

Variedades Computacionais

Capítulo 4

André Fakhoury e Pedro Rocha



Sumário I

- ① Modelagem geométrica
- ② Definições
 - Molas e malhas dinâmicas
- ③ Aplicações
 - Representação eficiente de formas
 - Edição de malhas e interpolação
- ④ Referências



Modelagem geométrica

Conjunto de técnicas e algoritmos utilizados para modelar determinadas formas matemáticas, sujeitas a condições particulares de forma e suavidade.

Utilizado em especial em CAD/CAM (*computer aided design / manufacturing*), por seu alto poder em modelagem de superfícies.

Uma forma possível de se modelar é com a utilização da **discretização do operador de Laplace-Beltrami**.



Métodos para modelagem

Com a utilização da discretização do operador de Laplace-Beltrami, existem métodos:

- baseados em malha (*mesh-based*)
- baseados em ponto (*point-based*)



Métodos para modelagem

Com a utilização da discretização do operador de Laplace-Beltrami, existem métodos:

- baseados em malha (*mesh-based*) (foco da apresentação)
- baseados em ponto (*point-based*)



Malhas Poligonais

Uma Malha Poligonal é um conjunto de Vértices, Arestas e Faces que formam um conjunto conexo. A malha poligonal pode ser categorizada pela forma de suas faces, como malhas triangulares, quadrilaterais, ou, de fato, que utilizam qualquer polígono convexo simples. Elas também podem ser bidimensionais ou tridimensionais.

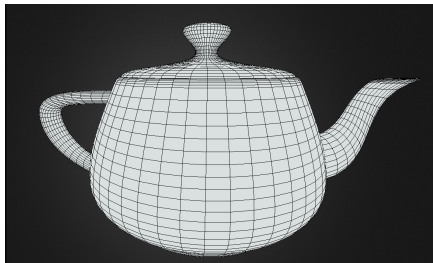


Figura 1: Malha Quadrilateral Tridimensional do Bule de Utah [New75]



Coordenadas diferenciais

Seja uma malha triangular $\mathcal{M} = (V, E, F)$. Cada vértice $\mathbf{v}_i \in V$ possui uma representação cartesiana dada por $\mathbf{v}_i = (x_i, y_i, z_i)$.

Coordenadas diferenciais (ou δ -coordenadas) de \mathbf{v}_i são definidas como a diferença entre a coordenada cartesiana e o centro de massa de seus vizinhos imediatos na malha:

$$\delta_i = (\delta_i^{(x)}, \delta_i^{(y)}, \delta_i^{(z)}) = \mathbf{v}_i - \frac{1}{d_i} \sum_{j \in N(i)} \mathbf{v}_j, \quad (2.1)$$

em que $N(i) = \{j | (i, j) \in E\}$ e $d_i = |N(i)|$.



δ -coordenadas ponderadas

Adicionando pesos às arestas E , é possível alterar o relacionamento entre um vértice e cada um de seus vizinhos.

Assim, as alterações a um vértice irão se propagar a cada um dos vizinhos deste de forma proporcional ao peso da aresta entre eles.

$$\delta_i = c_i \sum_{j \in N(i)} w_{ij}(v_i - v_j)$$



Ponderação padrão: média dos vizinhos

$$\delta_i = \frac{1}{d_i} \sum_{j \in N(i)} (v_i - v_j)$$

O peso de todas as arestas é constante e \mathbf{c}_i depende unicamente do número de vizinhos de \mathbf{v}_i . Desta forma, alterações sofridas por um vértice são distribuídas uniformemente entre os vértices vizinhos a ele.



Pesos cotangentes

Recomendáveis para malhas muito irregulares, gera apenas componentes normais.

$$\delta_i^c = \frac{1}{|\Omega|} \sum_{j \in N(i)} \frac{1}{2} (\cot \alpha_{ij} + \cot \beta_{ij}) (\mathbf{v}_i - \mathbf{v}_j)$$

em que $|\Omega|$ é o tamanho da célula de Voronoi de i e α_{ij}, β_{ij} são os dois ângulos opostos à aresta (i, j) .



Coordenadas diferenciais

Código em Octave (geração das δ -coordenadas)

```
function [ delta ] = geraDeltaCoords(v, w, c)
% geraDeltaCoords: Gera as delta-coordenadas
%   Gera as coords dif. a partir das absolutas e pesos w e c
% Entrada
%   v: vetor contendo as n (x, y, z) coordenadas;
%   w, c: pesos de cada adjacência w(i, j) e de cada vértice
%       delta_i = c_i * sum_j (w_ij * (v_i - v_j))
[n, x] = size(v); % número de [vertices, dimensão]
delta = zeros(n, x);
for i=1:n
    for j=1:n
        delta(i,:) = delta(i,:) + w(i,j) * (v(i,:) - v(j,:));
    end
    delta(i,:) = c(i) * delta(i,:);
end
end
```

Transformação para δ -coordenadas

Matriz transformação para δ -coordenadas

Seja A a matriz de adjacências da malha:

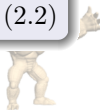
$$A_{ij} = \begin{cases} 1 & (i, j) \in E \\ 0 & \text{caso contrário.} \end{cases}$$

e D a matriz diagonal tal que

$$D_{ii} = d_i = |N(i)| = \text{número de vértices adjacentes a } i$$

A matriz L de transformação de coordenadas cartesianas para as coordenadas diferenciais é:

$$L = I - D^{-1}A. \quad (2.2)$$



Transformação para δ -coordenadas

Matriz simétrica de transformação

É mais comum utilizar a versão simétrica de L , L_s , tal que:

$$L_s = DL = D - A$$

e cada célula pode ser calculada por:

$$(L_s)_{ij} = \begin{cases} d_i & i = j \\ -1 & (i, j) \in E \\ 0 & \text{caso contrário.} \end{cases} \quad (2.3)$$



Transformação para δ -coordenadas

Matriz simétrica de transformação

Temos:

$$L_s \mathbf{x} = \delta^{(x)}$$

$$L_s \mathbf{y} = \delta^{(y)}$$

$$L_s \mathbf{z} = \delta^{(z)}$$

e L_s é denominado o **Laplaciano topológico** da malha \mathcal{M} .



Matriz laplaciana

Código em Octave (geração do Laplaciano topológico L_s)

```
function [ Ls ] = geraLaplaciano(adj)
% geraLaplaciano Gera a matriz simetrica do laplaciano
%   Gera a matriz simétrica do Laplaciano Topológico
% Entrada:
%   adj: matriz de adjacências da malha original
n = size(adj, 1); % número de vértices
D = diag(sum(adj));
Ls = D - adj; % Laplaciano simétrico L_s
end
```



Discretização de Laplace-Beltrami

Discretização de Laplace-Beltrami

Se considerar \mathcal{M} uma aproximação linear por partes de uma superfície suave,

$$\delta_i = \frac{1}{d_i} \sum_{j \in N(i)} (\mathbf{v}_i - \mathbf{v}_j)$$

é uma discretização de

$$\frac{1}{|\gamma|} \int_{v \in \gamma} (v_i - v) dl(v)$$

em que γ é uma curva de uma superfície fechada simples em volta de v_i e $|\gamma|$ é o comprimento de γ .



Discretização de Laplace-Beltrami

Sabe-se que:

$$\lim_{|\gamma| \rightarrow 0} \frac{1}{|\gamma|} \int_{v \in \gamma} (v_i - v) dl(v) = -H(v_i) n_i$$

em que $H(v_i)$ é a curvatura média de v_i e n_i é a normal à superfície.

Intuitivamente, as δ -coordenadas **encapsulam a forma local da superfície**.



Vantagens das δ -coordenadas

- Representam detalhes locais
 - A direção aproxima o vetor normal;
 - A norma aproxima a curvatura média;
- Utilização de matrizes esparsas (economizam tempo/memória computacionalmente).



Reconstrução das coordenadas cartesianas

Reconstrução das coordenadas cartesianas a partir das δ -coordenadas

$$L_s \mathbf{x} = \delta^{(x)}$$

- δ -coordenadas são invariantes à translação;
- L e L_s são singulares;
- $\mathbf{x} = L_s^{-1} \delta^{(x)}$ é **indefinido**.
- **Solução:** adicionar restrições de quem se conhece a localização.



Reconstrução das coordenadas cartesianas

Reconstrução das coordenadas cartesianas a partir das δ -coordenadas

$$L_s \mathbf{x} = \delta^{(x)}$$

- $\text{rank}(L) = n - k$, em que k é o número de componentes;
- Fixar vértices $C = \{1, 2, \dots, m\}$, em que sabe-se $\{v_{C_1}, v_{C_2}, \dots, v_{C_m}\}$;
- Ou seja, adicionar restrições $v_j = c_j$, $j \in C$;
- Tornando o sistema *full-rank*.



Reconstrução das coordenadas cartesianas

Reconstrução das coordenadas cartesianas a partir das δ -coordenadas

Fixar vértices $C = \{1, \dots, m\}$ em que sabe-se $\{v_{C_1}, v_{C_2}, \dots, v_{C_m}\}$:

$$\left(\frac{L}{I_{m \times m} | 0} \right) \mathbf{x} = \begin{pmatrix} \delta^{(x)} \\ c_{1:m} \end{pmatrix}$$

E o mesmo para os outros vetores de coordenadas $\mathbf{y}, \mathbf{z}, \dots$



Reconstrução das coordenadas cartesianas

Inicialmente, tem-se:

$$\boxed{L_s} \quad \boxed{x} \quad = \quad \boxed{\delta^{(x)}}$$



Reconstrução das coordenadas cartesianas

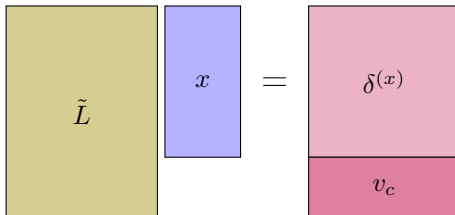
Fixar vértices $C = \{1, \dots, m\}$ em que sabe-se $\{v_{C_1}, v_{C_2}, \dots, v_{C_m}\}$:

$$\begin{array}{|c|} \hline L_s \\ \hline I_c \\ \hline \end{array} \quad x = \begin{array}{|c|} \hline \delta(x) \\ \hline v_c \\ \hline \end{array}$$



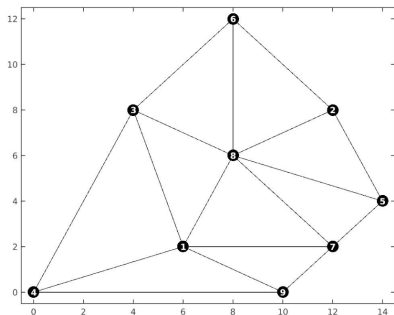
Reconstrução das coordenadas cartesianas

Fixar vértices $C = \{1, \dots, m\}$ em que sabe-se $\{v_{C_1}, v_{C_2}, \dots, v_{C_m}\}$:


$$\tilde{L} \quad x \quad = \quad \begin{array}{c} \delta(x) \\ v_c \end{array}$$



Reconstrução das coordenadas cartesianas

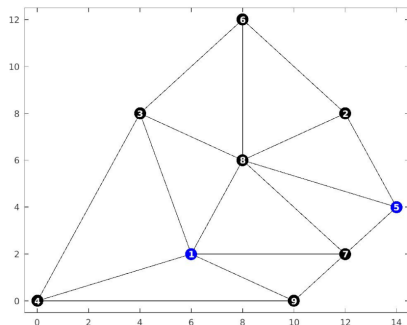


$$L_s = \begin{pmatrix} 5 & 0 & -1 & -1 & 0 & 0 & -1 & -1 & -1 \\ 0 & 3 & 0 & 0 & -1 & -1 & 0 & -1 & 0 \\ -1 & 0 & 4 & -1 & 0 & -1 & 0 & -1 & 0 \\ -1 & 0 & -1 & 3 & 0 & 0 & 0 & 0 & -1 \\ 0 & -1 & 0 & 0 & 3 & 0 & -1 & -1 & 0 \\ 0 & -1 & -1 & 0 & 0 & 3 & 0 & -1 & 0 \\ -1 & 0 & 0 & 0 & -1 & 0 & 4 & -1 & -1 \\ -1 & -1 & -1 & 0 & -1 & -1 & -1 & 6 & 0 \\ -1 & 0 & 0 & -1 & 0 & 0 & -1 & 0 & 3 \end{pmatrix}$$

Figura 2: Exemplo da matriz L_s



Reconstrução das coordenadas cartesianas



$$\tilde{L} = \begin{pmatrix} 5 & 0 & -1 & -1 & 0 & 0 & -1 & -1 & -1 \\ 0 & 3 & 0 & 0 & -1 & -1 & 0 & -1 & 0 \\ -1 & 0 & 4 & -1 & 0 & -1 & 0 & -1 & 0 \\ -1 & 0 & -1 & 3 & 0 & 0 & 0 & 0 & -1 \\ 0 & -1 & 0 & 0 & 3 & 0 & -1 & -1 & 0 \\ 0 & -1 & -1 & 0 & 0 & 3 & 0 & -1 & 0 \\ -1 & 0 & 0 & 0 & -1 & 0 & 4 & -1 & -1 \\ -1 & -1 & -1 & 0 & -1 & -1 & -1 & 6 & 0 \\ -1 & 0 & 0 & -1 & 0 & 0 & -1 & 0 & 3 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Figura 3: Exemplo da matriz \tilde{L} fixando os vértices 1 e 5



Reconstrução das coordenadas cartesianas

Código em Octave (recuperação das coordenadas cartesianas a partir das δ -coordenadas)

```
n = size(L, 1); % numero de vertices
[m, x] = size(values) % numero de [vert conhecidos, dimensão]

L = [L; zeros(m, n)]; % expande L com I_mxm
delta = [delta; zeros(m, n)]; % expande Delta
for i=1:m
    L(i+n, known(1,i)) = 1;
    delta(i+n,:) = values(i,:);
end

v = []; % resolve para cada coordenada
for i=1:x
    v = [v (L \ delta(:, i))];
end
```

Representação de molas e malhas dinâmicas

Utilizando pesos, é possível simular o comportamento de molas para as arestas.

Lei de Hooke

Seja F a força elástica exercida por uma mola, k a constante de resistência específica da mola e x a deformação sofrida pela mola. A Lei de Hooke descreve a força F em função de k e x pela fórmula:

$$F = -kx \quad (2.4)$$



Lei de Hooke e Pesos

Se k for inversamente proporcional à distância entre dois vértices vizinhos, i e j , ou seja, ao tamanho da aresta entre eles, arestas mais curtas serão mais rígidas, e portanto os efeitos de uma transformação em i serão refletidos mais claramente em j . Da mesma forma, conforme o tamanho da aresta aumenta, a propagação das alterações do vértice i no vértice j será menor.



Malhas Dinâmicas

O uso de Malhas Dinâmicas é um modelo de transformações de domínio em modelagem baseada em malhas em que não há alterações na interconectividade dos vértices, ou seja, novas arestas não são criadas e arestas existentes não são removidas. As transformações são realizadas movimentando os vértices da malha. Sendo assim, não há necessidade de remalhamento após transformações exceto em casos de deformações extremas.



Criação de Malhas Dinâmicas

A criação de malhas dinâmicas consiste na distribuição de molas virtuais pela malha original. Estas molas movem os vértices vizinhos quando um vértice i é manipulado, com estas transformações sendo propagadas de forma cada vez mais branda aos vértices vizinhos destes, de forma que a malha se conforme à nova descrição do domínio. Estas molas também possuem o objetivo de impedir que formas geométricas inválidas sejam criadas durante a manipulação dos vértices.



Representação eficiente de formas

Se utilizada uma boa base para representação, é necessária apenas uma parcela da função de base para representar toda a geometria.

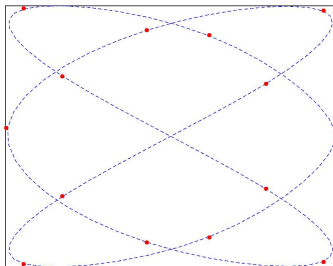


Figura 4: Função paramétrica estimada a partir dos pontos em vermelho (âncora) e de informações de conectividade da malha.



Representação eficiente de formas

Com apenas informações de conectividade da malha e alguns pontos fixados (denominados vértices âncora), é possível aproximar toda a geometria, a partir da resolução do sistema pelo método dos mínimos quadrados:

$$\left(\frac{L}{\omega I_{m \times m} | 0} \right) \mathbf{x}' = \begin{pmatrix} 0 \\ \omega c_{1:m}^{(x)} \end{pmatrix} \quad (3.5)$$

em que $c = \{v_1, v_2, \dots, v_m\}$ são os pontos âncora escolhidos como amostra, e $\omega > 0$ é o peso de cada restrição).

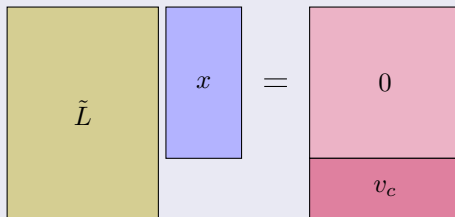


Aplicações

Representação eficiente de formas

Representação eficiente de formas

Graficamente, para facilitar a visualização, este sistema matricial também pode ser visto deste jeito:



The diagram illustrates a matrix equation. On the left, a tall yellow rectangle is labeled \tilde{L} . To its right is a shorter blue rectangle labeled x . An equals sign follows. On the right side of the equals sign is a pink rectangle divided into two horizontal sections. The top section is labeled 0 and the bottom section is labeled v_c .



Reconstrução - observações

Em vez de $\delta^{(x)}$ do lado direito, informa-se 0 nas coordenadas em que não sabe a localização.

Estas localizações serão calculadas por informações das malhas, descritas na matriz \tilde{L} .

Por isso, é importante que se tenha conhecimento de informações de conectividade da malha em análise.



Aplicações

Representação eficiente de formas

Reconstrução - observações

Caso se saiba que a malha é, por exemplo, uma curva contínua e fechada no \mathbb{R}^2 , pode-se utilizar o seguinte código para geração da matriz de adjacências:

Código em Octave (geração de matriz de adjacências)

```
function [ A ] = geraAdjacenciaPadrao(n)
% Entrada:
%   n: numero de vertices
A = zeros(n);
for i=1:n
    A(i,mod(i-2, n)+1) = 1;
    A(i,mod(i, n)+1) = 1;
end
end
```

Aplicações

Representação eficiente de formas

Código em Octave (reconstrução de curva a partir de poucos pontos)

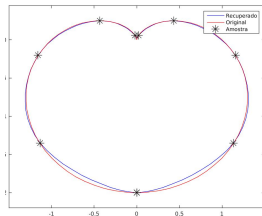
```
n = 100; % numero de pontos no total
n_coords = 2; % dimensao
t = linspace(0, 2*pi, n); % dominio
v = [cos(t) + sin(t); cos(t)]; % v(x, y)'; % v(x, y)

n_known = 11; % reconstrucao com pontos linearmente espaçados
known = round(linspace(1, n, n_known));
v_known = zeros(n, n_coords);
v_known(known,:) = v(known,:);

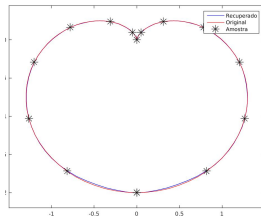
A = geraAdjacenciaPadrao(n);
Ls = geraLaplaciano(A);
delta = zeros(n, n_coords);
recovered = recuperaCoords(Ls, delta, known, v_known(known,:))
```

Aplicações

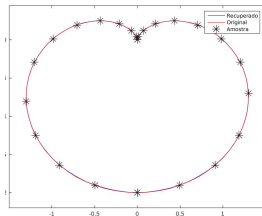
Representação eficiente de formas



(a) 10 pontos



(b) 15 pontos



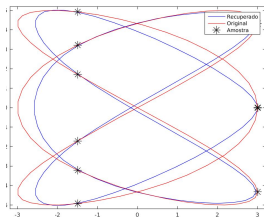
(c) 25 pontos

Figura 5: Representação da função paramétrica $(x(t), y(t)) = (\sin(t) + 0.5\sin(2t), -\cos(t) - 0.5 - 0.5\cos(2t))$ utilizando alguns pontos igualmente espaçados como amostra.

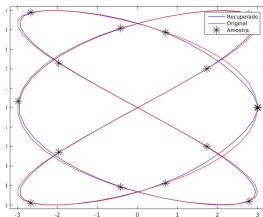


Aplicações

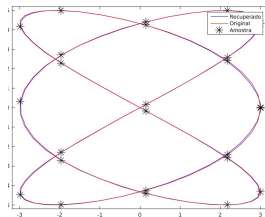
Representação eficiente de formas



(a) 10 pontos



(b) 15 pontos



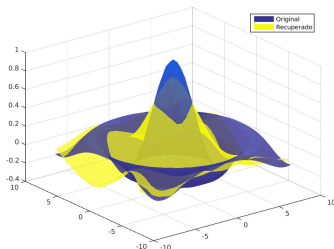
(c) 25 pontos

Figura 6: Representação da função paramétrica $(x(t), y(t)) = (3\cos(3t), 5\sin(2t))$ utilizando alguns pontos igualmente espaçados como amostra.

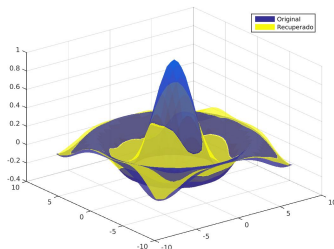


Reconstrução de superfícies

Também é possível representar superfícies lineares por partes a partir de poucas amostras na variedade original.



(a) 50 pontos



(b) 200 pontos

Figura 7: Reconstrução de superfície utilizando alguns poucos pontos.



Edição de malhas

Outra interessante aplicação é a de edições de malhas, como movimentos e deformações.

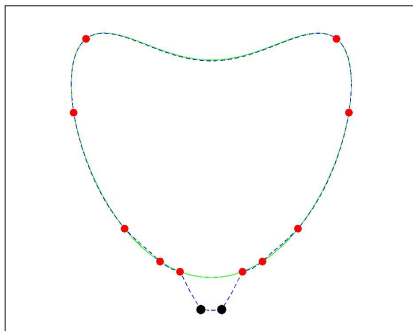


Figura 8: Malha original (verde), malha editada (vermelho), pontos âncora (vermelho) e pontos de edição (preto)



Edição de malhas

Além dos pontos âncoras fixados vistos anteriormente, também são escolhidos alguns pontos de edição, em que se deseja alterar a posição cartesiana, e são adicionadas restrições do tipo:

$$\mathbf{v}_i = \mathbf{c}_i, \quad i \in \{1, \dots, m\} \quad (3.6)$$

$$\mathbf{v}_j = \mathbf{e}_j, \quad j \in \{m+1, \dots, m+a\} \quad (3.7)$$

e, matricialmente, o sistema pode ser escrito como:

$$\left(\frac{L}{\omega I_{(m+a) \times (m+a)} | 0} \right) \mathbf{x}' = \begin{pmatrix} \delta^{(x)} \\ \omega c_{1:m}^{(x)} \\ \omega e_{1:a}^{(x)} \end{pmatrix} \quad (3.8)$$

Edição de malhas

Graficamente, para facilitar a visualização, este sistema matricial também pode ser visto como:

$$\begin{array}{c} L_s \\ I_c \\ I_e \end{array} \begin{array}{c} \mathbf{x}' \\ \\ \end{array} = \begin{array}{c} \delta^{(x)} \\ c^{(x)} \\ e^{(x)} \end{array}$$



Edição - observações

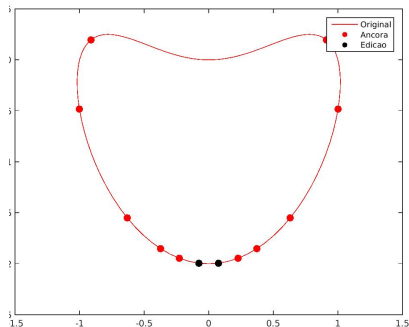
Como é utilizado o método dos mínimos quadrados para a resolução dos sistemas, talvez aconteça alguns erros de precisão.

Para minimizar estas falhas, após se encontrar a solução do sistema, pode-se forçar com que $v'_i = v_i$, para alguns i , para que pelo menos estes pontos não sofram erros de precisão;

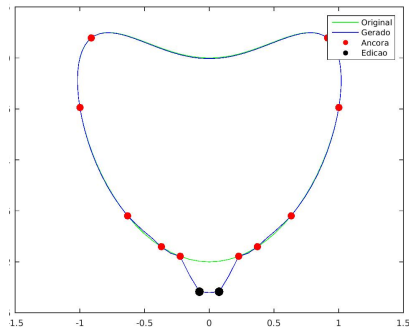


Aplicações

Edição de malhas



(a) Malha original










(b) Malha editada

Figura 9: Malha antes e após edição. Os pontos em vermelho são os pontos de âncora, e os pontos em preto são os editados.






Referências I

-  M. Agoston, *Computer graphics and geometric modeling*, Springer-Verlag, London, 2005.
-  C. M. Hoffmann, *Geometric and solid modeling: An introduction*, Morgan Kaufmann Publishers, 1989.
-  H. Hagen and D. Roller, *Geometric modeling: Methods and applications*, Springer-Verlag, Berlin, 1991.
-  M. Mântylâ, *An introduction to solid modeling*, Computer Science Press, 1988.
-  Martin Newell, *Utah teapot*, 1975.
-  U. Pinkall and K. Polthier, *Computing discrete minimal surfaces and their conjugates*, *Experimental Mathematics* **2** (1996).
-  F. Petronetto, A. Paiva, E. S. Helou, D. E. Stewart, and L. G. Nonato, *Mesh-free discrete laplace-beltrami operator*, *Computer Graphics Forum* **32** (2013), no. 6, 214–226.



Referências II

-  Francisco A. Rodrigues, *Análise morfológica de imagens*.
-  I. P. Soares, *Movimento de malhas e remalhamento de malhas superficiais*, Ph.D. thesis, ICMC-USP, São Carlos, 2007.
-  O. Sorkine, *Differential representations for mesh processing*, Computer Graphics Forum **25** (2006), no. 4, 789–807.

