Representação de malhas poligonais (surface meshes)

SCC-250 Computação Gráfica

Maria Cristina F. Oliveira

Modelo

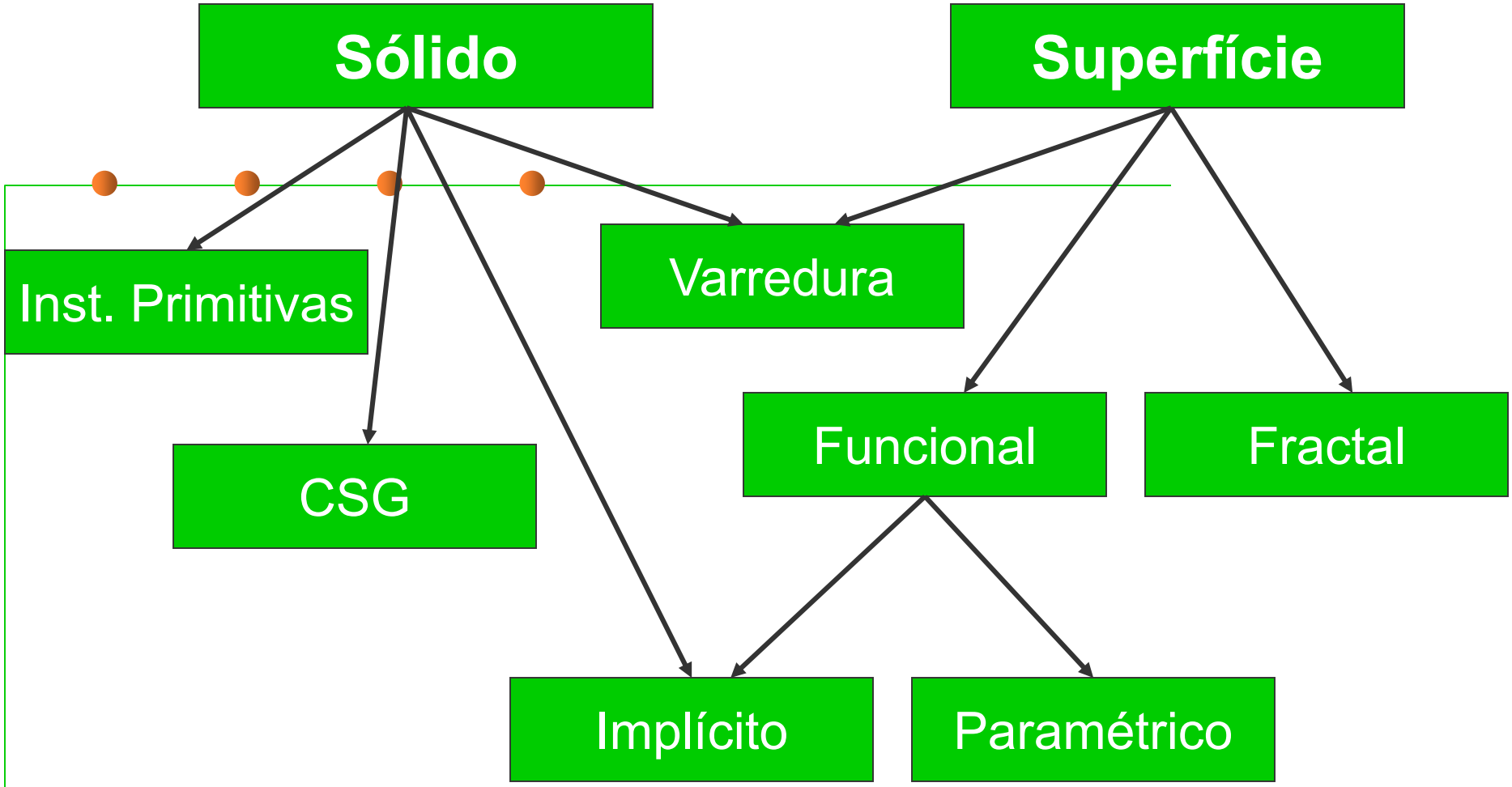
- Representação construída para tornar mais fácil a observação/análise de um objeto/fenômeno
  - Nível de detalhe definido pelas aplicações que o utilizam
  - Problemas práticos em CG: modelos geométricos

Modelagem Geométrica

- Início dos anos 70
- Coleção de métodos/algoritmos usados para descrever a forma e outras características geométricas de um objeto
- Modelos usados para renderização e para simular processos dinâmicos
- Sistema de modelagem geométrica: sistema que permite a criação, modificação e acesso à representação de objetos por meio de modelos geométricos

Modelos Geométricos

- Cenas gráficas podem conter muitos tipos diferentes de objetos e materiais
  - Não existe uma maneira única de descrever e representar todos os tipos de objetos
- Descrição vs. Representação
  - Descrição do objeto pelo usuário: processo de modelagem
  - Representação do objeto no sistema gráfico: como manter as informações necessárias para renderizar o objeto (e usar em simulações, por ex.)


Sólido
Superfície
Inst. Primitivas
Varredura
CSG
Funcional
Fractal
Implícito
Paramétrico

Sólido

Superfície

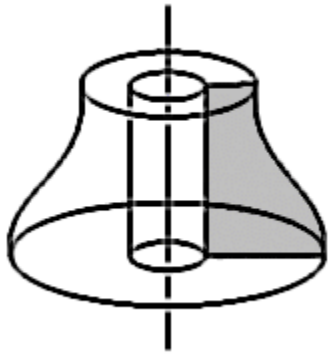
Inst. Primitivas

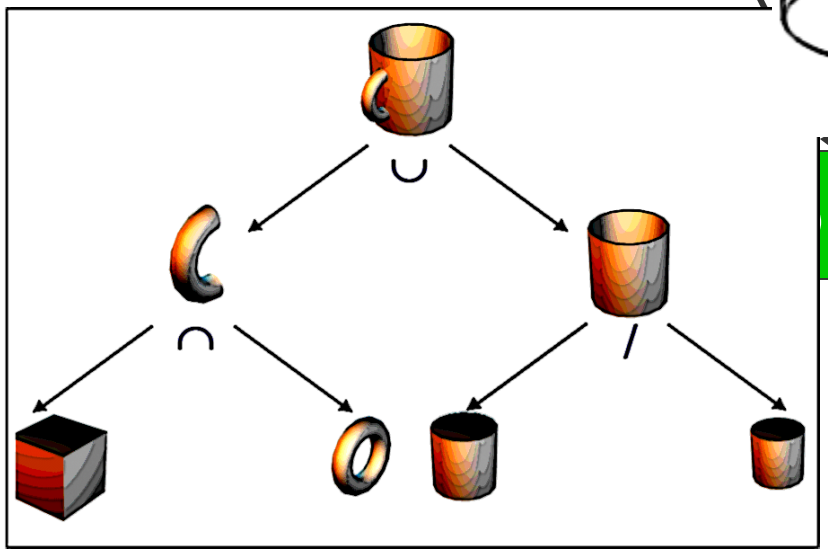
Varredura

CSG

cional

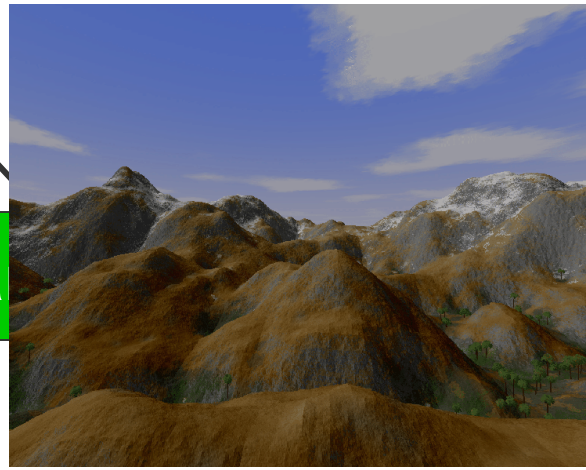
Fractal

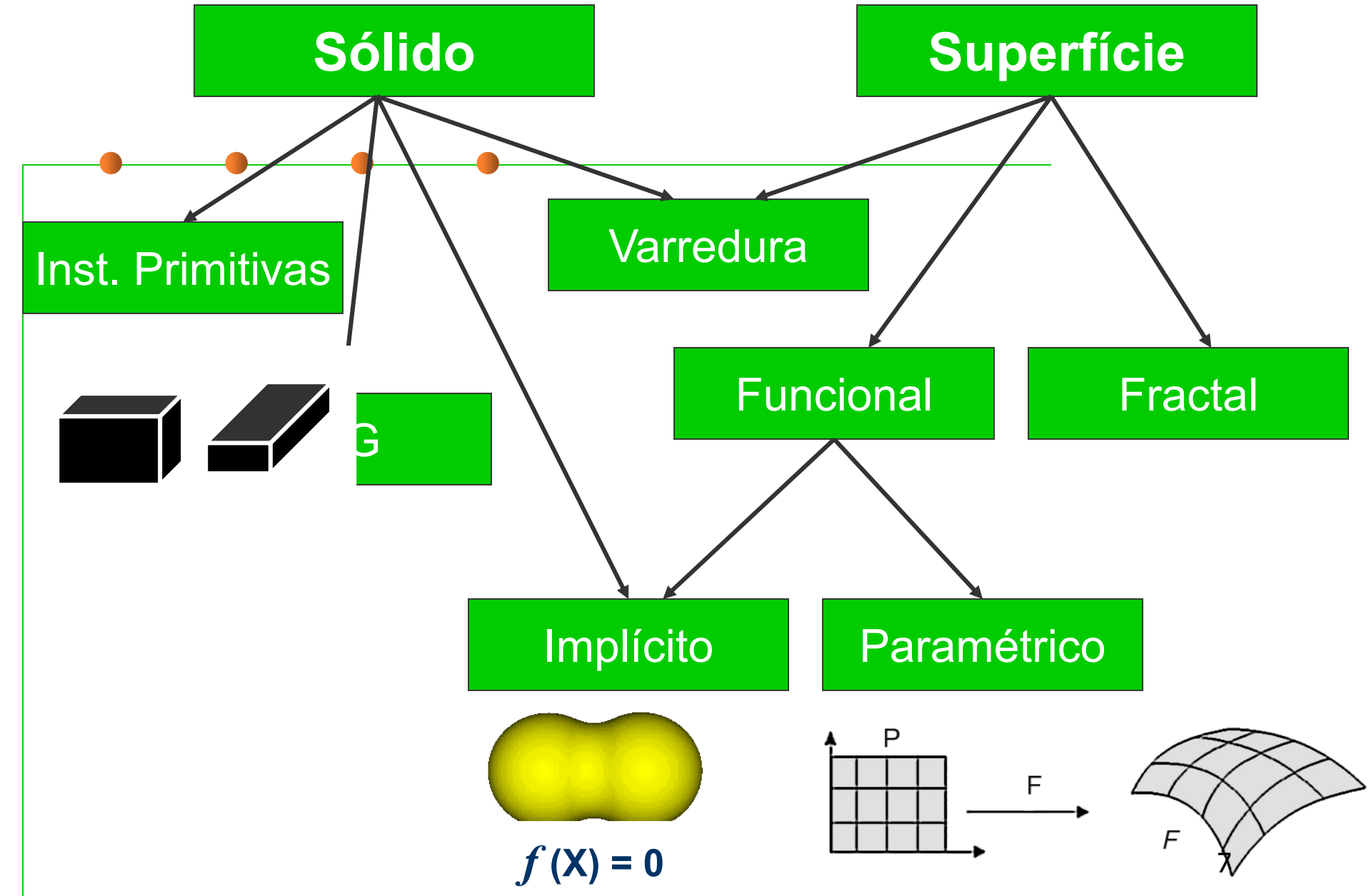




lícito

Para




Sólido
Superfície
Inst. Primitivas
Varredura
Funcional
Fractal
G
Implícito
Paramétrico
P
F
F
f (X) = 0

Superfícies vs Volumes

- Objetos tridimensionais
  - Representados pela sua superfície
    - representações por fronteira: objeto 3D descrito como um conjunto de superfícies que separa o seu interior do meio externo
    - Primitivas geralmente são triângulos, quadriláteros, ou superfícies paramétricas (sempre superfícies 2D)
  - *Surface mesh* & *surface rendering*
  - Pipeline OpenGL

Superfícies vs Volumes

- Objetos tridimensionais
  - Representação explícita do exterior e do interior
    - Modelos volumétricos
    - Representações por particionamento espacial que descrevem propriedades do interior dos objetos, em geral descritas por primitivas 3D (p.ex., tetraedros, ou cubos)
    - Representações por malhas volumétricas, em que as primitivas são 3D (p.ex. tetraedros)

Volumes

- *Volume representation*
- *Volume rendering*

- https://en.wikipedia.org/wiki/Volume_rendering

Superfícies

- *Surface/mesh representation*
  - "A mesh surface is a discretization of a smooth surface with a collection of connected faces"
- *Surface rendering*

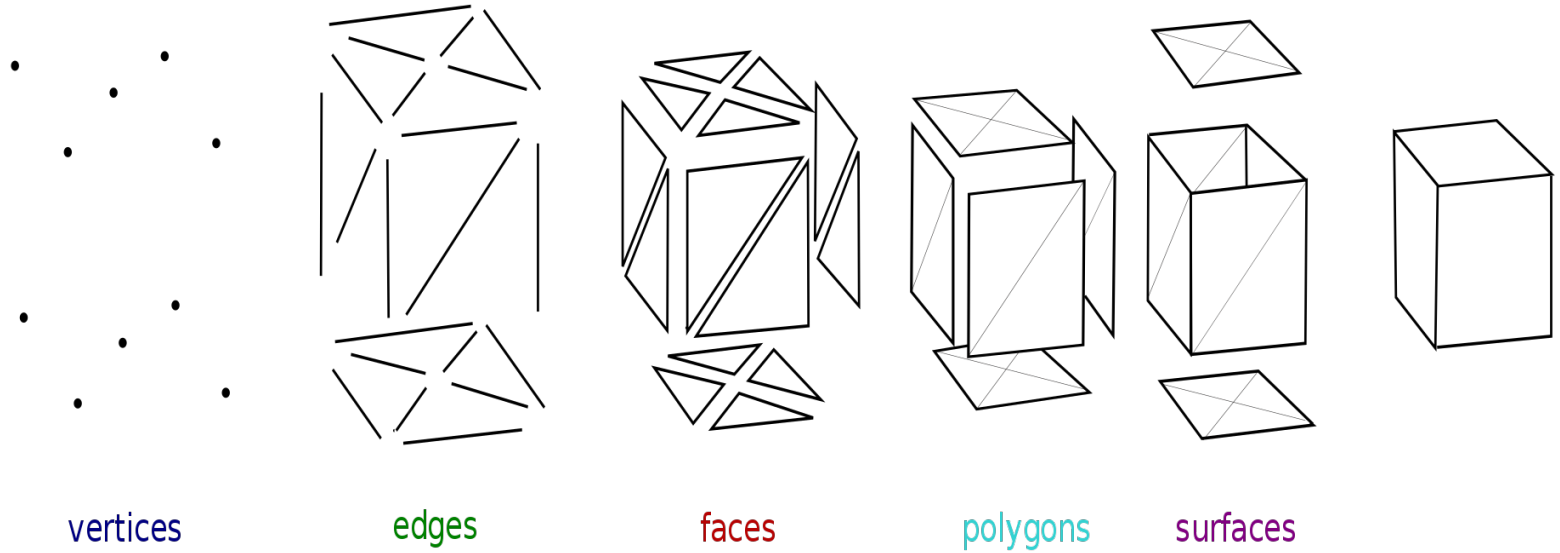
Volume vs *surface rendering*

- https://mphy0026.readthedocs.io/en/latest/graphics/surface_versus_volume.html

Malhas Poligonais

- Existe uma enorme diversidade de representações geométricas em CG
- Mas o pipeline padrão adota uma forma muito simples de representação da fronteira
  - Objetos descritos por malhas poligonais que representam a sua superfície (fronteira)
  - Conjunto de vértices e faces planares (triângulos)
  - Representação adequada para 'rendering' por placas gráficas: objetos gráficos padrão

Malhas poligonais



https://en.wikipedia.org/wiki/Polygon_mesh#/media/File:Mesh_overview.svg

Malhas Poligonais

- Poliedros podem ter uma representação poligonal exata

- Mas, em geral, objetos
  - São descritos por superfícies curvas
  - E decompostos (*discretized*, *tesselated*) para produzir uma representação poligonal aproximada

Malhas Poligonais Exemplos

Fonte:
http://www.bymath.com/studyguide/geo/geo21.htm

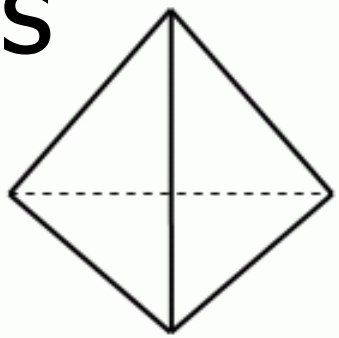


Fig. 99

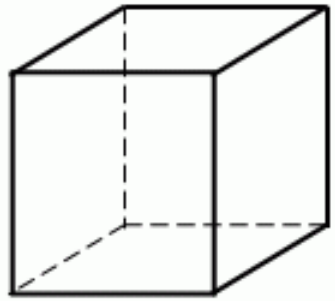


Fig. 100

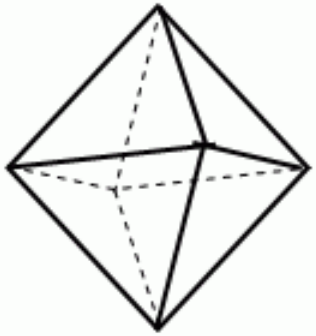


Fig. 101

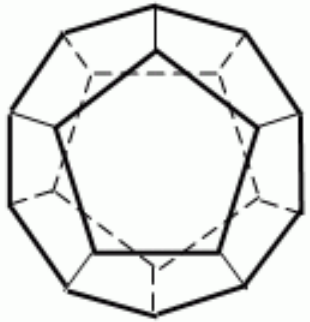


Fig. 102

Poliedros convexos regulares

tetraedro (4 faces, Fig.99)
hexaedro (cubo, 6 faces, Fig.100)
octaedro (8 faces, Fig.101)
dodecaedro (12 faces, Fig.102
icosaedro (20 faces, Fig.103)

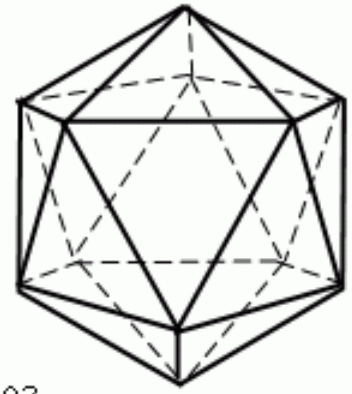


Fig. 103

Malhas Poligonais – Exemplos

Malhas Poligonais – Exemplos

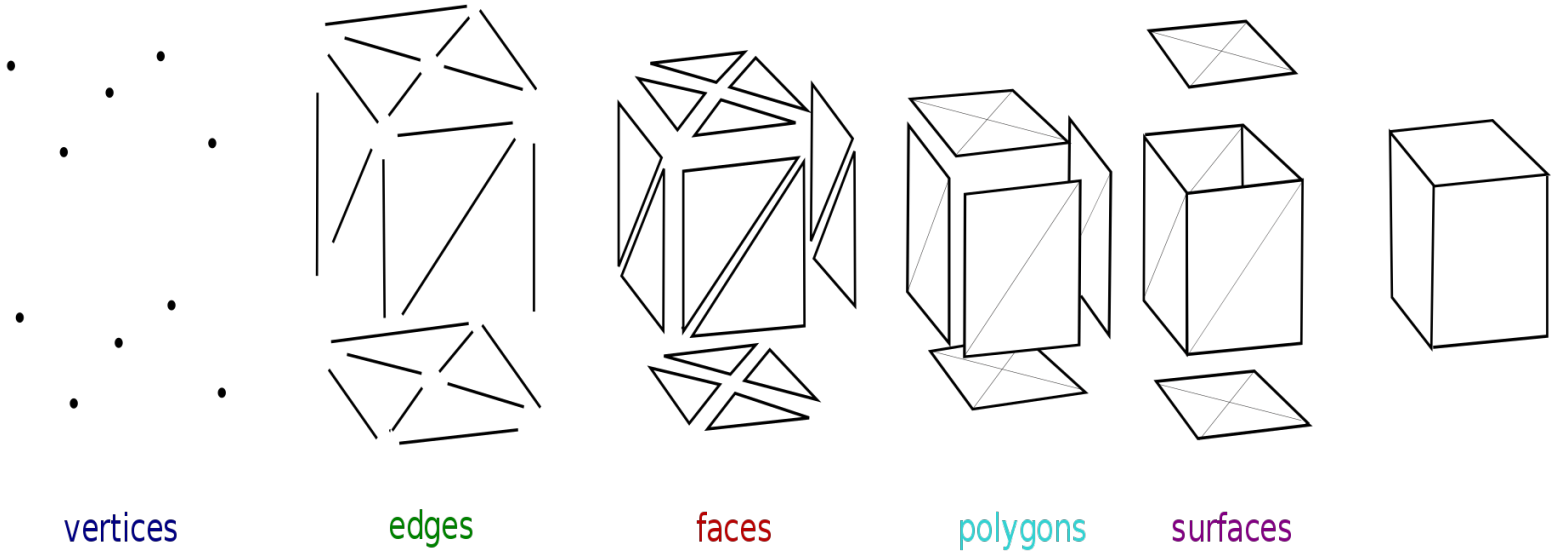
Representação de malhas

- Como armazenar uma malha poligonal, i.e., como descrever um objeto em termos das faces (triângulos) que definem a sua superfície?
  - Diversas soluções possíveis...
- Sugestões??

Estruturas de Dados

- O que armazenar?
  - Informações geométricas: coordenadas dos vértices
  - Informações sobre as relações de adjacência (conectividade)
    - Como os vértices definem as arestas, como as arestas definem as faces
    - Vértices, arestas e faces como elementos topológicos

Malhas poligonais

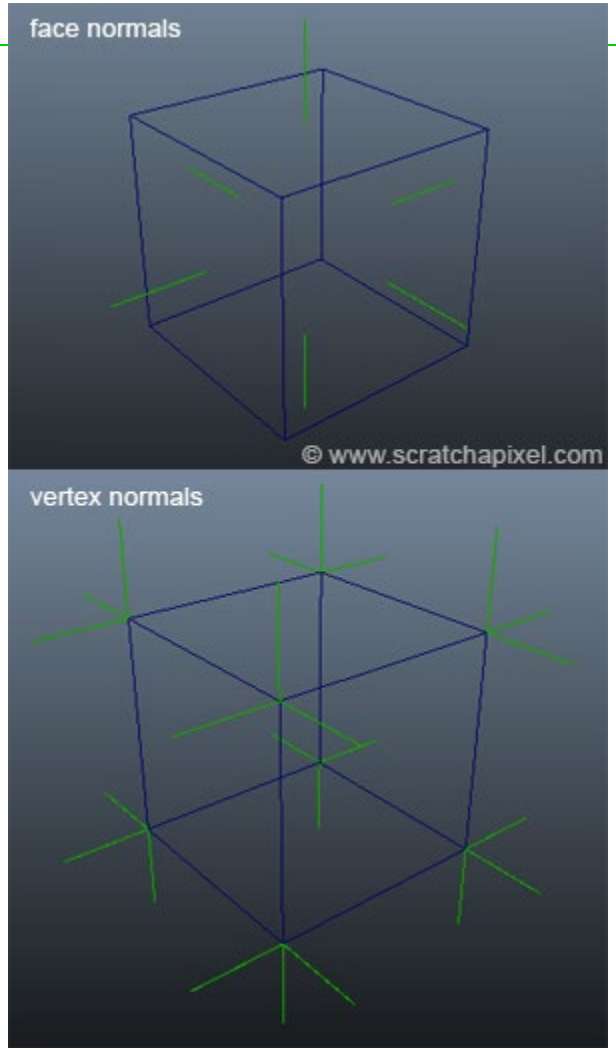


https://en.wikipedia.org/wiki/Polygon_mesh#/media/File:Mesh_overview.svg

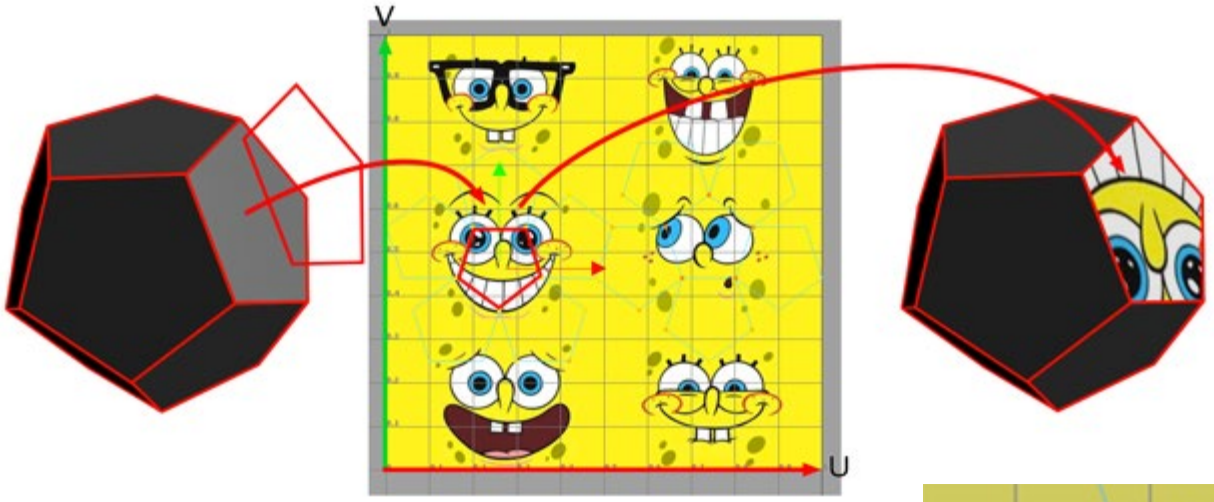
Estruturas de Dados

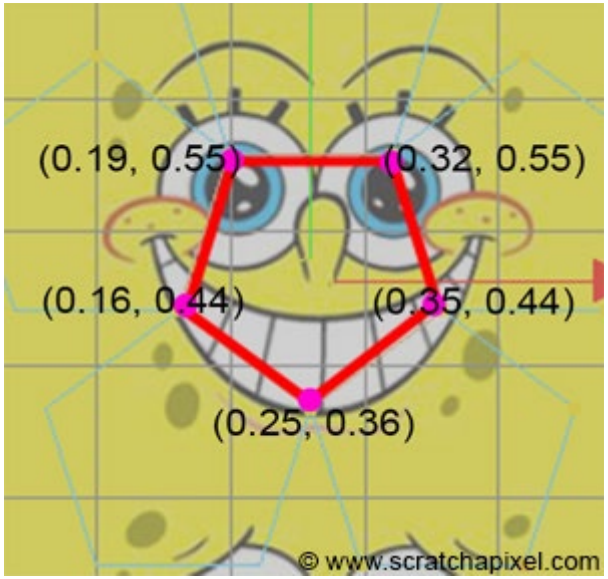
- O que armazenar?
  - Geometria + topologia
  - Outras informações associadas aos vértices, necessárias para o *rendering*
    - Normais às faces/vértices
    - Coordenadas de textura
    - Material (cor, etc.)

Normais



Textura


V
U


(0.19, 0.55)
(0.32, 0.55)
(0.16, 0.44)
(0.35, 0.44)
(0.25, 0.36)
© www.scratchapixel.com

Informações Geométricas

- As coordenadas dos vértices dão toda a informação geométrica necessária para o *rendering* do modelo
  - A partir delas pode-se computar a inclinação das arestas, o retângulo envoltório (*bounding box*) de cada face, a equação do plano que contém cada face e sua normal, etc.
  - Informações necessárias para algoritmos de recorte, remoção de superfícies ocultas e de *rendering*...

Informações Geométricas

- Equação do plano:

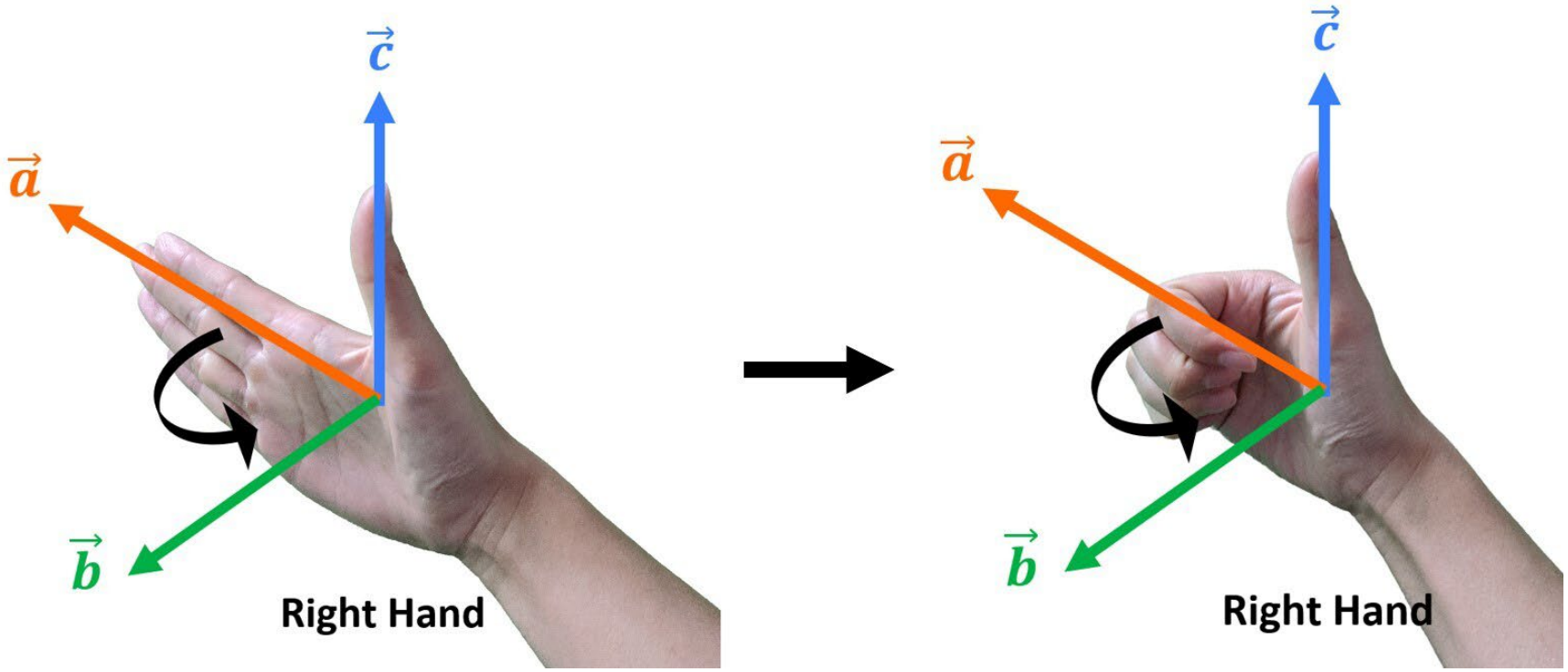
$$Ax + By + Cz + D = 0$$

em que (x,y,z) é um ponto qqr no plano, A, B, C e D são constantes.

Como obter a equação do plano que contém 3 pontos dados (não colineares)?

Veja, p.ex. Hearn & Baker section 3-15

Regra da mão direita



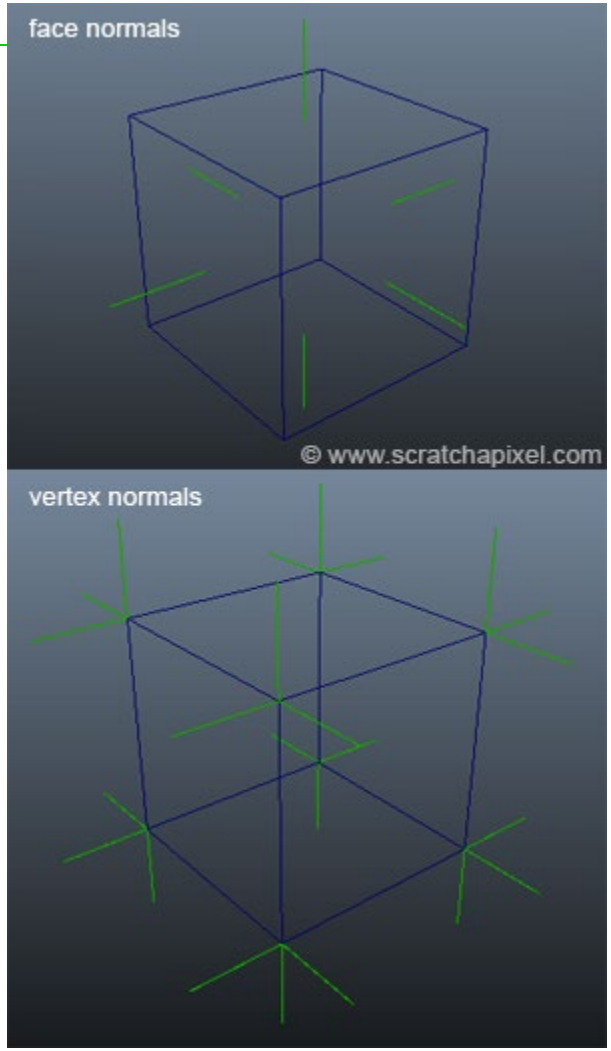
$$\mathbf{a} \times \mathbf{b} = \mathbf{c}$$

Informações Geométricas

- Convenção em CG: a direção positiva da normal ao plano indica a sua face potencialmente visível
- Importante calcular a normal de maneira consistente para diferenciar entre os 2 lados: face 'visível' e face 'não visível'

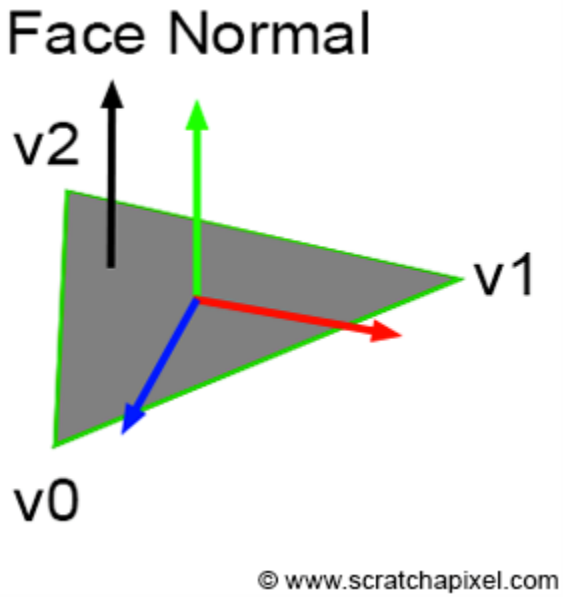
Normais

Convenção: normais às faces/vértices sempre apontam para o lado externo ao objeto



Orientação das normais

- Orientação das normais às faces indica a face visível ao observador
- Vértices da face são especificados no sentido anti-horário de alguém observando o plano do lado da face visível
- Regra da mão direita



Consistent Normal Orientation for Polygonal Meshes

Pavel Borodin Gabriel Zachmann Reinhard Klein

Institute of Computer Science II
University of Bonn
Römerstraße 164, 53175 Bonn, Germany
{borodin, zach, rk}@cs.uni-bonn.de

Abstract

In this paper, we propose a new method that can consistently orient all normals of any mesh (if at all possible), while ensuring that most polygons are seen with their front-faces from most viewpoints. Our algorithm combines the proximity-based *with a new* visibility-based *approach. Thus, it virtually eliminates the problems of proximity-based approaches, while avoiding the limitations of previous solid-based approaches.*

Our new method builds a connectivity graph of the patches of the model, which encodes the "proximity" of neighboring patches. In addition, it augments this graph with two visibility coefficients for each patch. Based on this graph, a global consistent orientation of all patches is quickly found by a greedy optimization.

We have tested our new method with a large suite of models, many of which from the automotive industry. The results

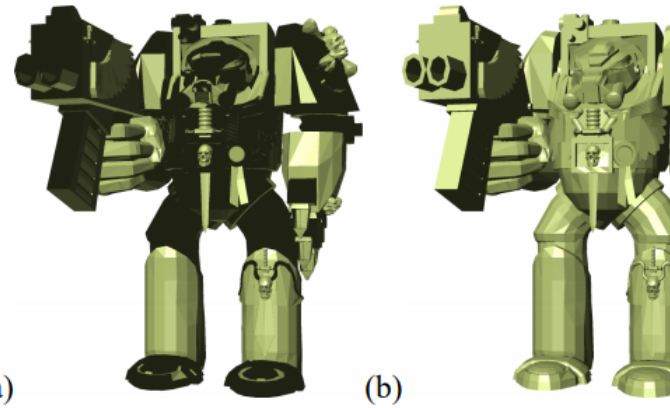


Figure 1. (a) A model consisting of 2398 sep rate surface patches with inconsistent orie tation; due to the lighting, the faces orient "backwards" are rendered dark. (b) Sa model after applying our algorithm (with actly the same lighting).

Exemplo – formato OBJ

- Lista das coordenadas dos vértices
- Lista das normais às faces (nos vértices)
- Lista das coordenadas de textura
- Lista das Faces: indexa tabela de vértices/textura/normal

Exemplo – formato OBJ

- P. ex., cubo

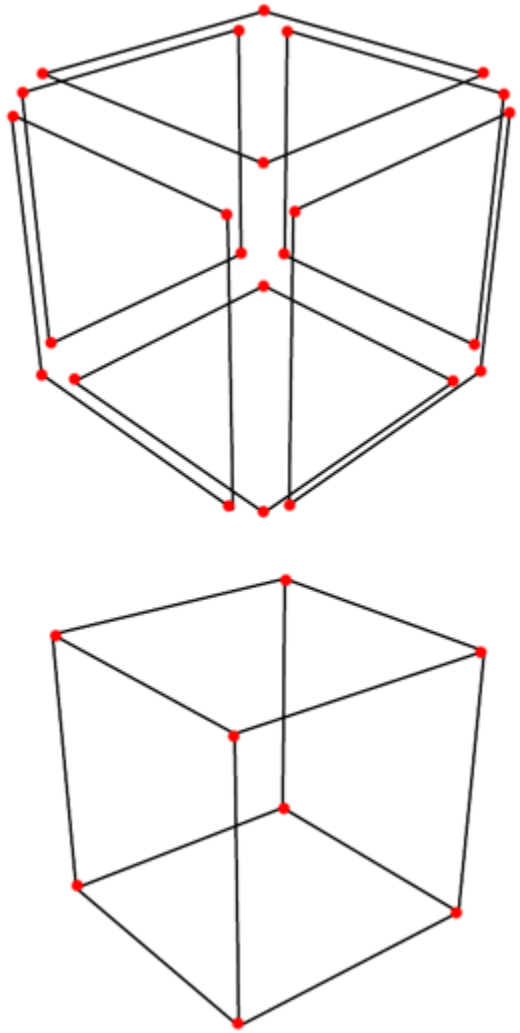
de (-1,-1,-1) a (1,1,1)

8 vértices, 12 faces

(cada face subdividida em 2

2 triângulos)

- formato OBJ não usa o elemento topológico 'aresta'



Exemplo – formato OBJ

- https://klebermota.eti.br/2023/04/18/definindo-um-objeto-3d-com-o-formato-de-arquivo-obj/

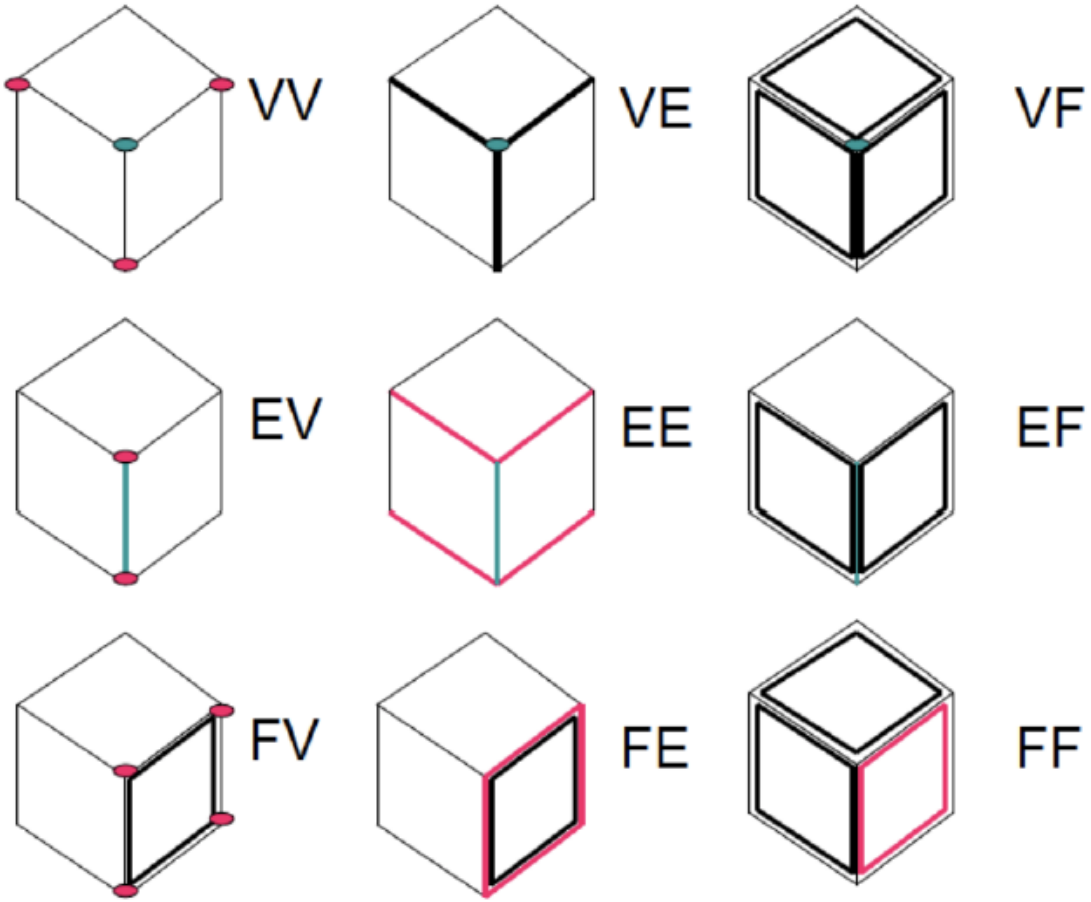
Mais informações

- http://immersivemath.com/ila/ch04_vectorproduct/ch04.html
- Introduction to Polygon Meshes (scratchapixel.com)

Estruturas de Dados

- Outras representações
  - Estruturas de dados topológicas
    - Mantém informação topológica (relações de adjacência)
    - Grafos
  - Tabelas de faces, de arestas e de vértices
    - Todos os 'elementos topológicos' (faces, arestas e vértices) são representados explicitamente...
  - Cada face faz referência às arestas que a compõem, cada aresta faz referência aos vértices que a definem

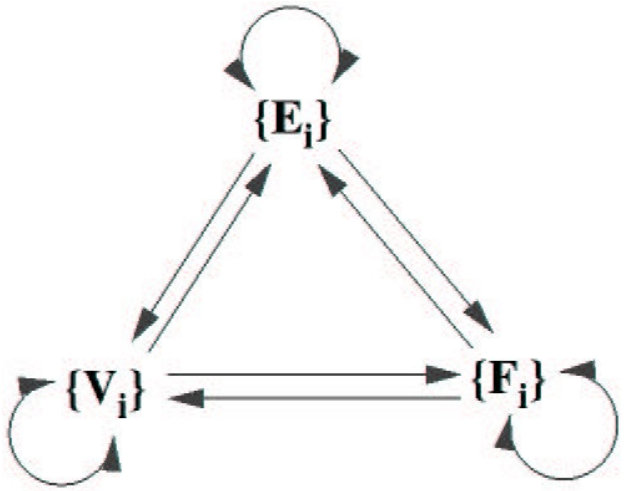
Estruturas de Dados



Exemplos de relações de adjacências (consultas topológicas)

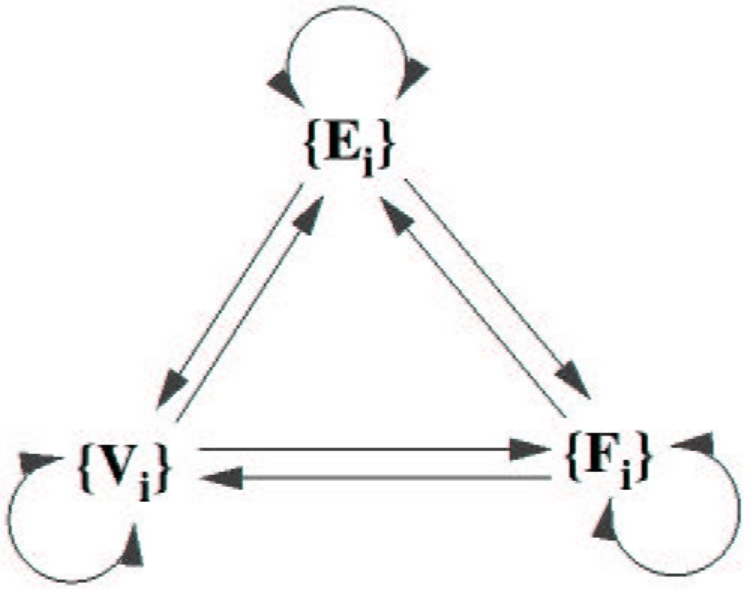
Estruturas de Dados

- Listas de adjacências
  - Armazena explicitamente todas as adjacências entre vértices, faces e arestas
  - Toda a informação topológica explícita
  - Custo extra de armazenagem



Estruturas de Dados

- Listas parciais de adjacências
  - Possível armazenar algumas relações de adjacência, e inferir outras?

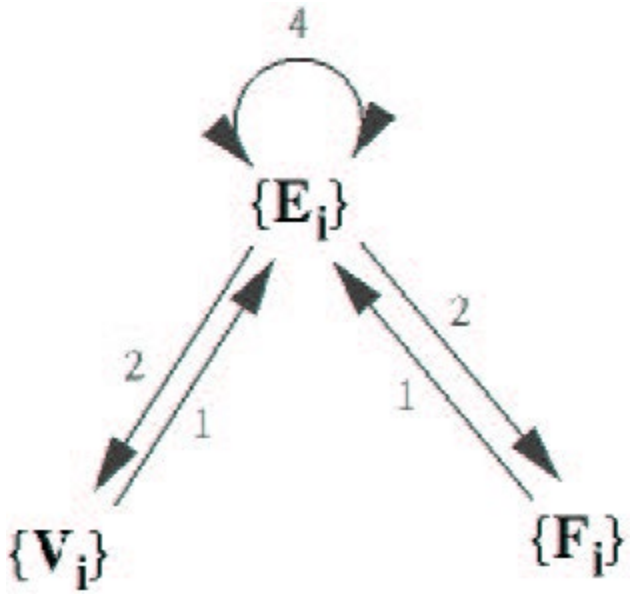


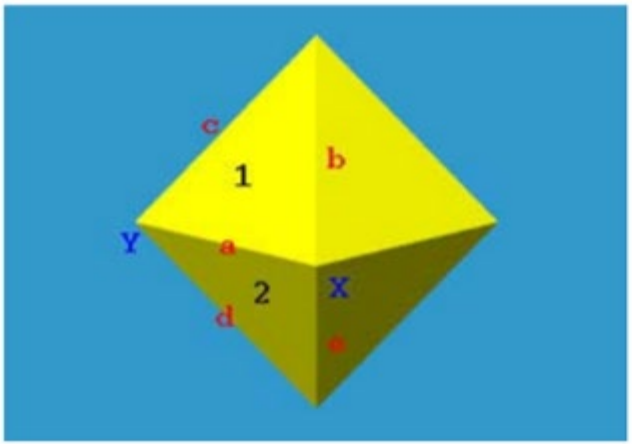
Estruturas de Dados

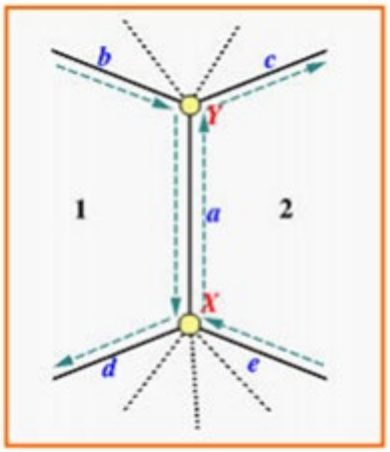
- Winged-edge (Baungart, 1972)
  - Associa informações de adjacência às arestas
    - Pode recuperar as adjacências entre arestas e faces, ou arestas e vértices, em tempo O(1)
    - Outras adjacências requerem busca na estrutura
    - Custo extra de armazenagem pequeno (registros de tamanho fixo)
    - Consegue representar objetos cujas faces são polígonos arbitrários

Estruturas de Dados

- Winged-edge – objetos `fechados´, toda aresta é compartilhada por 2 faces

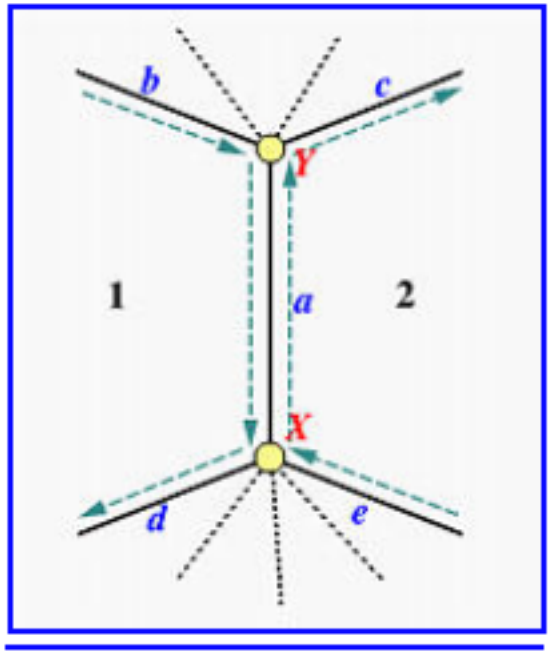






Estruturas de Dados

- Winged-edge – exemplo

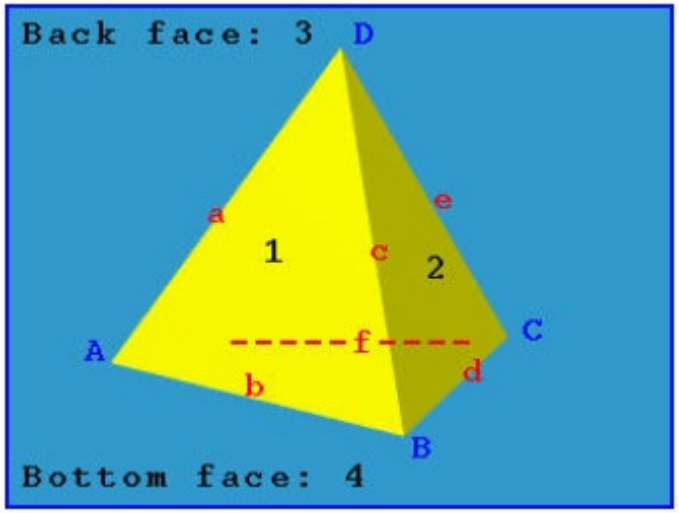


Edge	Vertices		Faces		Left Traverse		Right Traverse	
Name	Start	End	Left	Right	Pred	Succ	Pred	Succ
a	X	Y	1	2	b	d	e	c

- Fonte: http://www.cs.mtu.edu/~shene/COURSES/cs3621/NOTES/model/winged-e.html

Estruturas de Dados

- Winged-edge – fazer



Resposta parcial (completar):

Edge	Vertices		Faces		Left Traverse		Right Traverse	
Name	Start	End	Left	Right	Pred	Succ	Pred	Succ
a	A	D						
b	A	B						
c	B	D						
d	B	C						
e	C	D						
f	A	C						

Estruturas de Dados

- Winged-edge – resposta

Edge	*Vertices*		*Faces*		*Left Traverse*		*Right Traverse*	
Name	*Start*	*End*	*Left*	*Right*	*Pred*	*Succ*	*Pred*	*Succ*
a	A	D	3	1	e	f	b	c
b	A	B	1	4	c	a	f	d
c	B	D	1	2	a	b	d	e
d	B	C	2	4	e	c	b	f
e	C	D	2	3	c	d	f	a
f	A	C	4	3	d	b	a	e

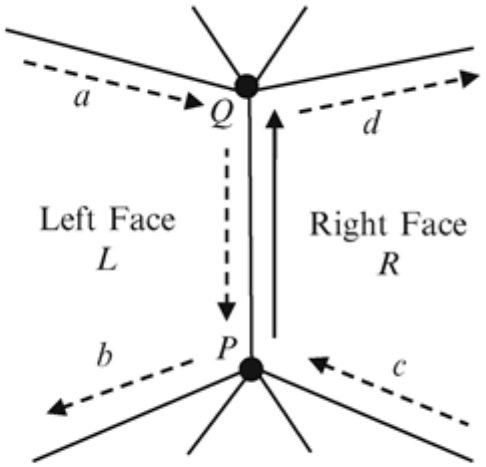
- Adicionalmente:

Vertex Name	*Incident Edge*
A	a
B	b
C	d
D	e

Face Name	*Incident Edge*
1	a
2	c
3	a
4	b

Winged-edge

```
struct W_edge
{
   Vertex *start, *end;
   Face   *left,  *right;
   W_edge *left_prev,  *left_next;
   W_edge *right_prev, *right_next;
};
struct Vertex
{
    float x, y, z;
    W_edge *edge;
};
struct Face
{
    W_edge *edge;
};
```



Winged-edge - exercícios

fonte: http://what-when-how.com/advanced-methods-in-computer-graphics/mesh-processing-advanced-methods-in-computer-graphics-part-2/

Desenvolver o pseudo-código de uma função, usando a estrutura dada, para determinar todas as arestas incidentes a um vértice.

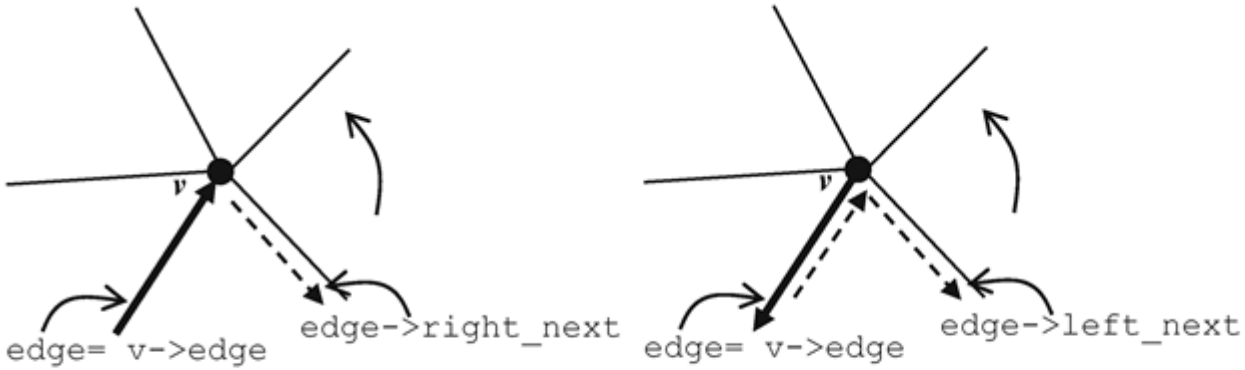


Fig. 8.11 Computation of all edges incident at a vertex. Both directions of an edge should be considered in algorithms using the winged-edge data structure

Winged-edge

Resposta:

```
1.   Input:  v                                   //A vertex
2.   W_edge *e0 = v->edge;                       //Initial edge
3.   W_edge *edge = e0;
4.   do
5.   {
6.      if(edge->end == v) edge = edge -> right_next;
7.                         else  edge = edge -> left_next;
8.      output(edge);
9.   } while (edge != e0);
```

Winged-edge

Desenvolver o pseudo-código de uma função, usando a estrutura dada, para determinar todas as arestas que formam uma face

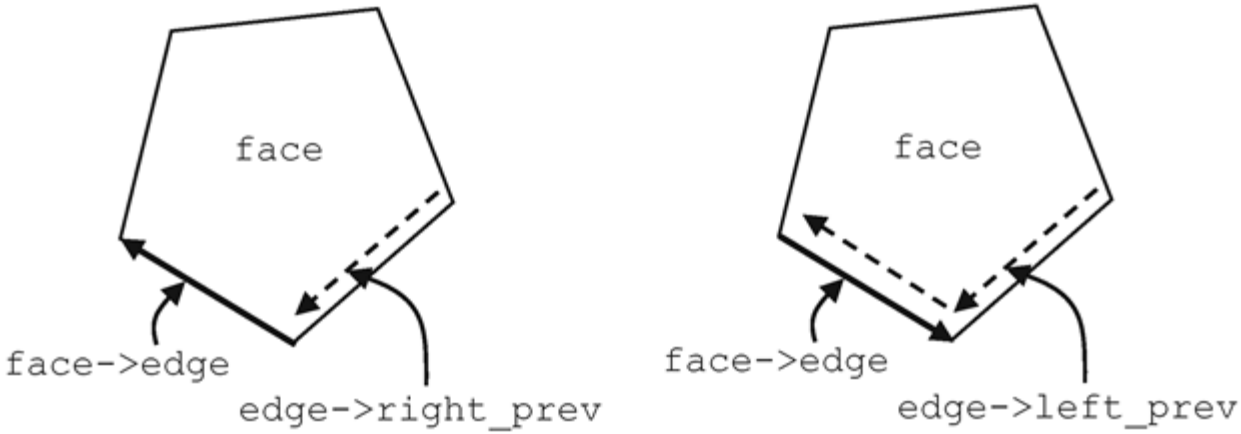


Fig. 8.12 Computation of edges around a polygonal face

Winged-edge

Resposta:

```
1.  Input:  face
2.  W_edge *e0 = face->edge;            //Initial edge
3.  W_edge *edge = e0;
4.  do
5.  {
6.     if(edge->right == face) edge = edge->right_prev;
7.                             edge = edge->left_prev;
8.     output(edge);
9.  } while (edge != e0);
```

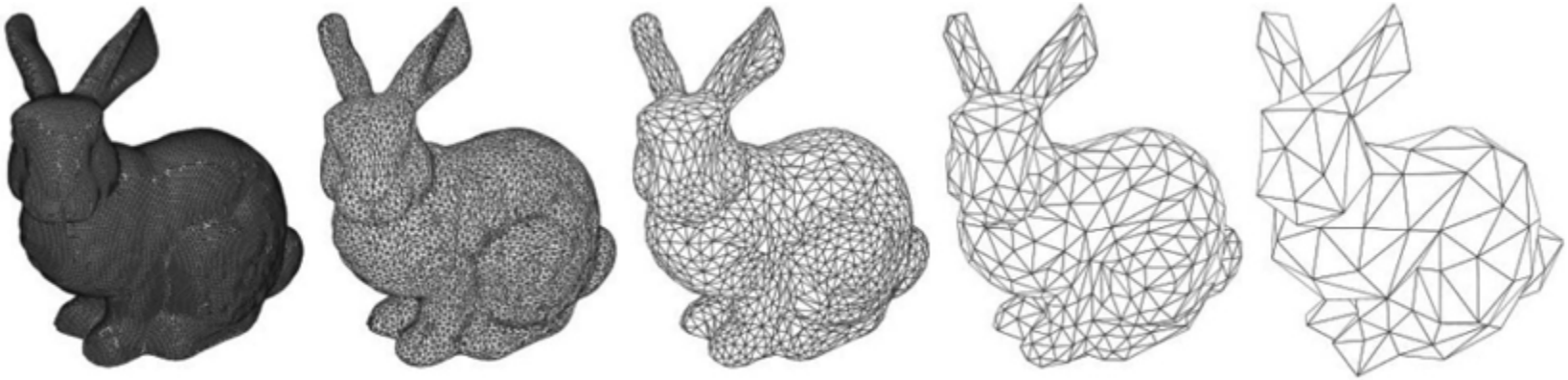
Estruturas de Dados

- Observação
  - Muitas E.D. podem apresentar inconsistências se o processo de geração da malha não for muito cuidadoso para evitá-las
  - a malha pode descrever objetos não 'factíveis'
    - Ex. vértices e arestas isolados, polígonos não fechados, polígonos isolados, etc.
- Certas estruturas, como a *winged-edge*, não admitem tais ocorrências: o modelo descreve objeto de maneira consistente

Malhas de Triângulos

- Um problema bastante relevante é o da simplificação de malhas (*decimação*)
  - Redução do número de polígonos/triângulos necessários para descrever um modelo
  - Porquê?
    - *Rendering* mais rápido
    - Menor custo de armazenagem
    - Manipulação mais simples

- https://techblog.floorplanner.com/2015/09/21/mesh-decimation/



https://github.com/neurolabusc/Fast-Quadric-Mesh-Simplification-Pascal-

Bibliografia

- Hearn & Baker Computer Graphics with OpenGL