An Method of Feature Line Extraction of Triangle Mesh Surface Model*

Zhao Jibin, Liu Weijun, Xia Renbo
Shenyang Inst. of Automation, Chinese Academy of Sciences
Shenyang, Liaoning Province, China
jbzhao@sia.cn

Abstract - An algorithm of effective feature line extraction for triangle mesh model is proposed. Feature line extraction of triangle mesh model is not only the key technology of reverse engineering, but also an important research project of computer vision and machine intelligence. In the paper, firstly the principal curvatures and principal directions of vertex of triangle mesh model are analyzed and feature vertex is confirmed, then the method of “seed growth” to extract feature line is proposed. Experiments and applications show that the method is not only simple and extracting feature rapid, but also extract stably and exactly feature line.

Index Terms - Feature Line, Triangle mesh model, conic surface.

I. INTRODUCTION

Reverse engineering [1] is the core of rapid Product R&D and manufacturing, and this new technology carries great weight among the manufacturing nowadays, which draws more and more attention of the industries. Triangle mesh model is widely applied in computer graphics, geometric modeling and reverse engineering because of its capability in describing topological relations between data points, its visual effects, and its convenience in Rapid Prototyping (RP) and manufacturing integration. A typical modeling procedure in reverse engineering is as the following: 1) Data acquisition 2) Preprocessing, 3) Region segmentation, 4) Surface approximation, 5) Computer modeling. On one hand, feature extraction is the key of modeling since region segmentation is to divide a data cloud into segments with single feature and without overlaps with each other. On the other hand, precise extraction of feature lines in a spatial triangular mesh surface model is a hot research project in the fields such as pattern recognition, computer vision and machine intelligence.

So far, there’re mainly two methods based on data clouds of sample parts in experiments: One is Edge-Based and the other one is Surface-Based. The edge-based one assumes that the normal vector or the abrupt change of curvature is along the edges of two regions, and it holds that the regions with closed edges are the final result of division [2]. In 1995, Huang and his team members [3] in Ohio State University triangulated a data cloud and recognized the edges through the edge-based method. Then they achieved region-segmentation through region-growth method to sort the patches. In the year 2002, Woo and his colleagues [4] in GIST proposed a 3D grid-subdivision method based on Octree-Structure. In their algorithm, data obtained by laser-scanning were triangulated first, and then the normal vectors were estimated. And then a bounding box containing data clouds was being subdivided based on Octree Structure until the standard deviations of the estimated normal vectors of the points in the grid space become less than user input value. The grids with their side lengths less than a specific value are integrated, and the edges of the clouds are recognized at last. Thus the region segmentation of the scanned data is achieved based on region-growth method. The surface-based method [5] is to sort apical points with similar geometric features to one region. Significant special-geometric-change feature shall exist on surfaces near the contour or in the normal direction of the triangular patches nearby. Extraction or tracking the surfaces nearby is similar to extracting edges of surface patches. (It’s also called ‘Surface feature line extraction in a surface-model reverse engineering project.) Liu [6] adopted the edge-based method and extracted the feature points twice of triangle mesh surface model. Then she conjoined the points extracted into initial feature lines, which could be interactively modified till desirable feature lines appeared. Jia [7] aimed at triangular patches, proposed a feature-line-extracting method through triangular-surfaces simplification of a multi-resolution model, which followed the line of planarity. (That is to combine the triangular patches available through comparison of normal vectors by utilizing triangle grids and their normal vectors in a triangular-patch model, to extract the edges of the combined regions, and to obtain the edges namely the feature lines of the surface at last.) Lv [8] managed to obtain the feature points of a data cloud by using angle-calculation scan. He determined the intersection points of feature curves and scanning lines by their projection. Reckoning them as constraints, he triangulated the scanned data on the cloud to enable the curve to reflect the abrupt curvature changes on the surface. At the end, he added features to have the curve more approximate to the original feature curve. Deschenes [9] extracted the corner points of a model in a 3D scene, and he obtained the contours composed of digitalized curves by combining them according to their topological relations. Milroy [10] divided the data on the basis of OCS(Orthogonal Cross Section) model, then he calculated the principal curvatures and directions of a coincide. At last splines were formed by edging points connection, following the line of energy minimization. Yang

∗ This work is supported by CNSF #50975274.
achieved feature lines extraction from a dense-data cloud through building a coincide with local parameters to calculate the principal curvatures and principal directions.

II. THE SETTLE METHOD OF FEATURE VERTICES

A. The Normal Vector of Triangular Mesh Vertex

Curvature becomes larger at ridge lines, surface-intersections and edges in a surface model, so it’s advisable to extract feature points by calculating the curvature features of the vertex on triangular mesh surface. The main idea is fitting parabolic surface $S$ with vertex $P_i$ and its neighboring vertices. The feature points would be determined by calculating the principal curvature and principal direction of $S$ to obtain curvature features of $P_i$.

To avoid singularity in an explicit surface fitting, in this paper, we proposed a least square method for surface fitting based on parameterization of local tangent space. The normal vector of the point $P_i$ on the triangular patch needs to be estimated in order to fit the surface. As is shown in Fig.1, $n_j$ is the unit normal vector of the triangular patch adjacent to the point $P_i$ and $S_j$ is the area, then the normal vector of the point $P_i$ is:

$$N_i = \frac{\sum_{j=1}^{k} s_j n_j}{\sum_{j=1}^{k} s_j} \cdot N_i = \frac{N_i}{|N_i|}. \quad (1)$$

Fig.1 Normal vector of $P_i$

B. Calculation Method of Principal Curvature on Mesh Vertices

According to the above, the local coordinates must be determined in first hand, before the neighborhood of the vertex is expressed. To be precise, one of the local coordinates axes should coincide with the normal vector of the vertex and the coordinate origin is selected as the vertex of the triangular mesh. Then local surface is fit for $k$ vertices adjacent to others. $V_j$ is the origin point of local coordinates and $K$ is taken as $Z$-axis. The set of neighborhood points is $\{v_s \mid s = 1, \cdots, n\}$, and the local coordinates of $V_i$ is $\{v_i, e_1, e_2, e_3\}$. As is shown in Fig.2, $e_3$-axis is in the direction of $K$ (along $h$-axis), so the plane equation perpendicular to $K$ can be expressed as:

$$K \cdot (X - X_i) = Ax + By + Cz + D = 0. \quad (2)$$

Suppose,

$$d_j = Ax_j + By_j + Cz_j + D, \quad v_j^p = v_j - d_j k. \quad (3)$$

where the unit vector of $h$-axis $e_3 = k$, the unit vector of $u$ axis $e_1 = (v_i^p - X_i) / |v_i^p - X_i|$, and the unit vector of $V$ axis $e_2 = e_3 \times e_1$. Thus it can be seen that the coordinate components of a point in this set of neighboring points can be expressed as:

$$(u_j, v_j, d_j) = ((v_i^p - X_i) \cdot e_1, (v_i^p - X_i) \cdot e_2, d_j). \quad (4)$$

With the coordinates conversion above, the set of neighboring points of position-vector $X_i$ is converted into the local coordinates system. Quadric surface is adopted to approximate the local point cloud, and suppose that

$$z(u, v) = \sum_{i=0}^{n} \sum_{j=0}^{m} a_{i-j, j} u^i v^j. \quad (5)$$

The neighboring vertices of point $V_i$ is fit by least squares method:

$$J = \min_{a_{i-j, j}} \left[ \sum_{k=1}^{n} [z(u_k, v_k) - d_k]^2 \right]. \quad (6)$$

The equations can be solved through solving the linear equation set. In this paper, the quadric surface is a parabolic surface with its expression $s = au^2 + buv + cv^2$, then equation (6) can be transformed into $A x = B$, where

$$A = \begin{bmatrix}
  u_1^2 & u_1v_1 & v_1^2 \\
  \vdots & \vdots & \vdots \\
  u_m^2 & u_mv_m & v_m^2 \\
\end{bmatrix}, \quad x = \begin{bmatrix}
  a \\
  b \\
  c \\
\end{bmatrix}, \quad B = \begin{bmatrix}
  d_1 \\
  \vdots \\
  d_m \\
\end{bmatrix}. \quad (7)$$

To assure full rank of the coefficient matrix of the normal equations, the number of neighboring vertices must be 3 at least, the only solution of the equations is as the following:

$$x = (A^T A)^{-1} A^T B \quad (8)$$

In this way, the surface equation of point $V_i$ can be obtained. Thus the first-order partial derivatives and the second-order partial derivatives of the parabolic surface at the local original point are as the following:
$S_u \Big|_{(0,0)} = (1, 0, 2au + bv) = (1, 0, 0),$
$S_{uu} \Big|_{(0,0)} = (0, 0, 2a),$
$S_u \Big|_{(0,0)} = (0, 1, bu + 2cv) = (0, 1, 0),$
$S_{uv} \Big|_{(0,0)} = (0, 0, 2c),$
$S_{vv} \Big|_{(0,0)} = (0, 0, b).$

The normal vector at the origin is

$N_i \Big|_{(0,0)} = \frac{S_u \times S_v}{|S_u \times S_v|} = (0, 0, 1).$

The first basis quantities of parabolic surface $S$:

$E = S_u S_u = 1; F = S_u S_v = 0; G = S_v S_v = 1.$

The second quantities of parabolic surface $S$:

$L = S_{uu} N_i = 2a; M = S_{uv} N_i = b; N = S_{vv} N_i = 2c$

Then the normal curvature $k$ of the original point can be described as

$k = \frac{L + 2M \lambda + N \lambda}{E + 2F \lambda + G \lambda^2} \quad (9)$

Where, the roots of $\lambda = \frac{du}{dv}; \frac{dk(\lambda)}{d\lambda} = 0$ are $\lambda_1$ and $\lambda_2$.

Meanwhile, the extreme value $k_1$ and $k_2$ can be obtained, respectively corresponding to the minimum principal curvature and the maximum principal curvature of the parabolic surface at the origin point. They are expressed as

$k_1 = a + c - \sqrt{(a-c)^2 + b^2}$
$k_2 = a + c + \sqrt{(a-c)^2 + b^2}$

The principal directions, $m_1$ and $m_2$, are as the following:

$m_1 = (c - a - \sqrt{(a-c)^2 + b^2}, -b)$
$m_2 = (c - a + \sqrt{(a-c)^2 + b^2}, -b)$

$m_3 = (b, c - a - \sqrt{(a-c)^2 + b^2})$
$m_4 = (b, c - a + \sqrt{(a-c)^2 + b^2})$.

C. Determination of Extreme value for Principal Curvatures

Firstly, a plane is set cross the normal vector of $P_i$, and the principal direction $m_1$ (As is shown in Fig.3). The plane intersect the edges of neighboring triangle areas at point $A$ and $B$. Secondly, $k_1$ along direction $m_1$ is calculated for the $k_1$ of the endpoints of the line-segment along direction $m_1$. Thirdly, the principal curvature $k_1$ of $P_i$ is compared with the $k_1$ of point $A$ and $B$. The point $P_i$ will be the feature point if the value $k_1$ is larger. In the same way, $k_2$ and $m_2$ can be judged on, and it can also be determined whether the vertex is the feature point along direction $m_2$ or not.

Fig.3 Determination of extreme value for principal curvatures

III. EXTRACTING ALGORITHM OF FEATURE LINES

Although the feature vertices of the triangular mesh model are already determined, it still requires systematization for them. To describe the shape of the model, these random scattered discrete vertices need connecting to form ordered feature lines. In this paper, an algorithm of extracting feature lines named ‘seed-growth’ is proposed. In the first place, ‘linear-growth’ of the seed starts at ‘seed point’ to research for neighboring candidates for the feature point that describes the same feature. The growth keeps going on until it goes back to the ‘seed point’ or there’s no candidate around. The feature lines stop growing then. In case the feature points on a feature line are less than 3, the feature line shall be discarded as it is not a significant feature, and the seed is considered to be a bad one. In the second place, the feature points are fitted into NURBS curve, which have already been connected into lines. Detailed algorithm is as the following:

Input: STL model of triangular mesh
Output: Feature lines in the form of NURBS

Step 1: Load the file of triangular mesh, and adopt red-black tree method to swiftly build the topological relations of the model, to implement search of neighboring vertices with high efficiency.

Step 2: Go through all the vertices of the triangular mesh. Determine the feature vertices. Set Feature_flag=1 and insert the feature vertices into a link list of feature points.

Step 3: Search the link list of feature points for the seed vertex VF of the feature. If the flag of the first feature point found out is Feature_flag=1, it’ll be the seed of the feature. Then build a link list of feature lines, and then set Feature_flag=0.

Step 4: Search the neighboring vertices of the seed vertex, and if there’s no feature vertex among them, go to step 7. In case there’s only one feature vertex, insert this feature vertex VF1 into the link list of feature lines, and set Feature_flag=0; In case there are two or more feature points among the neighboring vertices, select one of them as continuous point seed-growth, and this vertex is current-feature-vertex.

Step 5: Search the neighboring vertices of the current vertex, and if there’s no feature vertex among them, go to step 7. In case there’s only one feature point, insert it into the link list of feature lines. In case there’re two or more feature points, then calculate the line connecting VF1 and VF0, and
the angle. As is shown in picture 4, there are three neighboring feature points P0, P1 and P2. Choose feature point P2 with its least angle in the three, and put it into the link list of feature lines. Set Feature_flag=0 and it is the current feature point.

**Step 6**: Repeat step 5. The feature line stops growing, neither it goes back to the original namely the seed of the line or there’s no feature point near the current feature vertex.

**Step 7**: Go back to the origin namely the seed, repeat step 5 and 6 until the feature line stops growing again. Then the feature-line extraction completes. But it is not a significant feature in case there are less than 3 feature points in the link list of feature lines. If so, the link list of feature lines must be deleted.

**Step 8**: Repeat step 3 to 7, until feature points are all determined.

**Step 9**: Fit into NURBS curve the feature points in the link lists of feature lines.

In this paper, the whole algorithm is implemented with Visual C++, and as is shown in Fig.5 and Fig.6, here are the triangular mesh model and the feature lines from the model.

![Fig.4 selection of the feature point](image)

**Fig.4 selection of the feature point**

**IV. CONCLUSION**

In this paper, an extracting method of feature lines for triangular mesh is proposed. At first, the feature vertices are determined by analyzing the principal curvatures and the principal directions. Then a method of ‘seed-growth’ is presented to extract the feature lines. Examples and works in practices have testified to the simplicity, the efficiency, the precision and the stability of the method in the paper to extract significant features from a triangular mesh model.

![Fig.5 Triangular mesh model](image)

**REFERENCES**


