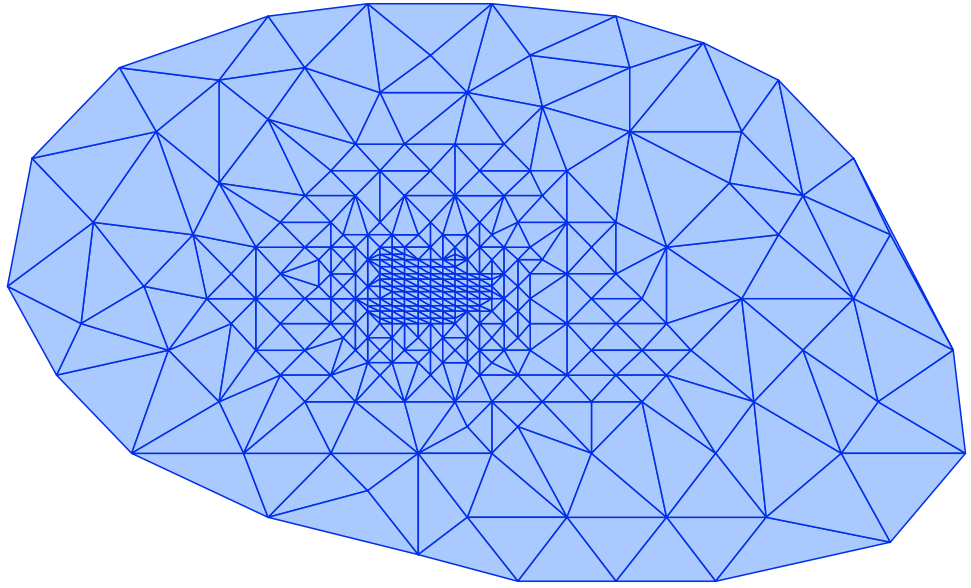


Modelagem Geométrica – SME0271

Estruturas de Dados: iVET

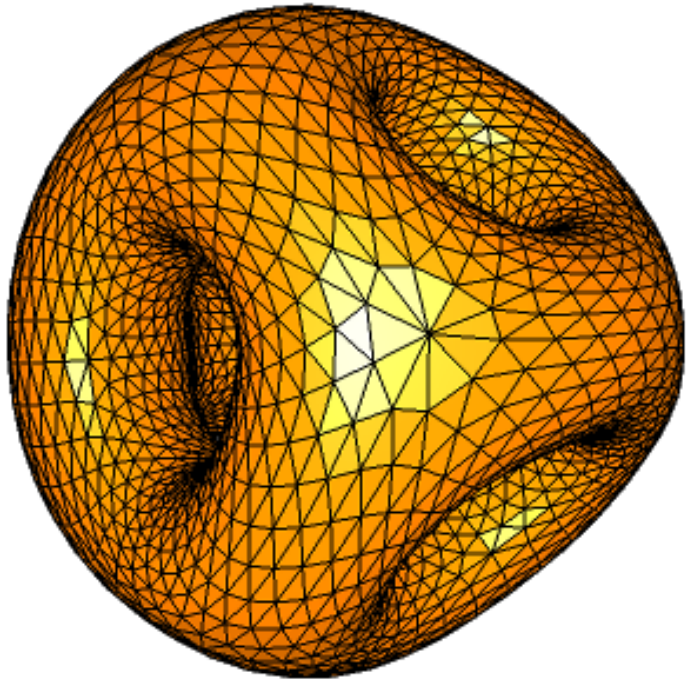
Luiz Otávio Toratti
ICMC-USP

19 de agosto de 2016

Estruturas de Dados Topológicas

Problema

Como codificar a estrutura geométrica e topológica (vizinhanças) de uma superfície?



Estruturas de Dados Topológicas

A representação da superfície pode ser vista como um banco de dados geométrico com consultas e operações:

Estruturas de Dados Topológicas

A representação da superfície pode ser vista como um banco de dados geométrico com consultas e operações:

- Consultas geométricas
  - Quais são os vértices da face k?
  - Os vértices i e j são adjacentes?

Estruturas de Dados Topológicas

A representação da superfície pode ser vista como um banco de dados geométrico com consultas e operações:

- Consultas geométricas
  - Quais são os vértices da face k?
  - Os vértices i e j são adjacentes?
- Operações geométricas
  - Adicionar/remover vértice;
  - Adicionar/remover face.

Estruturas de Dados Topológicas

A representação da superfície pode ser vista como um banco de dados geométrico com consultas e operações:

- Consultas geométricas
  - Quais são os vértices da face k?
  - Os vértices i e j são adjacentes?
- Operações geométricas
  - Adicionar/remover vértice;
  - Adicionar/remover face.
- Quão "boa" é uma estrutura de dados?
  - Tempo de construção
  - Tempo de responder uma consulta
  - Tempo de efetuar uma operação (atualização da estrutura)
  - Redundância

Lista de Faces

Dada uma malha de N vértices e M triângulos:

Vértices ($N \times 3$) Triângulos ($M \times 3$)

$$\begin{bmatrix} v_1 = (x_1, y_1, z_1) \\ v_2 = (x_2, y_2, z_2) \\ \vdots \\ v_i = (x_i, y_i, z_i) \\ \vdots \\ v_N = (x_N, y_N, z_N) \end{bmatrix} \quad \begin{bmatrix} f_1 = (v_a, v_b, v_c) \\ f_2 = (v_b, v_z, v_p) \\ \vdots \\ f_i = (v_k, v_q, v_m) \\ \vdots \\ f_M = (v_d, v_k, v_w) \end{bmatrix}$$

Lista de Faces

- Lista de vértices (coordenadas)
- Lista de triângulos (tripla de índices)
- Consultas:
  - Quais são os vértices da face k?

Lista de Faces

- Lista de vértices (coordenadas)
- Lista de triângulos (tripla de índices)
- Consultas:
  - Quais são os vértices da face k?
    Resposta em $O(1)$

Lista de Faces

- Lista de vértices (coordenadas)
- Lista de triângulos (tripla de índices)
- Consultas:
  - Quais são os vértices da face k?
    Resposta em $O(1)$
  - Os vértices i e j são adjacentes?

Lista de Faces

- Lista de vértices (coordenadas)
- Lista de triângulos (tripla de índices)
- Consultas:
  - Quais são os vértices da face k?
    Resposta em $O(1)$
  - Os vértices i e j são adjacentes?
    Percorrer as faces

Lista de Faces

- Lista de vértices (coordenadas)
- Lista de triângulos (tripla de índices)
- Consultas:
  - Quais são os vértices da face k?
    Resposta em $O(1)$
  - Os vértices i e j são adjacentes?
    Percorrer as faces
- Vantagem
  - Baixo consumo de memória

Lista de Faces

- Lista de vértices (coordenadas)
- Lista de triângulos (tripla de índices)
- Consultas:
  - Quais são os vértices da face k?
    Resposta em $O(1)$
  - Os vértices i e j são adjacentes?
    Percorrer as faces
- Vantagem
  - Baixo consumo de memória
- Desvantagem
  - Não possui informações "suficientes"

iVET: índice, vértice, estrela, triângulo

Desenvolvida por Felipe Montefuscolo em 2014.

iVET: índice, vértice, estrela, triângulo

Desenvolvida por Felipe Montefuscolo em 2014.

Dada uma malha de N vértices e M triângulos:

Vértices ($N \times 3$) Estrelas ($N \times S_i$) Triângulos ($M \times 3$)

$$\begin{bmatrix} \text{coord. } v_1 \\ \text{coord. } v_2 \\ \vdots \\ \text{coord. } v_i \\ \vdots \\ \text{coord. } v_N \end{bmatrix} \quad \begin{bmatrix} \text{estrela } v_1 \\ \text{estrela } v_2 \\ \vdots \\ \text{estrela } v_i \\ \vdots \\ \text{estrela } v_N \end{bmatrix} \quad \begin{bmatrix} \text{vérts. de } t_1 \\ \text{vérts. de } t_2 \\ \vdots \\ \text{vérts. de } t_j \\ \vdots \\ \text{vérts. de } t_M \end{bmatrix}$$

iVET: índice, vértice, estrela, triângulo

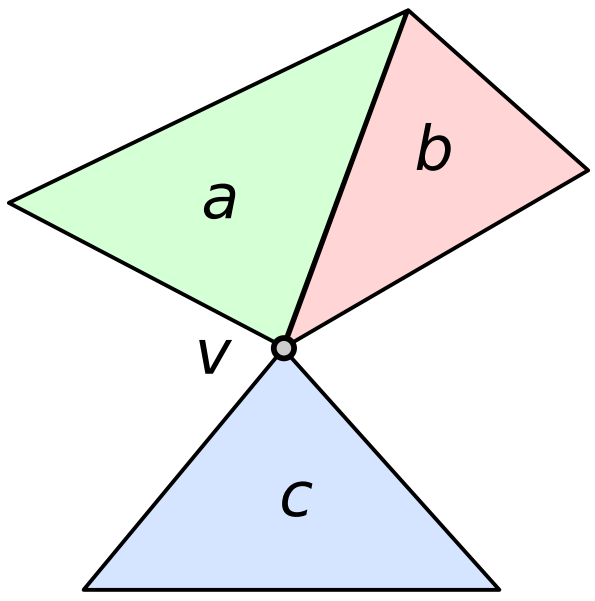
Definição

A estrela de um vértice v é o conjunto de todos os triângulos que contém v.

iVET: índice, vértice, estrela, triângulo

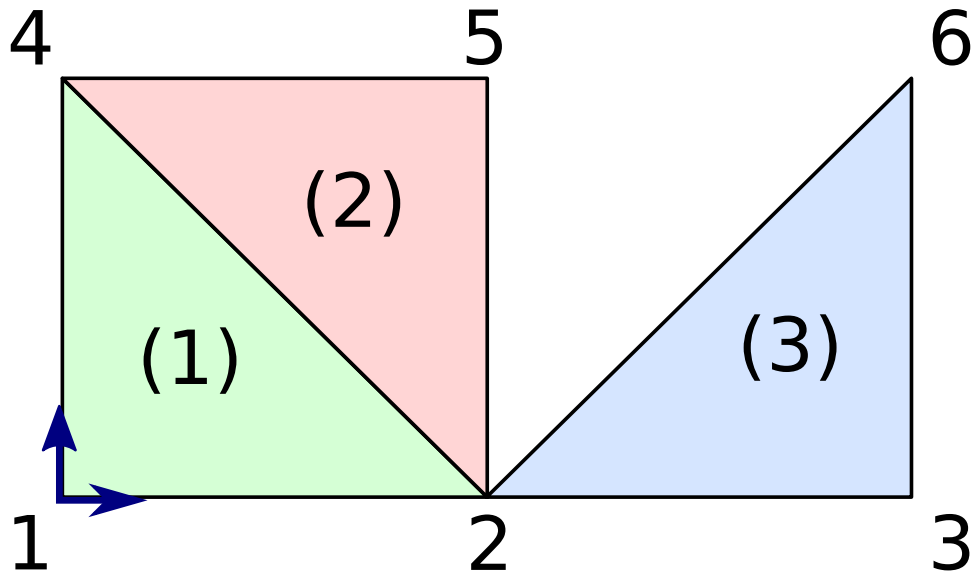
Definição

A estrela de um vértice v é o conjunto de todos os triângulos que contém v.

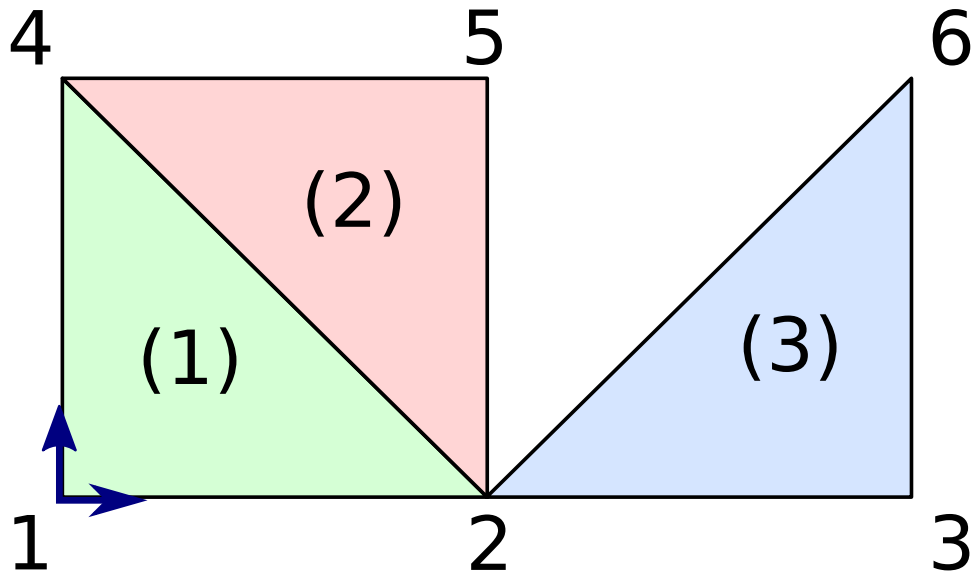


No exemplo acima, a estrela de v é o conjunto $\{a,b,c\}$.

Exemplo



Exemplo



$$\mathrm{V} = \begin{bmatrix} 0. & 0. & 0. \\ 1. & 0. & 0. \\ 2. & 0. & 0. \\ 0. & 1. & 0. \\ 1. & 1. & 0. \\ 2. & 1. & 0. \end{bmatrix} \quad \mathrm{E} = \begin{bmatrix} (1) \\ (1),(2),(3) \\ (3) \\ (1),(2) \\ (2) \\ (3) \end{bmatrix} \quad \mathrm{T} = \begin{bmatrix} 1 & 2 & 4 \\ 2 & 5 & 4 \\ 2 & 3 & 6 \end{bmatrix}$$

Representação das tabelas no MATLAB

As tabelas do exemplo anterior no MATLAB ficam:

```
V = [0., 0., 0.; ...
     1., 0., 0.; ...
     2., 0., 0.; ...
     0., 1., 0.; ...
     1., 1., 0.; ...
     2., 1., 0.];
```

```
E = { [1], ...
      [1,2,3], ...
      [3], ...
      [1,2], ...
      [2], ...
      [3]};
```

Observação:

- A forma de T é análoga a V (ou seja, também é uma matriz);

[1] http://www.mathworks.com/help/matlab/cell-arrays.html

Representação das tabelas no MATLAB

As tabelas do exemplo anterior no MATLAB ficam:

```
V = [0., 0., 0.; ...
     1., 0., 0.; ...
     2., 0., 0.; ...
     0., 1., 0.; ...
     1., 1., 0.; ...
     2., 1., 0.];
```

```
E = { [1], ...
      [1,2,3], ...
      [3], ...
      [1,2], ...
      [2], ...
      [3]};
```

Observação:

- A forma de T é análoga a V (ou seja, também é uma matriz);
- V é uma matriz, ao passo que E é uma lista de vetores;

[1] http://www.mathworks.com/help/matlab/cell-arrays.html

Representação das tabelas no MATLAB

As tabelas do exemplo anterior no MATLAB ficam:

```
V = [0., 0., 0.; ...
     1., 0., 0.; ...
     2., 0., 0.; ...
     0., 1., 0.; ...
     1., 1., 0.; ...
     2., 1., 0.];
```

```
E = { [1], ...
      [1,2,3], ...
      [3], ...
      [1,2], ...
      [2], ...
      [3]};
```

Observação:

- A forma de T é análoga a V (ou seja, também é uma matriz);
- V é uma matriz, ao passo que E é uma lista de vetores;
  - Cell Array[1]

[1] http://www.mathworks.com/help/matlab/cell-arrays.html

Acessando os elementos das tabelas

- Para acessar a componente j da coordenada do vértice i, faça:

```
V(i,j)
```

- Para obter um vetor com componentes da coordenada do vértice i, faça:

```
V(i,:)
```

- Para acessar o elemento j da estrela do vértice i, faça:

```
E{i}(j)
```

- Para obter um vetor com a estrela do vértice i, faça:

```
E{i}
```

Operações na tabela de coordenadas

- Adicionando a coordenada $[x, y, z]$ ao final da matriz V:
  - Alternativa 1:

```
V = [V; [x, y, z]];
```

  - Alternativa 2:

```
V = vertcat(V, [x, y, z]);
```

  - Alternativa 3:

```
V(end+1,:) = [x, y, z];
```

- Removendo a i-ésima coordenada da matriz V:

```
V(i,:) = [];
```

Operações na lista de estrelas

- Adicionando um vértice no final da lista E:

```
E{end+1} = []; % adiciona um vetor vazio!
```

- Removendo o i-ésimo vértice de E:

```
E(i) = [];
```

- adicionando um triângulo t à estrela do vértice i:

```
E{i} = [E{i}, t];
```

Opa, mas e se t já estava em E{i}? Faça:

```
E{i} = unique( [E{i}, t] );
```

A função `unique` remove índices repetidos.

Operações básica implementadas

Quatro operações básicas já estão implementadas: inserção de vértices, inserção de triângulos, remoção de triângulos e coletor de lixo.

- **op_insere_vtc.m:** insere um vértice ou mais (modifica V).

Operações básica implementadas

Quatro operações básicas já estão implementadas: inserção de vértices, inserção de triângulos, remoção de triângulos e coletor de lixo.

- **op_insere_vtc.m:** insere um vértice ou mais (modifica V).
- **op_insere_tri.m:** insere um triângulo ou mais (modifica E e T).

Operações básica implementadas

Quatro operações básicas já estão implementadas: inserção de vértices, inserção de triângulos, remoção de triângulos e coletor de lixo.

- **op_insere_vtc.m:** insere um vértice ou mais (modifica V).
- **op_insere_tri.m:** insere um triângulo ou mais (modifica E e T).
- **op_remove_tri.m:** marca um triângulo ou mais (modifica E e T).

Operações básica implementadas

Quatro operações básicas já estão implementadas: inserção de vértices, inserção de triângulos, remoção de triângulos e coletor de lixo.

- **op_insere_vtc.m:** insere um vértice ou mais (modifica V).
- **op_insere_tri.m:** insere um triângulo ou mais (modifica E e T).
- **op_remove_tri.m:** marca um triângulo ou mais (modifica E e T).
- **op_limpa_lixo.m:** remove os triângulos marcados e os vértices não referenciados. (modifica V, E e T).

Como usar

As funções op_* recebem as tabelas V, E e T como argumento e as retornam atualizadas.

Exemplo de uso:

```
[V,E] = op_insere_vtc(V,E, [0,0,0]);
```

Como usar

As funções op_* recebem as tabelas V, E e T como argumento e as retornam atualizadas.

Exemplo de uso:

```
[V,E] = op_insere_vtc(V,E, [0,0,0]);
```

Antes de construir a malha, as tabelas devem ser inicializadas da seguinte forma:

```
% inicializando corretamente
V = zeros(0,3);   % matriz vazia
E{1} = [];
T = zeros(0,3);
```

Operações: inserir vértice

```
function [V,E] = op_insere_vtc(V,E,X)

  if isempty(V)
    V = X;
    else
    V = vertcat(V,X);
  end

  E{size(V,1)} = [];

end
```

Operações: inserir triângulo

```
function [E,T] = op_insere_tri(E,T, new_trigs)

Nt_old = size(T,1);
N_adds = size(new_trigs,1);

if isempty(T)
  T = new_trigs;
else
  T = vertcat(T,new_trigs);
end

for n=1:N_adds
  c_idx = Nt_old+n;
  E{new_trigs(n,1)} = unique([E{new_trigs(n,1)}, c_idx]);
  E{new_trigs(n,2)} = unique([E{new_trigs(n,2)}, c_idx]);
  E{new_trigs(n,3)} = unique([E{new_trigs(n,3)}, c_idx]);
end

end
```

Operações: remover triângulo

```
function [E,T] = op_remove_tri(E,T, trigs)

for kk=1:length(trigs)
  t = trigs(kk);

  vtcs = T(t,:);
  E{vtcs(1)} = E{vtcs(1)} ( E{vtcs(1)}~= t );
  E{vtcs(2)} = E{vtcs(2)} ( E{vtcs(2)}~= t );
  E{vtcs(3)} = E{vtcs(3)} ( E{vtcs(3)}~= t );

  T(t,:) = [0,0,0];
end

end
```

Operações: limpar lixo (atualiza a estrutura)

```
function [V,E,T] = op_limpa_lixo(V,E,T)

Nt_old = size(T,1);
Nv_old = size(V,1);

for i=Nt_old:-1:1
  if T(i,1) == 0
    T(i,:) = [];
    for k=1:Nv_old
      E{k}(E{k}>i) = E{k}(E{k}>i) - 1;
    end
  end
end

for i=Nv_old:-1:1
  if isempty(E{i})
    E(i) = [];
    V(i,:) = [];
    T(T>i) = T(T>i) - 1;
  end
end
```

Como usar

Um típico programa deve seguir o seguinte roteiro:

1. Inicializar as tabelas V, E e T;
2. Adicionar vértices com a função op_insere_vtc;
3. Adicionar triângulos com a função op_insere_tri;
4. Fazer as operações necessário com op_insere_vtc, op_insere_tri e op_limpa_lixo;
5. Chamar o coletor de lixo op_limpa_lixo.