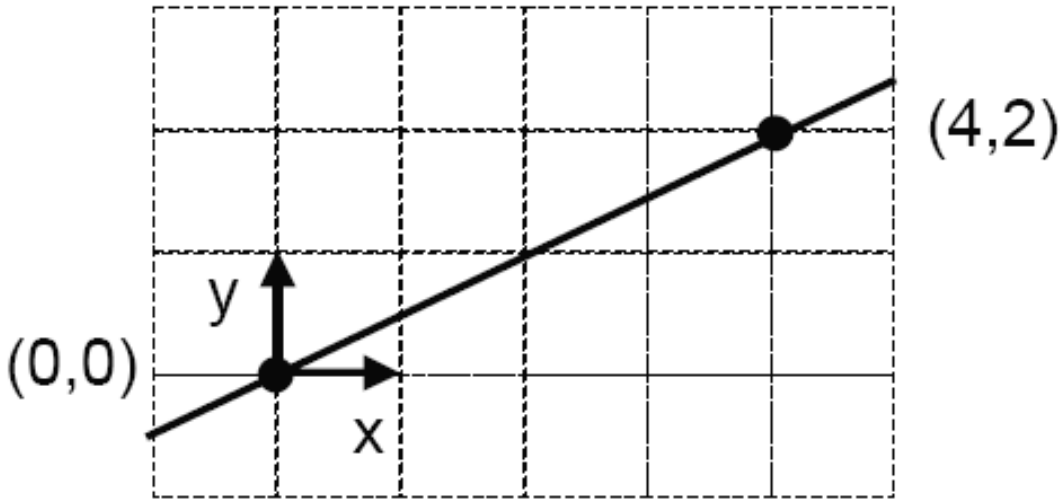
$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 2 \\ 1 \end{bmatrix} + t \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

Pontos **(2,1)** e **(4,2)**

Representação de Rectas e Curvas

- Quais são os pontos da recta para os seguintes valores de **t** ?

$$\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} + t \begin{bmatrix} 4 \\ 2 \end{bmatrix}$$



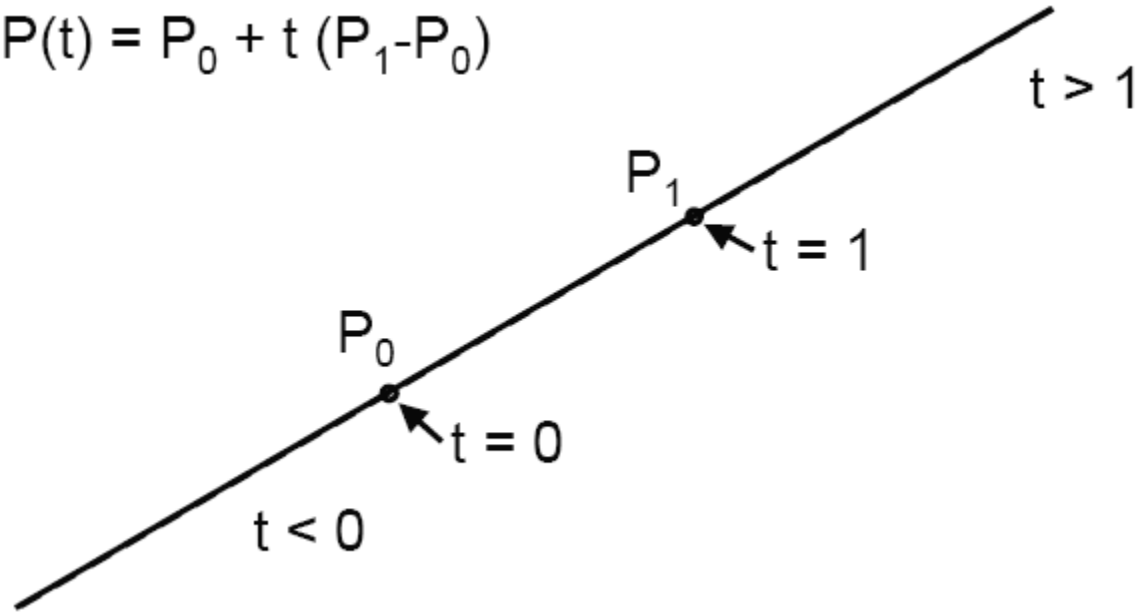
$$\mathbf{P}(t=0) = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\mathbf{P}(t=1) = \begin{bmatrix} 4 \\ 2 \end{bmatrix}$$

$$\mathbf{P}(t=0.5) = \begin{bmatrix} 2 \\ 1 \end{bmatrix}$$

$$\mathbf{P}(t=-0.25) = \begin{bmatrix} -1 \\ -½ \end{bmatrix}$$

Representação de Rectas e Curvas



• Recta	$-\infty < t < \infty$
• Segmento de recta	$0 \leq t \leq 1$
• Raio (*Ray*)	$0 \leq t < \infty$

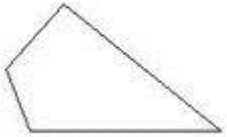
Comparação entre representações

- Forma explícita $y=f(x)$
  - É adequada para gráficos de funções $y=f(x)$
  - Não permite representar linhas verticais (e. g., $x=0$)
- Forma implícita $f(x,y)=0$
  - Fácil testar se um ponto está sobre a curva
  - Permite testar acima/abaixo ou fora/dentro (>0, <0)
- Forma paramétrica
  - Permite desenhar todo o tipo de curvas
  - É fácil gerar pontos sobre a curva

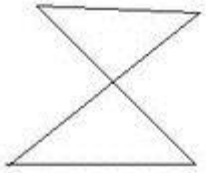
- **Todas estas representações se estendem facilmente para 3D**

Polígonos

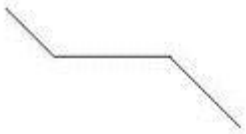
- Definição de polígono
  - *Caminho planar fechado composto por uma sequência finita de segmentos de recta*, ou *Linha poligonal fechada (simples ou não-simples)*



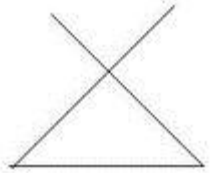
Linha poligonal fechada simples



Linha poligonal fechada não-simples



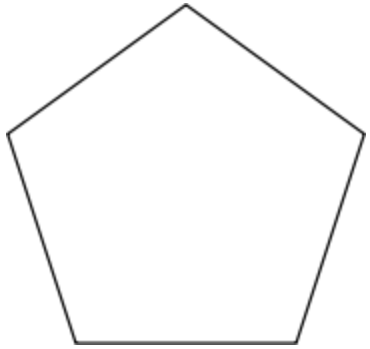
Linha poligonal aberta simples

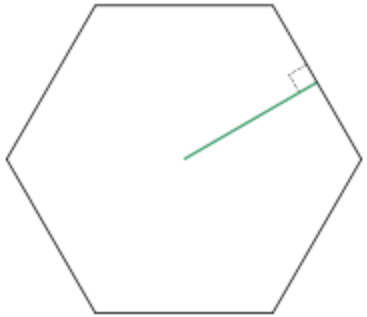


Linha poligonal aberta não-simples

Polígonos

- Polígonos regulares
  - São polígonos simples, equiangulares e equiláteros; podem (ou não) ser convexos





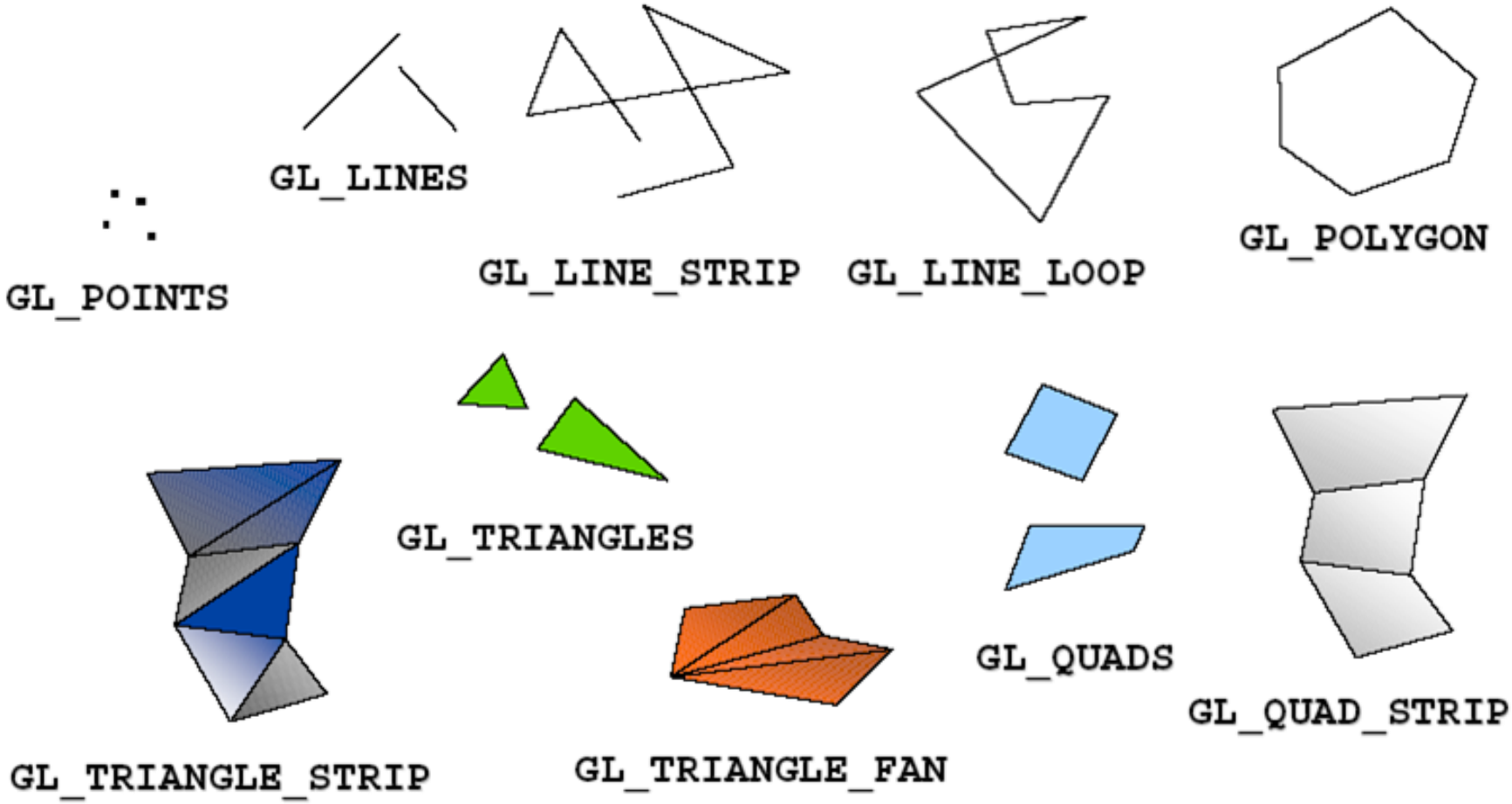
Polígonos

Nome	Número de lados	Nome	Número de lados
triângulo	3	quadrilátero	4
pentágono	5	hexágono	6
heptágono	7	octógono	8
Eneágono	9	decágono	10
Hendecágono	11	dodecágono	12
Tridecágono	13	Tetradecágono	14
Pentadecágono	15	hexadecágono	16
Heptadecácogo	17	octodecágono	18
eneadecágono	19	icoságono	20
triacontágono	30	tetracontágono	40
pentacontágono	50	hexacontágono	60
heptacontágono	70	octacontágono	80
eneacontágono	90	hectágono	100
quilógono	1000	googólgono	10^{100}

Fonte: wikipedia

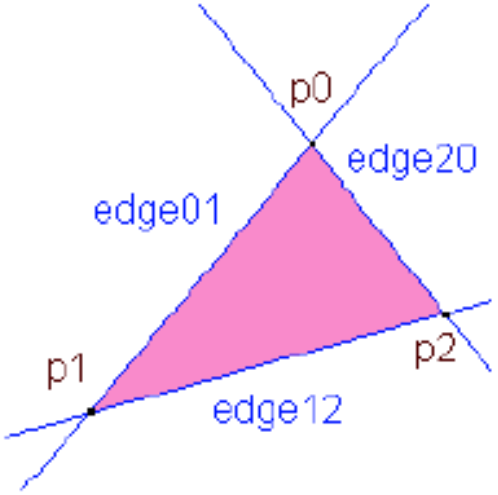
Polígonos

- Exemplo: primitivas geométricas em OpenGL



Triângulos

- Polígono mais simples
  - os três vértices definem **sempre** um plano



$$\begin{bmatrix} x_1 \\ y_1 \end{bmatrix}, \begin{bmatrix} x_2 \\ y_2 \end{bmatrix}, \begin{bmatrix} x_3 \\ y_3 \end{bmatrix}$$

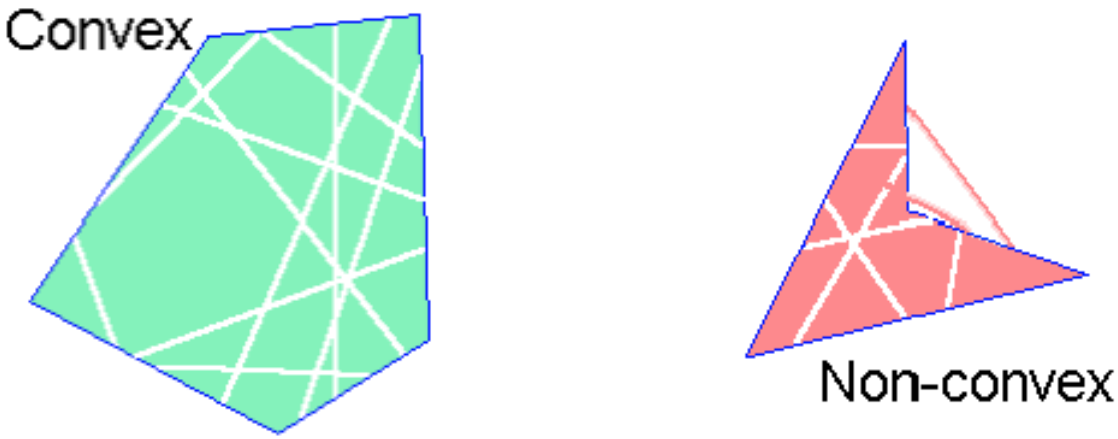
- Formas implícitas para os três lados:

$$A_1 x + B_1 y + C_1 = 0, \quad A_2 x + B_2 y + C_2 = 0, \quad A_3 x + B_3 y + C_3 = 0$$

Triângulos

- **Outra vantagem:** um triângulo é sempre convexo
- O que significa "convexo"?

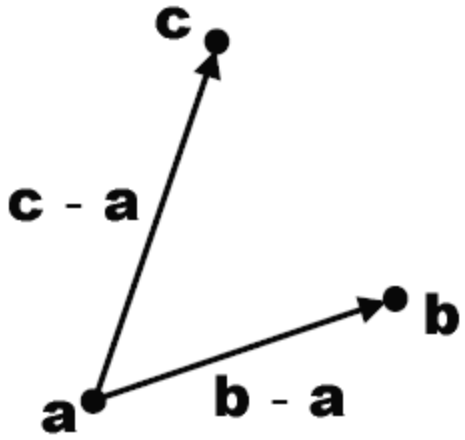
Um conjunto é convexo se qualquer segmento de recta ligando dois pontos na fronteira estiver inteiramente contido no conjunto (ou na fronteira)



Coordenadas baricêntricas

- Qualquer ponto interior pode ser expresso como combinação linear convexa dos vértices: **coordenadas baricêntricas**

- Este sistema de coordenadas é invariante a translacções, rotações e escalamentos



$$\mathbf{p} = \mathbf{a} + \beta(\mathbf{b} - \mathbf{a}) + \gamma(\mathbf{c} - \mathbf{a})$$

Coordenadas baricêntricas

- Manipulando algebricamente a equação:

$$\begin{aligned}\mathbf{p} &= \mathbf{a} + \beta(\mathbf{b} - \mathbf{a}) + \gamma(\mathbf{c} - \mathbf{a}) \\ &= (1 - \beta - \gamma)\mathbf{a} + \beta\mathbf{b} + \gamma\mathbf{c} \\ &= \alpha\mathbf{a} + \beta\mathbf{b} + \gamma\mathbf{c}\end{aligned}$$

$$\begin{aligned}0 < \alpha < 1, \\ 0 < \beta < 1, \\ 0 < \gamma < 1.\end{aligned}$$

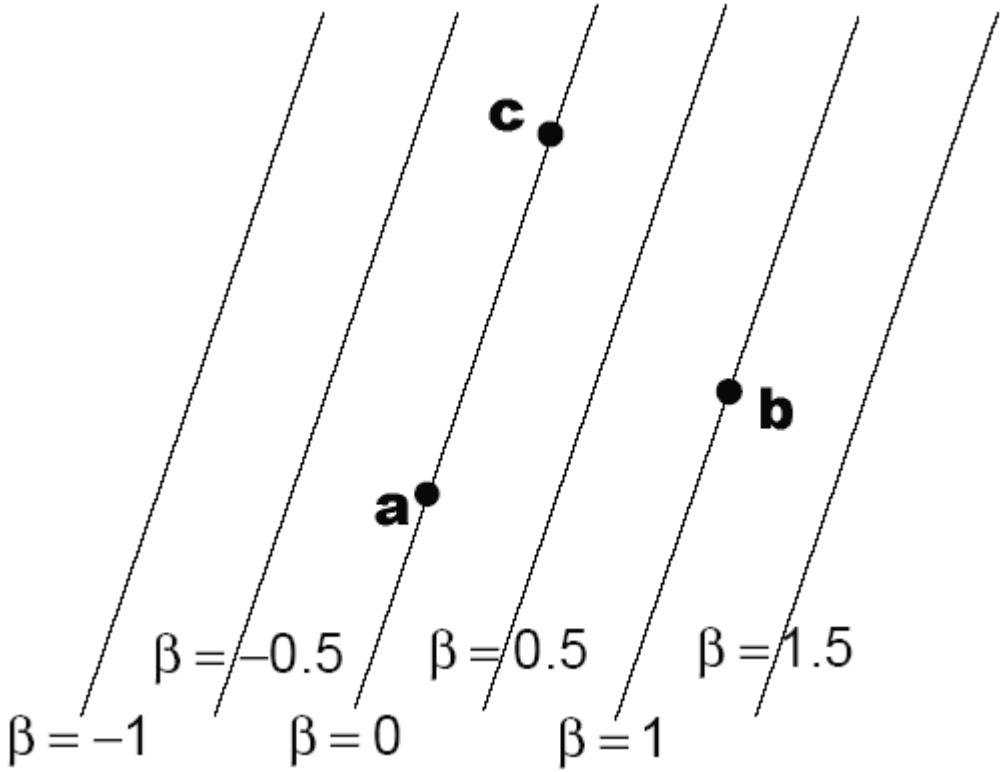
$$\mathbf{p}(\alpha, \beta, \gamma) = \alpha\mathbf{a} + \beta\mathbf{b} + \gamma\mathbf{c}$$

$$\alpha + \beta + \gamma = 1$$

- Trata-se de uma combinação linear convexa, ou de uma combinação afim (*affine combination*) dos vértices.

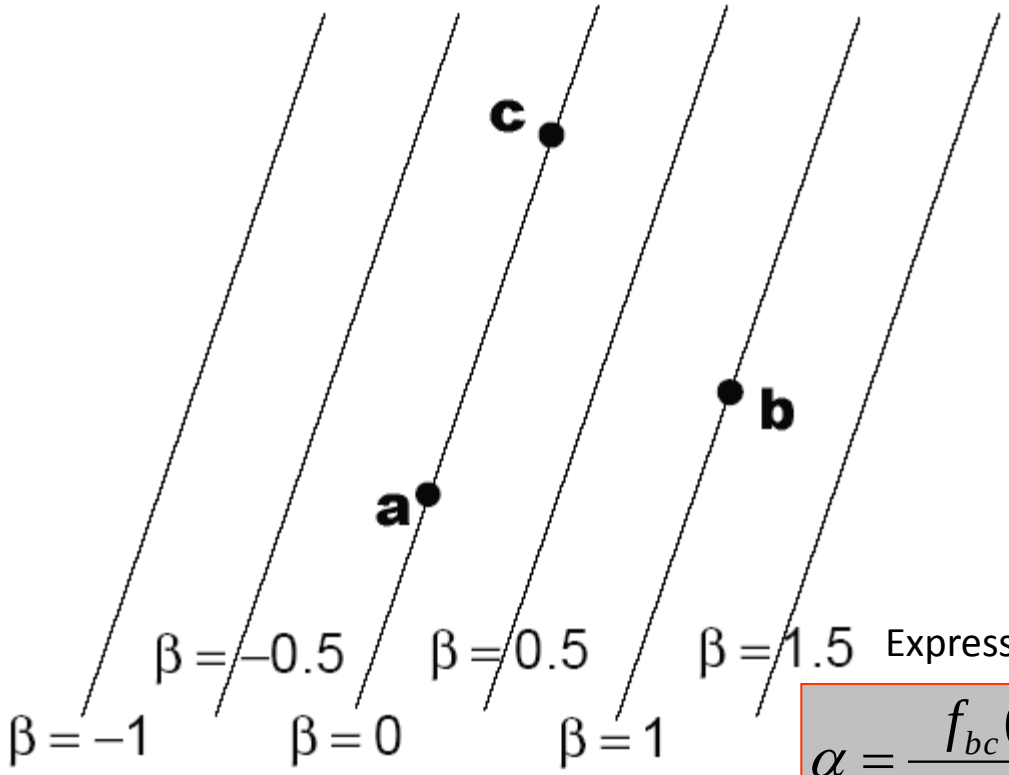
Coordenadas baricêntricas

- Significado geométrico de cada coordenada
  - distância (com sinal +/-) ao lado oposto



Coordenadas baricêntricas

- Basta usar a equação implícita $f_{ac}(x,y)$ do segmento de recta **ac**



$$f(x,y) = 0 \Leftrightarrow kf(x,y) = 0$$

$$kf_{ac}(x,y) = \beta$$

$$kf_{ac}(x_b,y_b) = 1 \Leftrightarrow k = \frac{1}{f_{ac}(x_b,y_b)}$$

Expressões equivalentes para as outras coordenadas:

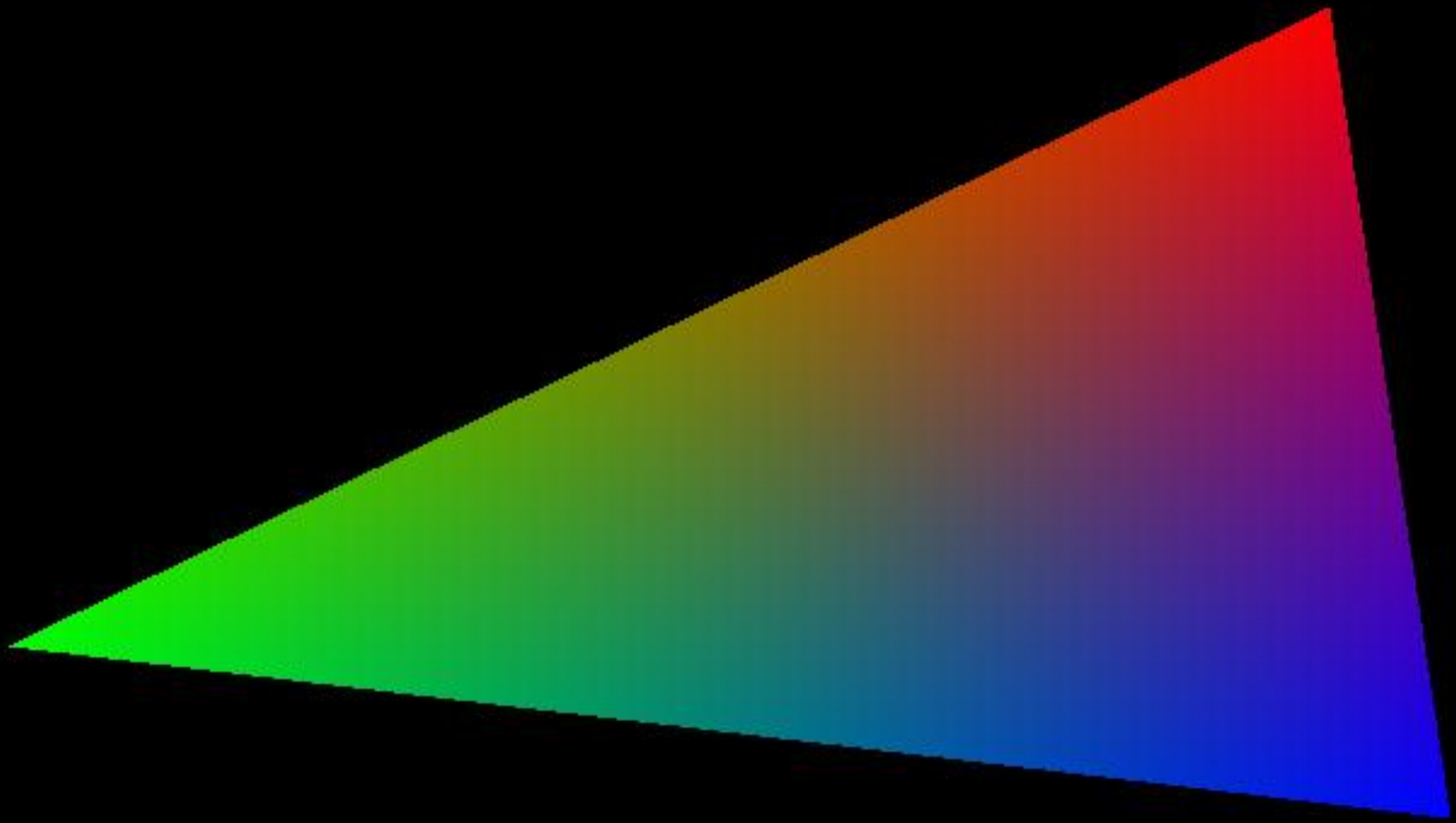
$$\alpha = \frac{f_{bc}(x,y)}{f_{bc}(x_a,y_a)} \qquad \beta = \frac{f_{ac}(x,y)}{f_{ac}(x_b,y_b)} \qquad \gamma = 1-\alpha-\beta$$

Rasterização de triângulos

- **Objectivo:** rasterizar e colorir o interior de um triângulo
- Algoritmo de interpolação de *Gouraud* (faz parte de um procedimento mais geral, bem conhecido, designado *Gouraud shading*)
  - Utiliza coordenadas bariêntricas

```
for all x do
   for all y do
       compute (alpha, beta, gamma) for (x,y)
       if ( 0 < alpha < 1 and
            0 < beta < 1 and
            0 < gamma < 1 ) then
              c = alpha c0 + beta c1 + gamma c2
              drawpixel(x,y) with color c
```

Algoritmo *Gouraud shading* em *pseudo-código*



Representação alternativa

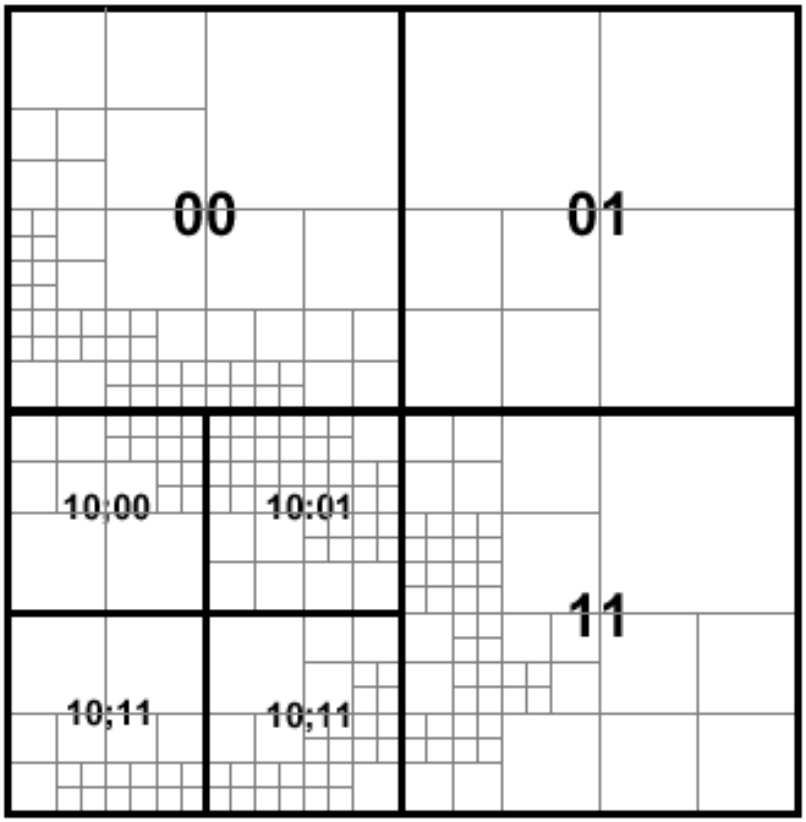
- Em vez de usar polígonos, podemos enumerar todos as regiões do plano ocupadas por um objecto geométrico.

- Será eficiente?
- Será útil?

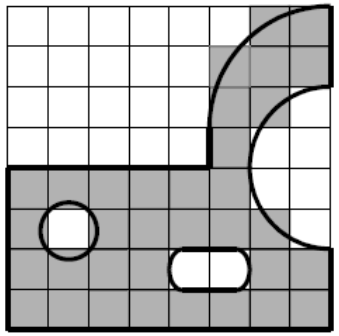
Depende!

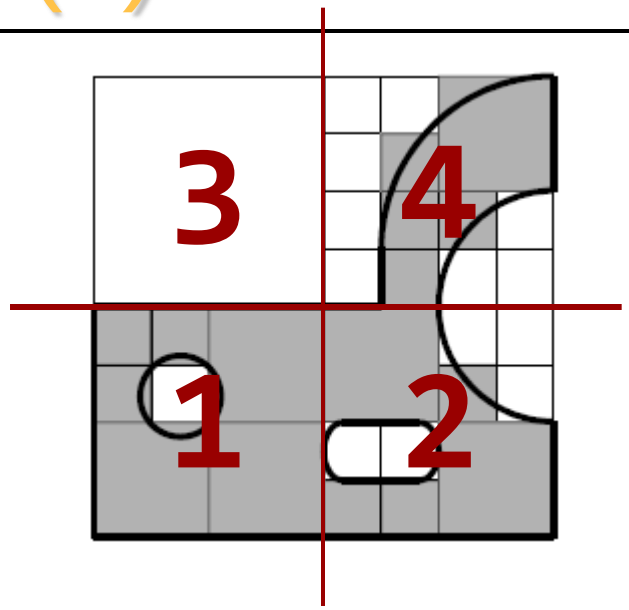
Quadtrees (1)

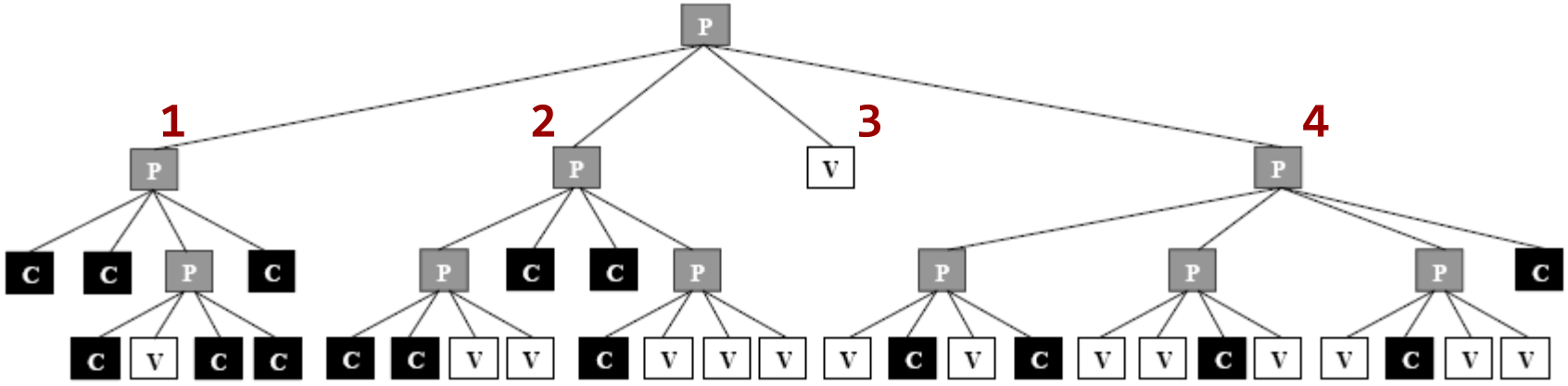
- *Quadtree*: estrutura de dados em árvore, contendo até 4 filhos por nó



Quadtrees (2)

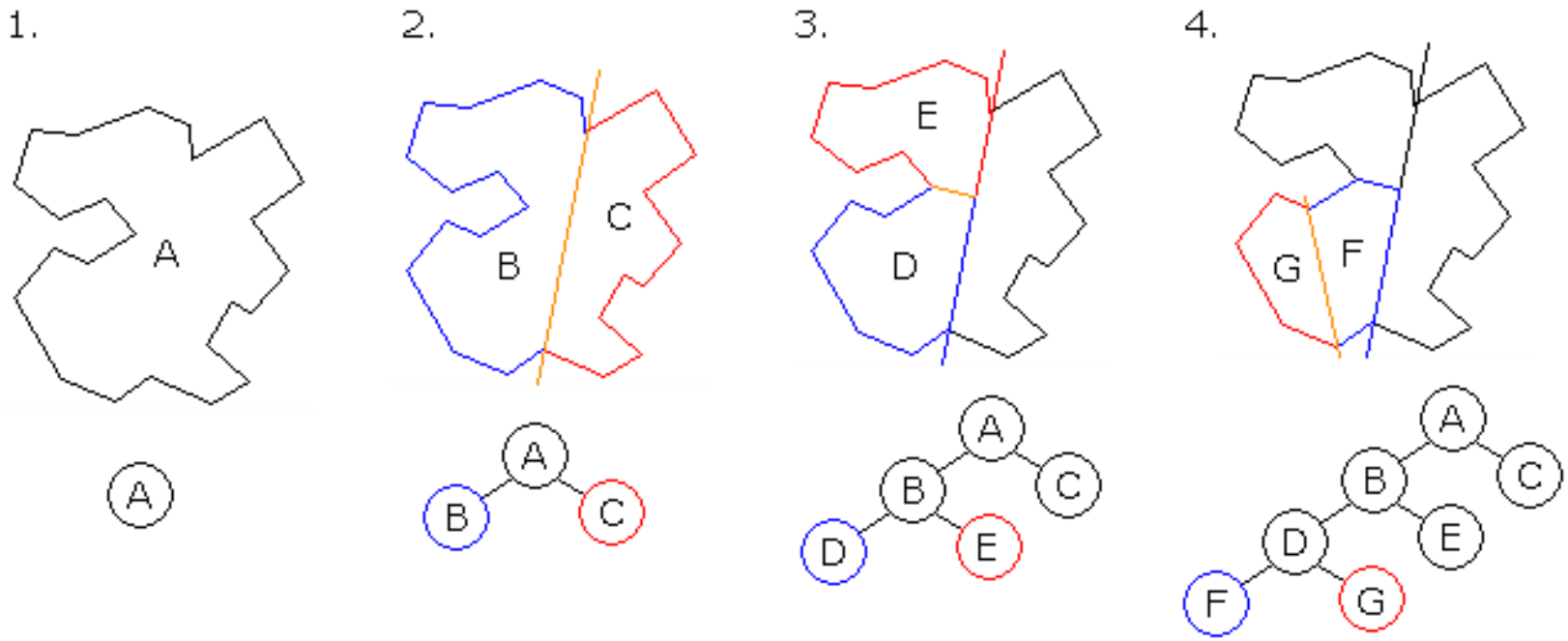




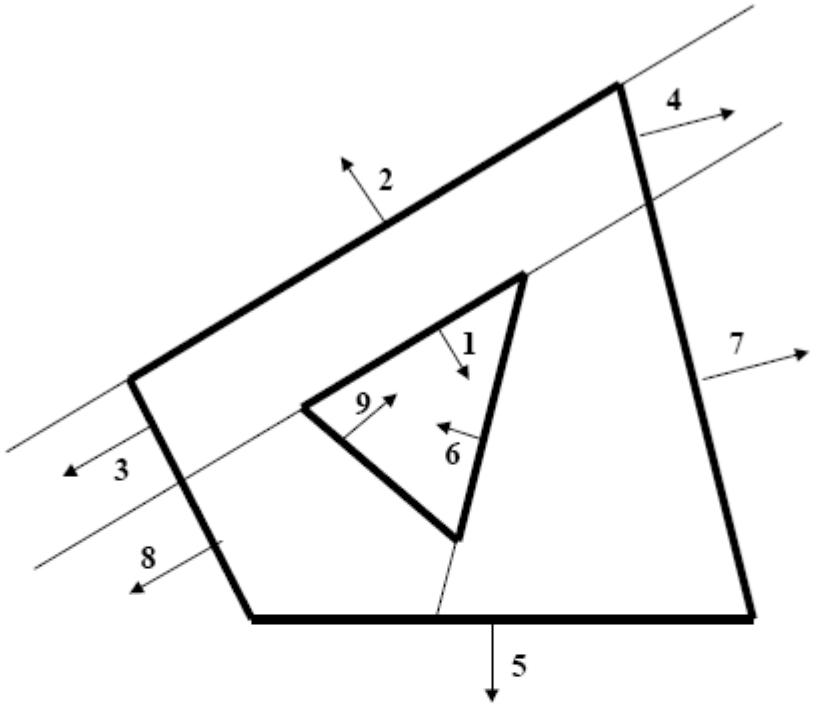


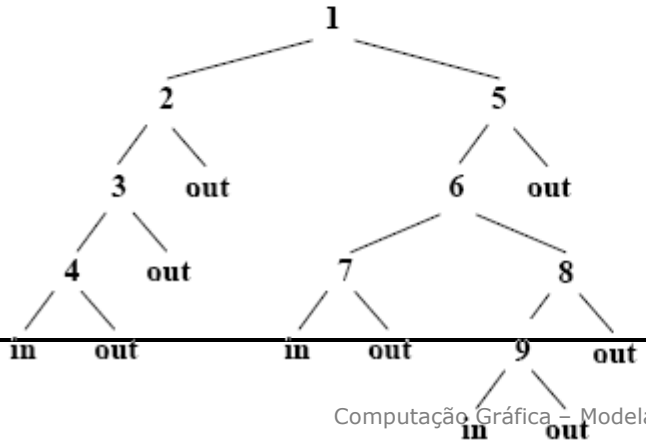
BSP *trees* (1)

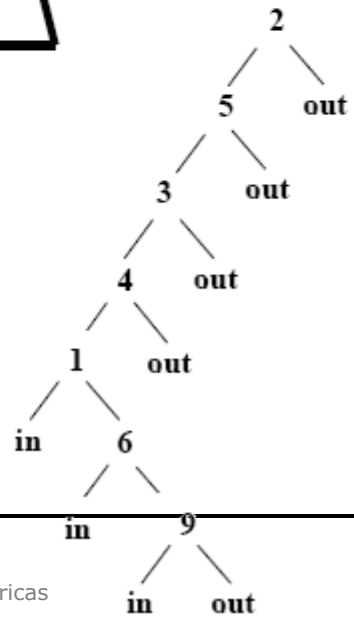
- BSP: *Binary Space Partitioning*
  - Existem diversas utilizações para BSP
  - Neste caso, para decompor um polígono em sub-polígonos convexos



BSP *trees* (2)


1
2
3
4
5
6
7
8
9


1
2
5
3
out
6
out
4
out
7
8
in
out
in
out
9
out
in
out


2
5
out
3
out
4
out
1
out
in
6
in
9
in
out

Sumário

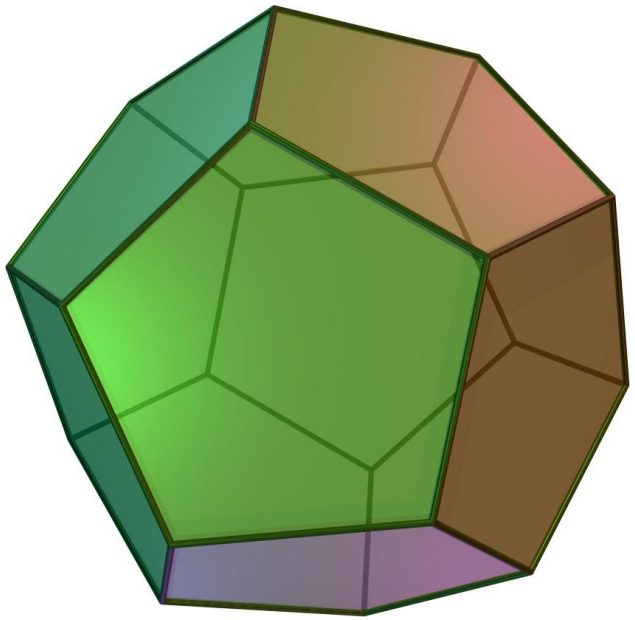
- Formas 2D
  - Representação de rectas e curvas
  - Polígonos e Triângulos
    - Coordenadas baricêntricas
    - Rasterização de triângulos
  - Partição do espaço (plano): *Quadtrees* e BSP *trees*

- Formas 3D
  - Poliedros
  - *Meshes*
  - Partição do espaço: *Octrees* (e BSP *trees*)
  - Represenção do espaço: *Voxels*
  - Modelação de Sólidos: CSG

- Normais: ponte entre forma e luz

Poliedros

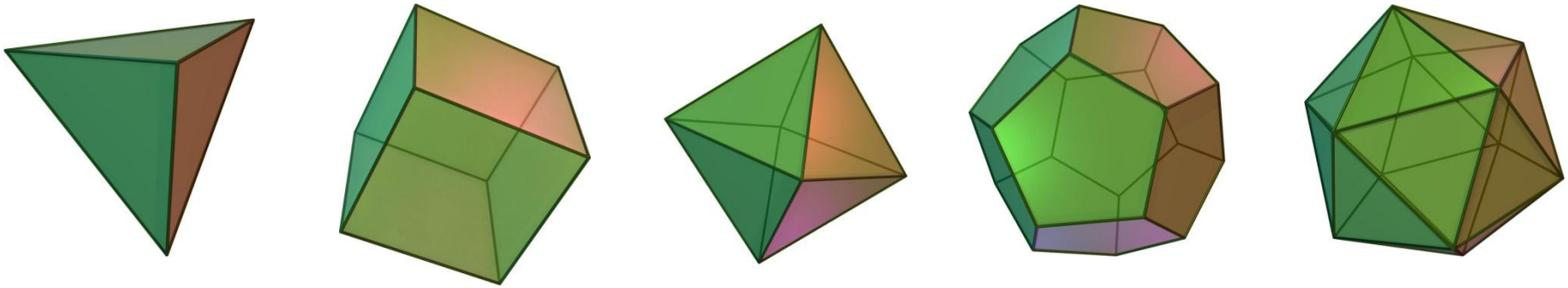
- Definição de **poliedro**
  - *Forma tridimensional composta por um número finito de faces poligonais, cada uma formando um plano, que partilham arestas e vértices por forma a delimitar um volume.*



- Generalização tridimensional do conceito de polígono
- Os poliedros podem, ou não, ser regulares (i.e. com faces que são polígonos regulares) e convexos

Poliedros

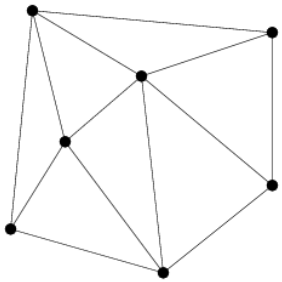
- Poliedros regulares convexos ou sólidos Platónicos:

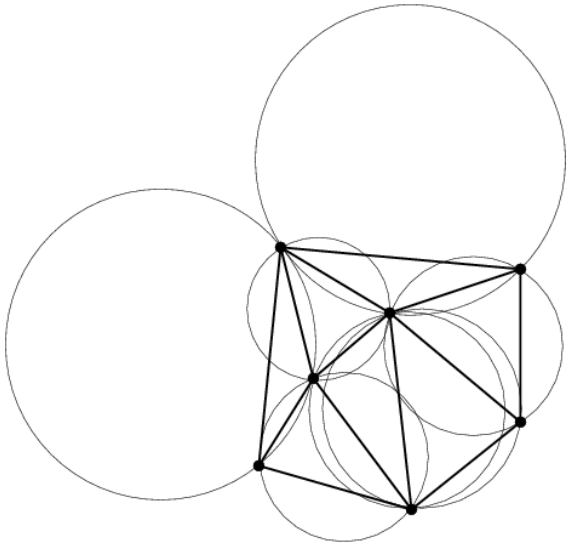


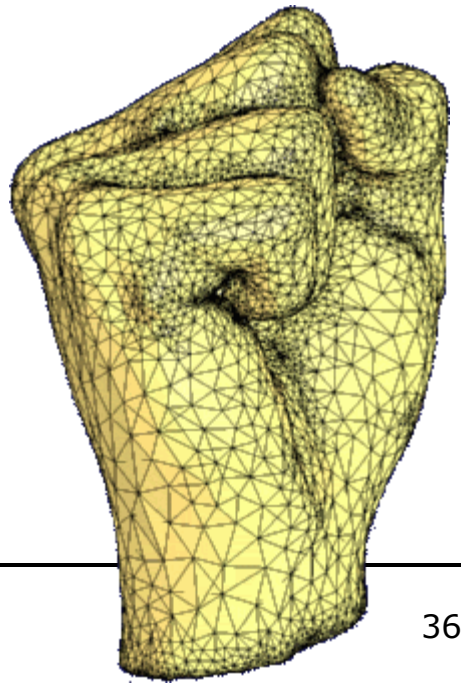
Tetraedro **cubo** **octaedro** **dodecaedro** **icosaedro**

Malhas (*meshes*)

- Malha ou (*mesh*): divisão de um objecto geométrico em polígonos
  - No caso de triângulos (um caso particular de *simplex*), chama-se à malha uma triangulação
  - Preferencialmente, usam-se triangulações de Delaunay, nas quais a circunsfera de um triângulo não contêm vértices de nenhum outro triângulo – isto leva a triângulos equilibrados

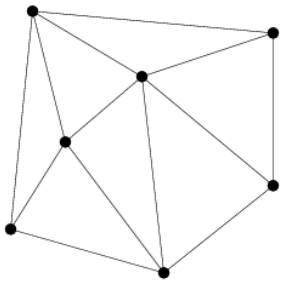


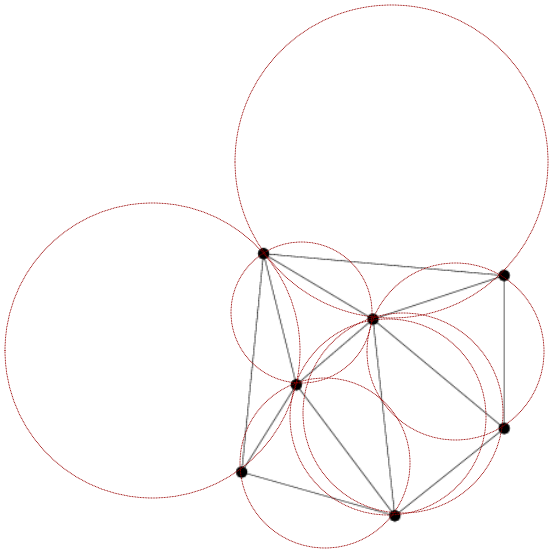


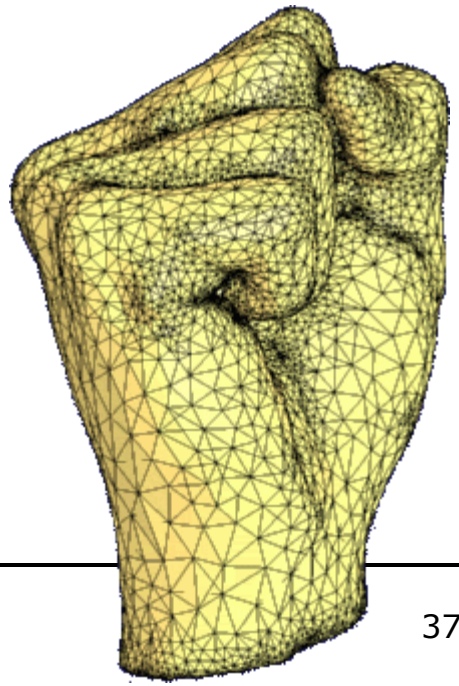


Malhas (*meshes*)

- Malha ou (*mesh*): divisão de um objecto geométrico em polígonos
  - No caso de triângulos (um caso particular de *simplex*), chama-se à malha uma triangulação
  - Preferencialmente, usam-se triangulações de Delaunay, nas quais a circunsfera de um triângulo não contêm vértices de nenhum outro triângulo – isto leva a triângulos equilibrados







Malhas *(meshes)*

- Estruturas de dados
  - Lista de coordenadas dos vértices de cada face
    Ineficiente

```
P1 = (x1, y1, z1), (x2, y2, z2), (x3, y3, z3), (x4, y4, z4)
```

  - Lista única de vértices

```
V = (V1 (x1, y1, z1), V2(x2, y2, z2) ....)
```

    - Depois: lista de faces

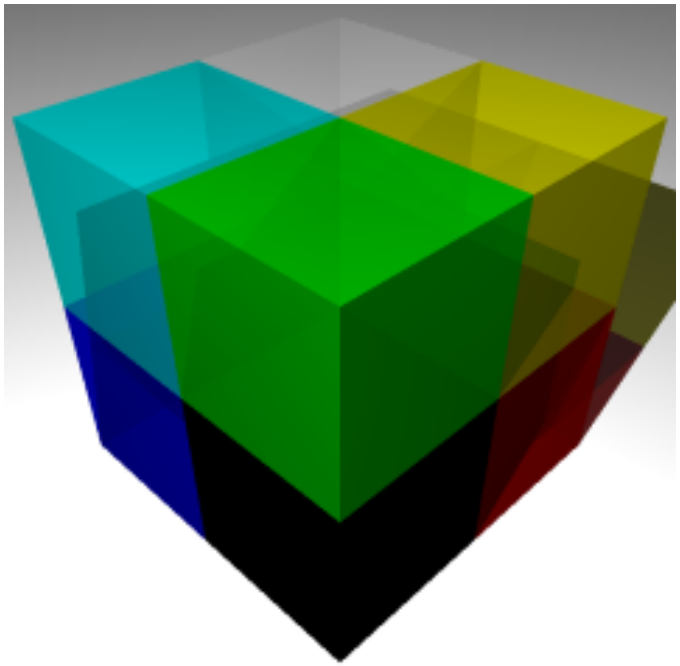
```
P1 = (V1, V2, V3, V4)
```

    - ou: lista de arestas

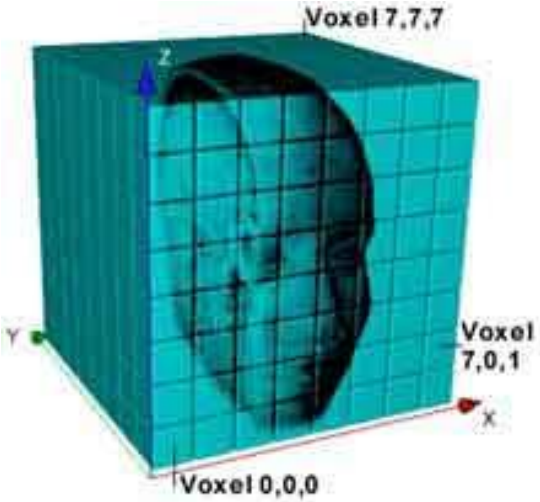
```
E1 = (V1, V2)    P1 = (E1, E2, E3, E4)
```

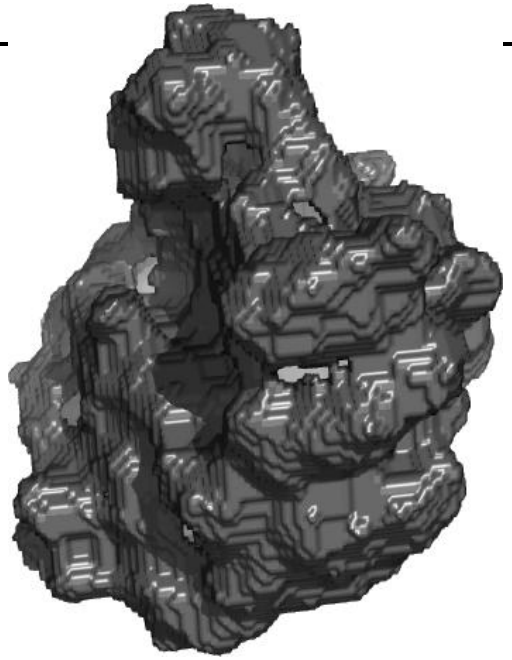
Octrees

- *Octree*: estrutura de dados em árvore, contendo até 8 filhos por nó



Voxels





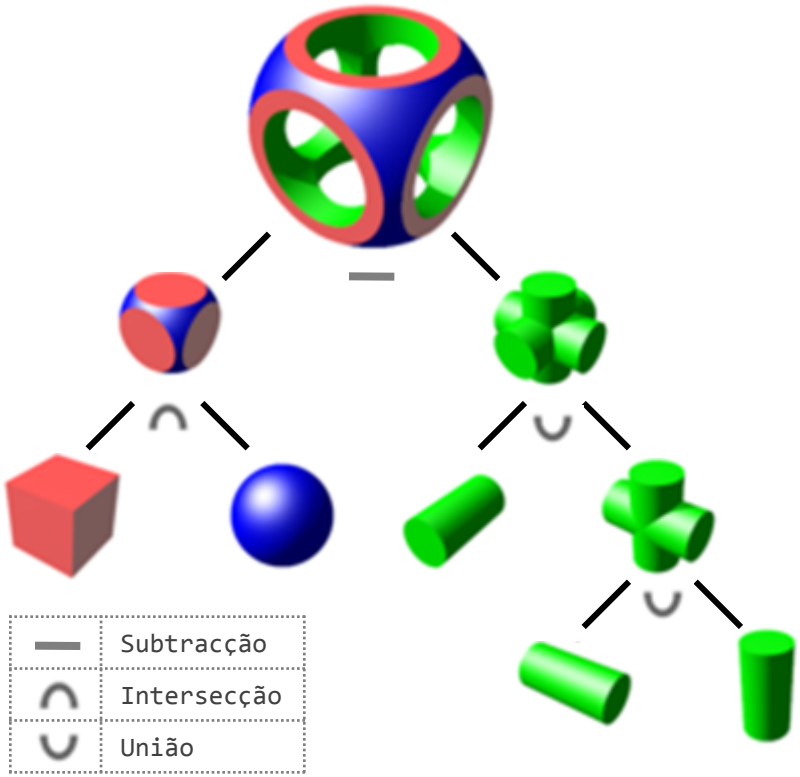
Macromolécula

Exemplo: *Visible Human Project*
(digitalização do interior do corpo humano)

Links:
http://www.nlm.nih.gov/research/visible/
http://www.voxel-man.de/vm-web-neu/galerie/io.en.html

Modelação de Sólidos: CSG

- Outra forma de modelar/representar sólidos é através de operações binárias entre sólidos
  - Esta forma de modelação designa-se de CSG: Constructive Solid Geometry
  - As operações binárias típicas são:
    - União
    - Intersecção
    - Subtracção
  - Não é possível aplicar operações cujo resultado não seja um sólido
    - Ex: partir um cílindro em dois



Sumário

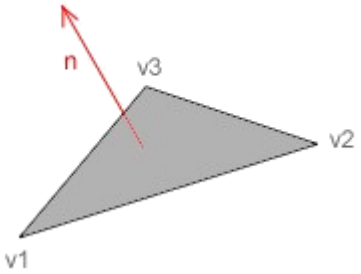
- Formas 2D
  - Representação de rectas e curvas
  - Polígonos e Triângulos
    - Coordenadas baricêntricas
    - Rasterização de triângulos
  - Partição do espaço (plano): *Quadtrees* e BSP *trees*

- Formas 3D
  - Poliedros
  - *Meshes*
  - Partição do espaço: *Octrees* (e BSP *trees*)
  - Represenção do espaço: *Voxels*
  - Modelação de Sólidos: CSG

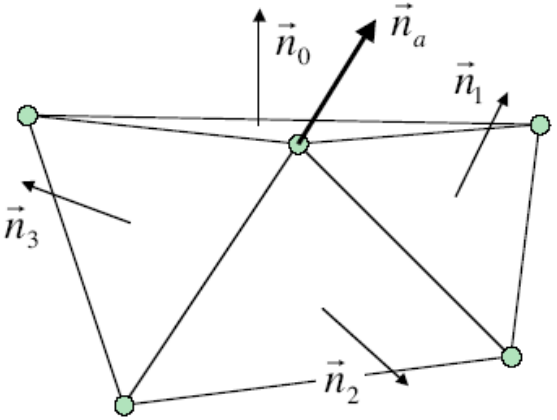
- Normais: ponte entre forma e luz

Normais

- Os vectores normais são necessários em muitos algoritmos de iluminação e sombreamento



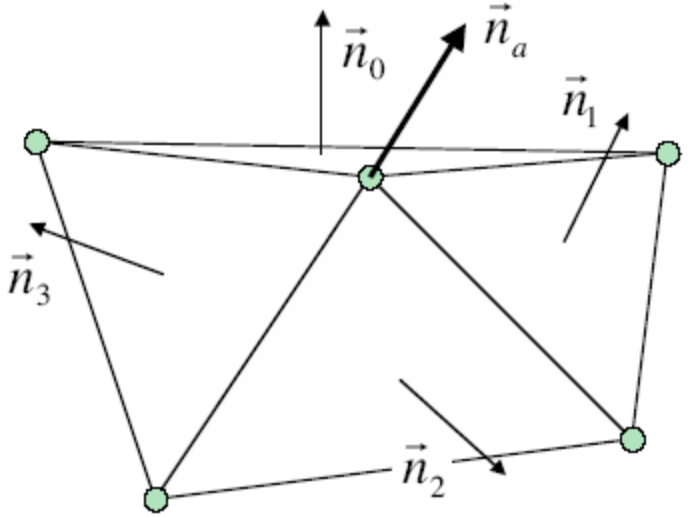
  - O vector normal a um triângulo pode calcular-se através do produto externo entre dois lados
  - Contudo, se o ângulo for muito pequeno, pode levar a problemas numéricos
  - Além disso: qual é a normal num vértice de uma malha?



Cálculo de normais

- **Método 1:**
  - Média das normais das faces incidentes

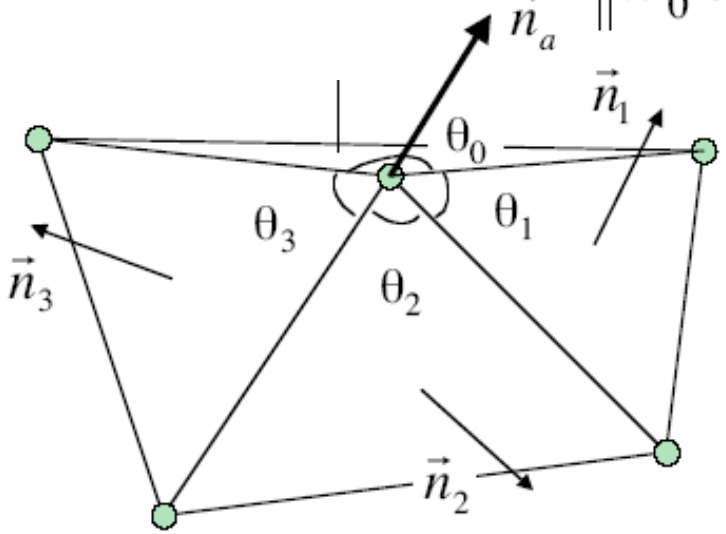
$$\vec{n}_a = \frac{\vec{n}_0 + \vec{n}_1 + \vec{n}_2 + \vec{n}_3}{\left\| \vec{n}_0 + \vec{n}_1 + \vec{n}_2 + \vec{n}_3 \right\|}$$



Cálculo de normais

- **Método 2:**
  - Média <u>ponderada</u> das normais das faces incidentes

- Method 2 $\vec{n}_a = \dfrac{w_0\vec{n}_0 + w_1\vec{n}_1 + w_2\vec{n}_2 + w_3\vec{n}_3}{\left\| w_0\vec{n}_0 + w_1\vec{n}_1 + w_2\vec{n}_2 + w_3\vec{n}_3 \right\|}$

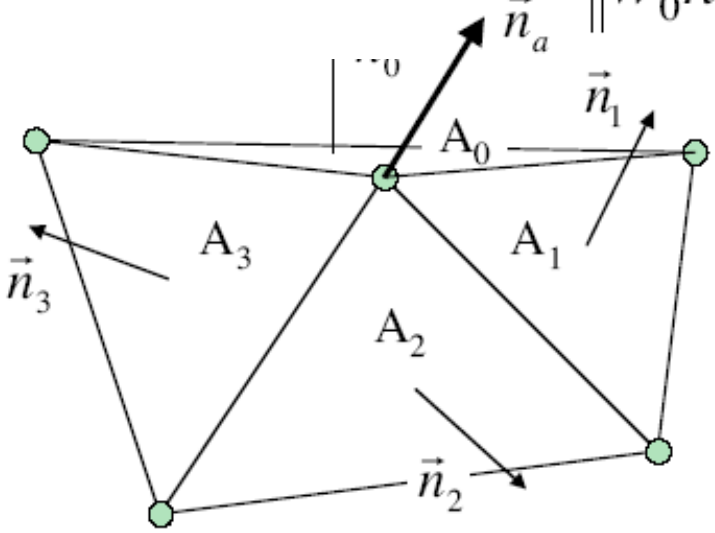


$$w_i = \frac{\boldsymbol{\theta}_i}{360°};$$
$$i = \{0 \ldots 3\}$$

Cálculo de normais

- **Método 3:**
  - Média ponderada, mas tendo em conta que os ângulos podem não somar 360º

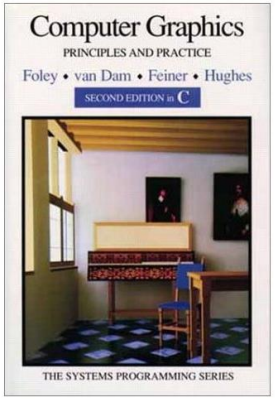
- Method 3 $\vec{n}_a = \dfrac{w_0\vec{n}_0 + w_1\vec{n}_1 + w_2\vec{n}_2 + w_3\vec{n}_3}{\left\| w_0\vec{n}_0 + w_1\vec{n}_1 + w_2\vec{n}_2 + w_3\vec{n}_3 \right\|}$



$$w_i = \frac{A_i}{\sum_{j=0}^{j=3} A_j};$$

$$i = \{0 \ldots 3\}$$

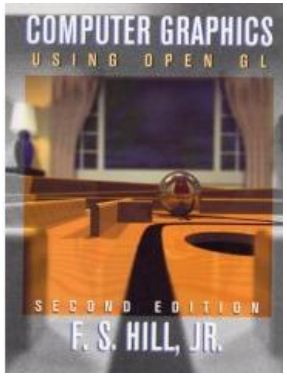
Referências



Computer Graphics: Principles and Practice in C,

James D. Foley, Andries van Dam, Steven K. Feiner, John F. Hughes, Addison-Wesley Professional; 2nd edition (1995)

-> Capítulo 12



Computer Graphics using OpenGL,

Francis S. Hill, Prentice-Hall, 2nd Edition (2003)

- Wikipedia, http://en.wikipedia.org/, 2006
- http://www.nlm.nih.gov/research/visible/
- http://www.voxel-man.de/vm-web-neu/galerie/io.en.html