AUTOMATIC EXTRACTION OF FEATURE LINES ON 3D SURFACE

Zhihong Mao, Ruichao Wang and Yulin Zhou

Division of Intelligent Manufacturing, Wuyi University,
Jiangmen529020, China

ABSTRACT

Many applications in mesh processing require the detection of feature lines. Feature lines convey the inherent features of the shape. Existing techniques to find feature lines in discrete surfaces are relied on user-specified thresholds, inaccurate and time-consuming. We use an automatic approximation technique to estimate the optimal threshold for detecting feature lines. Some examples are presented to show our method is effective, which leads to improve the feature lines visualization.

KEY WORDS

Feature Lines; Extraction; Meshes

1. INTRODUCTION

Advances in scanner technology and algorithms for constructing polygonal meshes made polygonal models currently dominate the field of 3D computer graphics. In addition, meshes support wide variations in complexity. Almost any shape representation may be converted with arbitrary accuracy to a polygonal mesh. Fine shape representation is important for 3D geometry processing, such as shape analysis, shape matching and shape editing. Many applications in mesh processing require the detection of feature lines. Feature lines convey the inherent features of the shape. Mathematically feature lines are described via extrema of the surface principal curvatures along their corresponding lines of curvature, which are traced by using high-order curvature derivatives. Forrester Cole et al. [1] pointed out that current computer graphics line drawing techniques can effectively depict shape and even match the effectiveness of artist's drawings, and that errors in depiction are often localized.

Recent research in 3D computer graphics has focused on estimating feature lines over triangle mesh models [2-6]. The basic idea of these papers is first to robustly estimate surface principal curvature and then to identify the feature lines as lines of curvature extrema. Polygonal surface models usually have large flat regions with sharp creases on which the surface bend sharply.

One obvious way to extract the sharp creases from the model is to select a threshold T that separates these points with surface curvature extrema. Because of its intuitive properties and simplicity of implementation, the threshold method is used by most existing methods to achieve the extraction of the sharp features on a mesh model. The success of this method depends largely on how to set the threshold. Unfortunately, improper threshold produces many spurious feature lines. It is difficult for a user to modify the threshold if no any apriori knowledge. A successful outcome demands a lot of skills for the threshold method.

Natarajan Meghanathan et al. (Eds) : CCSEA, CLOUD, SIPRO, AIFU, SEA, DKMP, NCOM - 2019
pp. 01-11, 2019. © CS & IT-CSCP 2019 DOI: 10.5121/csit.2019.90901

We present an automatic method for robustly extracting feature lines on triangle mesh models. In the first step, fundamental descriptors for shape analysis--namely, the principal curvature and corresponding direction—are computed for each vertex in the mesh. However, the computation of curvature and its derivatives is sensitive to noise and irregularities of the triangulation. In order to accurately extract the feature lines, in the second step we introduce the saliency value [7, 8] computed by a linear combination of the maximal absolute curvature and the absolute curvature difference, a measure of regional importance for graphics meshes, to reduce the false feature points. Finally, we automatically determine the threshold according to the saliency value.

The paper is structured as follows. Section 2 reviews related work. Section 3 describes necessary background of differential geometry. Section 4 describes the automatic algorithm to robustly extract feature lines from triangulated meshes. Section 5 presents some results and compares them to the other methods. Section 6 concludes the paper.

2. RELATED WORK

Many applications in 3D computer graphics require the extraction of feature lines in a discrete mesh. Mathematically feature lines are defined as extrema of the principal curvatures corresponding to their curvature directions. So extraction of feature lines requires estimation of the principal curvature and direction as a first step.

Estimation of Curvature Briefly, Curvature estimation methods can be categorized as follows: 1) Normal curvature approximation method where the Weingarten matrix is used to estimate the principal curvature and direction [9, 10]. 2) Surface (curve) fitting method where a polynomial surface (curve) is fitted to points in a local region [11, 12]. 3) Discrete differential geometry method [13, 14] where discrete versions of differential geometry theorems, such as the Laplace-Beltrami operator and Gauss-Bonnet theorem, are developed and applied to the neighborhood of each vertex. However, a purely curvature-based metric may not necessarily be a good metric of feature lines extraction. Non-uniform sampling, noise and irregularities of triangulation make the curvature-based method to often get false feature lines or incomplete feature lines.

Saliency Value Saliency map [15] that assigns a saliency value to each image pixel has done excellent work in image processing. By the saliency value, salient image edges will be distinct from their surroundings. Encouraged by the success of the method on 2D problem, Lee [7] introduces the idea of mesh saliency as a measure of regional importance for graphics meshes. Lee discusses how to apply the saliency value to graphics applications such as mesh simplification and viewpoint selection. Ran Gal [8] defined salient geometric features for partial shape matching and similarity by the salient parts of an object. By “salience of a part”, he captured the object’s shape with a small number of parts. He proposed that the salience of a part depends on (at least) two factors: its size relative to the whole object, the number of curvature changes and strength. Similar to the two methods, we will compute the saliency value of each mesh point for our automatic algorithm.

Extraction of Feature Lines Feature lines are curves of curvature extrema and therefore encode important information used in segmentation, registration, matching and surface analysis. Sometimes we also call them ridges and valleys or crest lines.

Feature lines extraction techniques aimed to build an economical and accurate representation of surface features have become increasingly popular in computer graphics[16, 17, 23]. A number of approaches have been described. Feature lines extraction can be roughly classified into image-based method and model-based method. Image-based method extracts feature lines on a rendered projection image by edge detection algorithm in image processing [16]. The result is visually

pleasing and less computational complexity. But pixel-based representation gives low precision. Model-based method extracts lines directly on 3D models in term of the differential geometric properties of the surface. Benefited from the development of 3D scanning techniques, digitizing the model by a high resolution makes it possible to depict the differential geometric properties on the digital model.

There are a number of lines that serve to extract the geometric linear features of the surface. The first class is view-dependent curves. The silhouette (contour) is the curves that are only defined with respect to a viewing screen. Suggestive contours are contours that would first appear with a minimal change in viewpoint [18]. Apparent ridges are defined as the ridges of view-dependent curvature, the variation of surface normal with respect to a viewing screen plane [6]. View-dependent curves depend on the viewing direction and produce visually pleasing line drawings. So they are usually used for non-photorealistic rendering methods. The second class is view-independent curves that depend only on the differential geometric properties of the surface. Many researchers have tried to depict the shape of the 3D model with a high-quality feature lines. But the features are traced using high-order curvature derivatives and are not easy to be detected from discrete surface. The most common curves are ridges and valleys which occur at points of extremal principal curvatures. This paper focuses on the problem of accurately detecting feature lines on surfaces. Ohtake et al. [2] proposed a method for detecting ridge-valley lines by combining multi-level implicit surface fitting and finite difference approximations. Because the method involves computing curvature tensors and their derivatives at each vertex, it is time-consuming. Yoshizawa [3] detected the crest lines based on a modification of Ohtake's method. The method reduced the computation times since Yoshizawa's method estimated the surface derivatives via local polynomial fitting. Soo-Kyun Kim [4] found ridges and valleys in a discrete surface using a modified MLS approximation. The algorithm was quick because the modified MLS approximation exploited locality. Stylianou and Farin [5] presented a reliable, automatic method for extracting crest lines from triangulated meshes. The algorithm identified every crest point, and then joined them using region growing and skeletonization.

Other types of curves aren't defined as the maximum of curvature, but defined as the zero crossings of some function of curvature. Demarcating curves [19] are the loci of points for which there is a zero crossing of the curvature in the curvature gradient direction. Demarcating curves can be viewed as the curves that typically separate ridges and valleys on 3D surface. Relief edges [20] are defined as the zero crossing of the normal curvature in the direction perpendicular to the edge. Compared to view-dependent curves, view-independent curves are locked to the object surface, and do not slide along it when the viewpoint changes. Moreover, they are pure geometrical and convey prominent and meaningful information about the shape, we believe there is merit in using these curves.

Our approach is closely related to traditional ridges and valleys. They define the so-called feature lines as the extrema of the surface principal curvatures along their corresponding curvature lines.

3. PRELIMINARIES

A good introduction to differential geometry can be found in [21]. We just recall some notions of differential geometry, useful for the introduction of the extraction of feature lines on 3D meshes.

3.1 Polyhedral Surfaces

A polyhedral surface $M \subset R^3$ is a connected topological 2-manifold which is made up of flat triangles that are glued along their common edges such that no vertex appears in the interior of an edge. We will focus on discrete surfaces represented by triangular meshes, i.e., by a couple

$(\mathbf{V}, \mathrm{T}, \mathbf{N})$ where $\mathbf{V}$ is a set of n vertices $\mathbf{V} = \{\mathbf{V}_i \subset \mathbf{R}^3 \mid 0 \le i \le n-1\}$, T is a set of triplets $T_i(i_1, i_2, i_3) \in \{0,1,2,.....,n-1\}$ and (i_1, i_2, i_3) represents the indices of vertices forming a triangle, $\mathbf{N}$ is a set of the normal vectors attached to the corresponding vertices, $\mathbf{N} = \{\mathbf{N}_i \in \mathbf{R}^3 \mid 0 \le i \le n-1\}$.

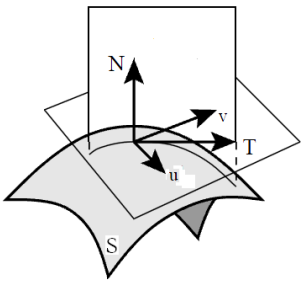

Figure 1 Normal section of Surface S.

3.2 Differential Geometry

Differential geometry of a 2D manifold embedded in 3D is the study of the intrinsic properties of the surface. Here, we will recall basic notions of differential geometry required in this paper. The unit normal $\mathbf{N}$ of a surface S at $\mathbf{p}$ is the vector perpendicular to S, i.e., the tangent plane of S at $\mathbf{p}$. A normal section curve at $\mathbf{p}$ is constructed by intersecting S with a plane normal to it, i.e., a plane that contains $\mathbf{N}$ and a tangent direction $\mathbf{T}$. The curvature of this curve is the normal curvature of S in the direction $\mathbf{T}$ (See Fig.1).

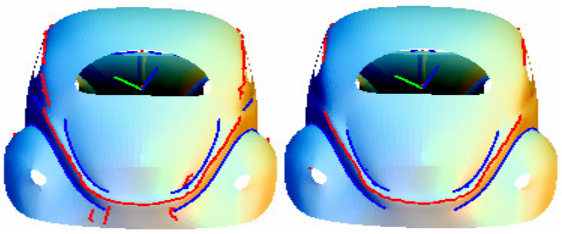

Figure 2: Comparison of the detection algorithm by saliency values with the algorithm only by principal curvatures. Left: Detect the feature lines only by principal curvatures; Right: Detect the feature lines by saliency values, a measure of regional importance.

For a smooth surface, the normal curvature in direction $\mathbf{V}$ is $k(\mathbf{V}) = \mathbf{V}^T II \mathbf{V}$, where the symmetric matrix II is the second fundamental form. The eigenvalues of II are the principal curvature values $(k_{\max}, k_{\min})$. The eigenvectors of II are the coordinates of the principal curvature directions $(\mathbf{t}_{\max}, \mathbf{t}_{\min})$. Let $e_{\max}$ and $e_{\min}$ be the derivatives of the principal curvatures $k_{\max}, k_{\min}$ along their corresponding curvatures directions $\mathbf{t}_{\max}, \mathbf{t}_{\min}$. Mathematically feature lines are described via extrema of the surface principal curvatures along their corresponding lines of curvature:

$$e_{\max} = 0, \quad \nabla e_{\max} \cdot \mathbf{t}_{\max} < 0, \quad k_{\max} > |k_{\min}| \quad \text{(Ridges)} \tag{1}$$

$$e_{\min} = 0, \quad \nabla e_{\min} \cdot \mathbf{t}_{\min} < 0, \quad k_{\min} < -|k_{\max}| \quad \text{(Valleys)} \tag{2}$$

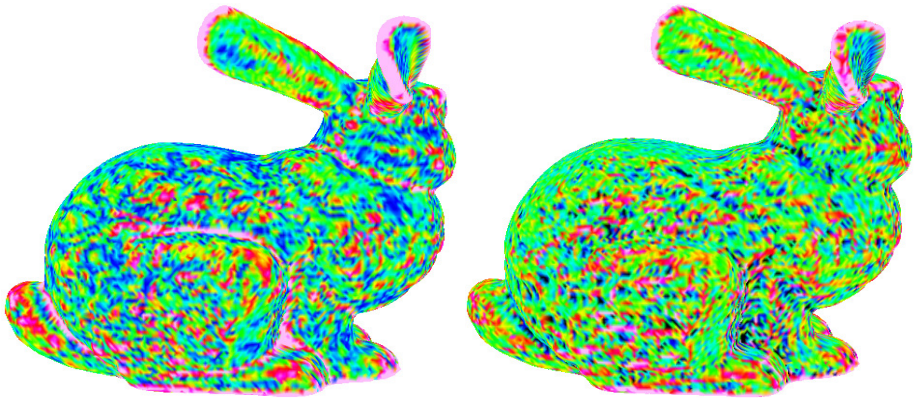

Figure 3: The saliency values can capture the interesting features at all perceptually meaningful scales and reveals the difference between the vertex and its surrounding context. Left: Visualize the saliency values of each mesh points with color. Right: Visualize the principal curvature (max) of each mesh points with color.

4. AUTOMATIC METHOD FOR FEATURE DETECTION

4.1 Detect the Feature Points Via Local Polynomial Fitting

In order to achieve an accurate estimation of the principal curvatures and their derivatives an implicit surface $F(\mathbf{x}) = 0$ need to be constructed to approximate locally the neighborhood of each mesh vertex [3, 4]. We use upper indices for vector components, sub-indices for partial derivatives. So the components of the surface unit normal vector are given by $\mathbf{n}^i = -F_i / |\nabla F|$, where $|\nabla F|$ is the absolute value of the gradient. Let $\kappa, \mathbf{t}$ and s stand for a principal curvature, the associated principal vector, and the arc-length parameter along the associated normal section respectively. Thus the principal curvature κ is given by

$$\kappa = \frac{F_{ij} t^i \mathbf{t}^j}{|\nabla F|} \tag{3}$$

The curvature derivative e is defined by differentiating (3),

$$e = \frac{d\kappa}{ds} = \frac{d}{ds}\left(\frac{F_{ij}\mathbf{t}^i\mathbf{t}^j}{|\nabla F|}\right) = \frac{F_{ijl}\mathbf{t}^i\mathbf{t}^j\mathbf{t}^l + 3\kappa F_{ij}\mathbf{t}^i\mathbf{n}^j}{|\nabla F|} \tag{4}$$

Having found the maximal and minimal curvatures $(k_{\max}, k_{\min})$ and their derivatives $(e_{\max}, e_{\min})$ at each mesh vertex, we can extract ridges and valleys by equation (1) and (2). Computing of the feature lines involves estimation of high-order surface derivatives, so these surface features are very sensitive to noise and irregularities of the triangulation (see Fig.2).

4.2 Modify the Detection Algorithm by Saliency Values

Mesh saliency, a measure of regional importance, can identify regions that are different from their surrounding context and reduce ambiguity in noisy circumstances. In this paper, the computation of the salience value is built on the methods of the salience of visual parts proposed by Ran Gal [8] and mesh saliency proposed by Lee [7]. Differing from the salient parts, we compute a saliency value of each mesh point to identify mesh points that are different from their surrounding

context. We have modified their algorithms slightly to define a saliency value S as a linear combination of two terms :

$$S = W_1 Curv(\mathbf{p}) + W_2 Var(\mathbf{p}) \qquad (5)$$

Where $Curv(\mathbf{p})$ is the maximal absolute curvature of a point $\mathbf{p}$ and $Var(\mathbf{p})$ is the absolute curvature difference. The first term of equation (5) expresses the saliency of the mesh point. The second term expresses the degree of the difference between the point and its surrounding context. We use 0.4 for W_1 and 0.6 for W_2. Let $k(\mathbf{p})$ denote the maximal absolute value of the principal curvatures and $G(k(\mathbf{p}))$ denote the Gaussian-weighted average of $k(\mathbf{p})$,

$$G(k(\mathbf{p})) = \frac{\sum_{x \in neighbor(p)} k(\mathbf{x}) \exp[-|\mathbf{x}-\mathbf{p}|^2 / (2\sigma^2)]}{\sum_{x \in neighbor(p)} \exp[-|\mathbf{x}-\mathbf{p}|^2 / (2\sigma^2)]} \qquad (6)$$

σ is a scale factor that is estimated by $\sigma = \lambda \bar{e}$, where $\bar{e}$ is the average length of edges of the mesh. We compute the saliency value $\varphi_i(\mathbf{p})$ of a vertex $\mathbf{p}$ as the absolute difference between the Gaussian-weighted averages $G(k(\mathbf{p}))$ computed at the two neighboring boundaries:

$$\varphi_K(p) = |G(k(p), K+1) - G(k(p), K)| \qquad (7)$$

$Var(p)$ is computed by an average value at multiple scales:

$$Var(p) = \sum_{K=1}^{n} \varphi_K(\mathbf{p}) \qquad (8)$$

Where n is set 3 or 4. We define a salient geometric feature point as a measure of regional importance which is salient and interesting compared to its neighborhood. The top graded points define the salient geometric feature of a given shape. So it is better than the method extracting the feature lines only by the curvature (see Fig.2 and Fig.3).

4.3 An Automatic Feature Points Detection Method

Suppose that the surface points are composed of feature points and flat points. One obvious way to extract the feature points on a mesh is to select a threshold T that separates these mesh points. Because of its intuitive properties and simplicity of implementation, threshold method enjoys a central position in applications of the extraction of the surface feature points. Unfortunately it is difficult for a user to set the threshold if no any apriori knowledge (see Fig.4 and Fig.6)

Our automatic algorithm is based on the observation that polygonal surface models with features usually contain many flat regions and a little sharp edges. Sorting the mesh saliency value of each point by the ascending order, we can find: The saliency values begin to increase slowly and the plot almost corresponds to a planar line, after arriving to a value, the plot rises steeply. Obviously, flat regions correspond to the planar line part of the plot and sharp edges correspond to the steep line part of the plot. So this value is the optimal threshold to extract the feature points (see Fig.5).

Finally, we summarize the automatic feature lines extraction algorithm as follows:

1. Estimate necessary surface curvature and their derivatives via local polynomial fitting.

2. Compute the saliency values as a measure of regional importance for graphics meshes.
3. Seek the optimal threshold by the analysis mentioned above.
4. Extract the feature points by the threshold.

For step 3, we give the pseudocode as follow:

Procedure SeekThreshold(saliency)

1. Quicksort (saliency) /* Sort the mesh saliency value of each point.*/
2. for i = vertex_num / 2 to vertex_num step k
   /* vertex_num represents the number of mesh vertices. Firstly, we search the value at a coarse scale, usually set k = vertex_num /200 . */

2.1. m_i = saliency[i] – saliency[i-1]; m_{i-1} = saliency[i-1] – saliency[i-2]

2.2 if $m_i > 1.3 * m_{i-1}$ /*From Fig.5 we can find the plot rises steeply, we used 1.3 for the coefficient.*/

Then shrink the search range in size and repeat step 2 at a finer scale till finding the threshold.

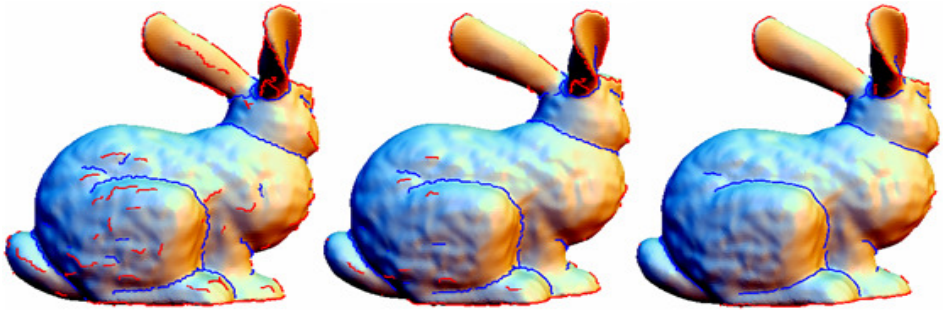

Figure 4 The comparison of the saliency algorithm with different thresholds. Left: Top 30% for ridge points, bottom 30% for valley points; Middle: Top 20% for ridge points, bottom 25% for valley points; Right: Top 8%for ridge points, bottom 15% for valley points.

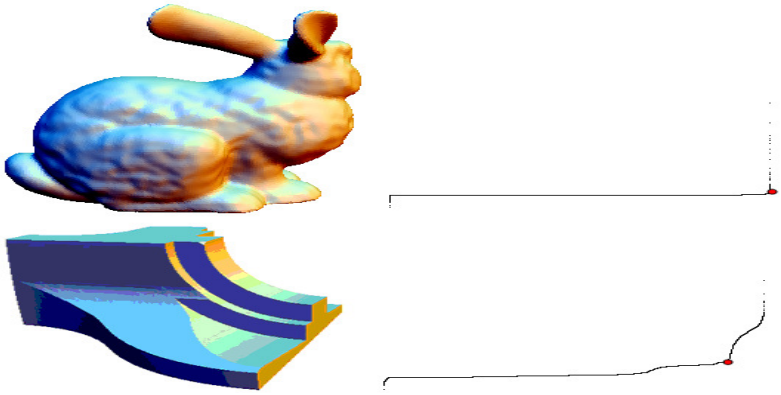

Figure 5 Sort the mesh saliency value of each point by the ascending order, we can get a plot: Firstly, the plot almost corresponds to a planar line, after arriving to a value, the plot rises steeply. Left: 3D model; Right: The plot corresponding to the mesh saliency values by the ascending order.

4.4 Generation of Feature Lines

Once the feature points have been detected, we need to connect them together. We follow the procedure proposed in [4] with an addition which can reduce the fragmentation of the feature lines:

1) Get the optimal threshold in section 4.3. Flag the vertex which saliency value is greater than the threshold T as feature points set M_1. Flag the vertex which saliency value is less than T and greater than $0.8*T$ as weak feature points set M_2. Define k-neighbor of a point $\mathbf{p}$ as $N(\mathbf{p},k)$.

2) For a feature point $\mathbf{p}$, examine the point $\mathbf{q}$ in $N(\mathbf{p},1)$. If only one point $\mathbf{q} \in \mathbb{M}_1$, then connect it to $\mathbf{p}$.

3) If two or more points $\mathbf{q}_i \in \mathbb{M}_1, i = 1,...,n$ and $n \geq 2$, then we connected $\mathbf{p}$ to one of them by following the vertex $\mathbf{q}_i$ corresponding to the smaller dihedral angle with the orientation of the principal curvature.

4) No any point in M_1, but at least one point $\mathbf{q} \in \mathbb{M}_2$. If having $\mathbf{k} \in N(\mathbf{q},1)$ *and* $\mathbf{k} \in M_1$ and $\mathbf{k} \notin N(\mathbf{p},1)$, then connect $\mathbf{p}$ to $\mathbf{q}$ similarly following the rule in step 2 and step 3.

Repeat step 2 to step 4.

5. RESULTS

For evaluating the effectiveness of our automatic mesh saliency method, this section shows results of our algorithm and compares it to the user-specified threshold algorithm. All of our tests were run on a PC with Intel® Core™ 2 1.73.GHz processor and 1.0 GB of main memory.

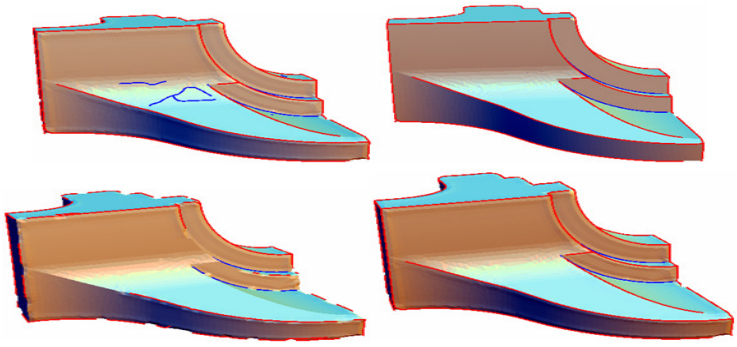

Figure 6 The comparison of the saliency algorithms with different thresholds and our automatic detection algorithm. Upper Left: Top 25% for ridge points, bottom 30% for valley points ; Upper Right: Top 15% for ridge points, bottom 20% for valley points ; Bottom Left: Top 5% for ridge points, bottom 5% for valley points; Bottom Right: The automatic detection algorithm.

Fig.2 and 3 show some results. Fig.2 shows the comparison of the detection algorithm by saliency values and the algorithm only by principal curvatures. Owing to its locality, the method only by principal curvature is sensitive to noise and irregularities of the triangulation and usually produces spurious feature lines. The result shows on the left of Fig.2. Mesh saliency that measure the region importance at multiple scales in the neighborhood can reveal the difference between the vertex and its surrounding context. So it can extract the most salient features points robustly. The result on the right shows that the lines are clearly detected by saliency values. In Fig. 3, we visualize the magnitude of principal curvature value and the saliency value of each point with color on a bunny model. Warm color corresponds to the sharp features and cool color corresponds

to the flat regions. We can find that saliency values differentiate the neck and the leg from their circumferences

.

Fig.4 and Fig.6 show the comparison between the different thresholds. The surface points are composed of feature points and flat points. One obvious way to extract the feature points on the surface is to select a threshold T , for example, TOP 20% means that the feature points are the top 20% points with high saliency values. But improper threshold produces many spurious feature lines. Polygonal surface models can be roughly divided into two classes: CAD-like models showed in Fig.6, which usually have large flat regions; Non-CAD-like model showed in Fig.4, which have many fine details. We can't find a unified threshold for the detection method. So how to decide the threshold is a problem for user-specified threshold methods.

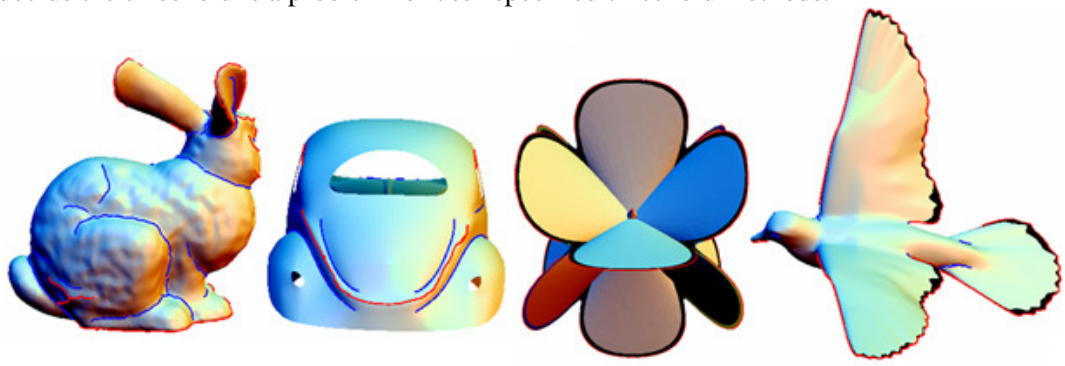

Figure 7 Our automatic algorithm for feature line detection on various models.

Fig.5 gives an analysis for the optimal threshold. On the left are the 3D models, on the right are the corresponding plots of the saliency values in ascending order. The plot begins to rise slowly and almost corresponds to a planar line, after arriving to a value (Flagged in red circle dot), the plot rises steeply. The value at the red circle dot is the threshold we need. Fig.6 shows results from the fandisk CAD model, in which our automatic method works well. Moreover, our method doesn't need a user to select the threshold. It is not easy for a user to modify the threshold if no any apriori knowledge. Fig. 7 illustrates further results on bunny, car, focal and dove model. Ohtake's method [2, 22] is a reliable detection of ridge-valley structures on surfaces approximated by dense triangle meshes and has become a representative method of feature lines detection algorithm.. Fig.8 shows that our automatic method almost has the same precise as Ohtake's method.

6. CONCLUSION

This paper has presented an automatic algorithm for the detection of feature lines on triangular meshes based on the concept of salient geometric features. The utility of saliency values for robustly detecting the feature lines on surface has been demonstrated. The results show saliency values effectively capture important shape features.

Our automatic algorithm is a fully automatic method. It can advantageously select a "good" threshold without any user intervention. The results show that our automatic algorithm can find an optimal threshold to detect the feature lines on 3D meshes. In the future we intend to develop data clustering method into the field of the feature lines detection.

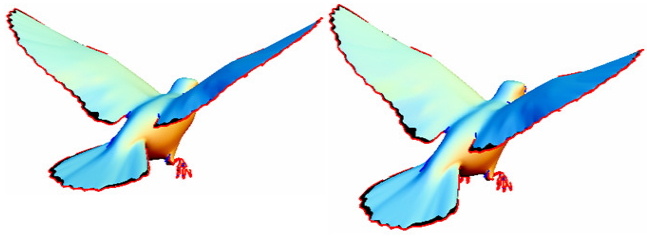

Figure 8 Our automatic algorithm VS. Ohtake's method, left: Our automatic algorithm; right: Ohtake's method.

ACKNOWLEDGEMENT

This study was supported by the Innovation projects of Department of Education of Guangdong Province, China (NO.2017KTSCX183) and the introduction project of Jiangmen innovative research team(2018).

REFERENCES

[1] Forrester Cole, Kevin Sanik, Doug Decarlo, Adam Finkelstein, Thomas Funkhouser, Szymon Rusinkiewicz & Manish Singh, (2009) "How Well Do Line Drawings Depict Shape?", ACM Transaction on Graphics, Vol. 28, No.3, pp43-51.

[2] Ohtake Y., Belyaev A., & Seidel H.P, (2004) "Ridge-valley Lines on Meshes via Implicit Surface Fitting", ACM Transactions on Graphics, Vol. 23, No. 3, pp609-612.

[3] Shin Yoshizawa, Alexander Belyaev & Hans-Perter Seidel, (2005) "Fast and Robust Detection of Crest Lines on Meshes", Symposium on Solid and Physical Modeling'05, pp227-232.

[4] Soo-Kyun Kim & Chang-Hun Kim, (2006) "Finding Ridges and Valleys in A Discrete Surface Using A Modified MLS Approximation", Computer-Aided Design, Vol. 38, No.2, pp173-180.

[5] Georgios Stylianou & Gerald Farin, (2004) "Crest Lines for Surface Segmentation and Flattening", IEEE Transaction on Visualization and Computer Graphics, Vol. 10, No. 5, pp536-543.

[6] Tilke Judd, Fredo Durand & Edward H. Adelson, (2007) " Apparent ridges for line drawing" , ACM Transactions on Graphics, Vol. 26, No. 3, pp19-26.

[7] Chang Ha Lee, Amitabh Varshney & David W.Jacobs, (2005) "Mesh Saliency". Proceedings of ACM Siggraph'05, pp659-666.

[8] Ran Gal & Daniel Cohen-Or, (2006) "Salient Geometric Features for Partial Shape Matching and Similarity", ACM Transactions on Graphics, Vol. 25, No. 1, pp130-150.

[9] Taubin G, (1995) "Estimating the Tensor of Curvature of a Surface from a Polyhedral Approximation", In Proceedings of Fifth International Conference on Computer Vision'95, pp902-907.

[10] Sachin Nigam & Vandana Agrawal, (2013) " A Review: Curvature approximation on triangular meshes", Int. J. of Engineering science and Innovative Technology, Vol. 2, No. 3, pp330-339.

[11] Xunnian Yang & Jiamin Zheng, (2013) "Curvature tensor computation by piecewise surface interpolation", Computer Aided Design, Vol. 45, No. 12, pp1639-1650.

[12] Gady Agam & Xiaojing Tang, (2005) "A Sampling Framework for Accurate Curvature Estimation in Discrete Surfaces", IEEE Transaction on Visu alization and Computer Graphics, Vol. 11, No. 5, pp573-582.

[13] Meyer M., Desbrun M., Schroder P. & Barr A. H, (2003) "Discrete Differential-geometry Operators for Triangulated 2-manifolds", In Visualization and Mathematics III' 03, pp35-57.

[14] Stupariu & Mihai-Sorin, (2016) "An application of triangle mesh models in detecting patterns of vegetation", WSCG' 2016, pp87-90.

[15] Chen L., Xie X., Fan X., Ma W., Zhang H., & Zhou H, (2003) "A visual attention model for adapting images on small displays", ACM Multimedia Systems Journal, Vol. 9, No. 4, pp353-364.

[16] Lee, Y., Markosian, L., Lee, S., & Hughes, J. F, (2007) "Line drawings via abstracted shading", ACM Transactions on Graphics, Vol. 26, No. 3, pp1-9.

[17] Jack Szu-Shen & His-Yung FEng, (2017) "Idealization of scanning-derived triangle mesh models of prismatic engineering parts", International Journal on Interactive Design and Manufacturing, Vol. 11, No. 2, pp205-221.

[18] Decarlo D., Finkelstein A., Rusinkiewicz S. & Santella A,(2003) "Suggestive Contours for Conveying Shape", ACM Transactions on Graphics, Vol.22, No. 3, pp848-855.

[19] M. Kolomenkin, I. Shimshoni, & A. Tal,(2008) "Demarcating curves for shape illustration", ACM Transactions on Graphics, Vol.27, No.5, pp157-166.

[20] Michael Kolomenkin,(2009) "Ilan Shimshoni and Ayellet Tal. On Edge Detection on Surface", IEEE CVPR' 09, pp2767-2774.

[21] M. P. Do Carmo (2004) Differential geometry of curves and surfaces, Book, China Machine Press.

[22] A. Belyaev, P.-A. Fayolle, & A. Pasko, (2013) "Signed Lp-distance fields", CAD, Vol.45, No. 2, pp523-528.

[23] Y Zhang, G Geng, X Wei, S Zhang & S Li, (2016) "A statistical approach for extraction of feature lines from point clouds" ,Computers & Graphics, Vol. 56, No. 3, pp31-45.