I.J. Information Engineering and Electronic Business, 2011, 3, 1-7
Published Online June 2011 in MECS (http://www.mecs-press.org/)

Extracting Feature Curves on Point Sets

X. F. Pang
1Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen, China
2The Chinese University of Hong Kong, Hong Kong, China
Email: xf.pang@siat.ac.cn

M. Y. Pang
Department of Educational Technology, Nanjing Normal University, Nanjing, China
Email: panion@netease.com

Z. Song
1Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, Shenzhen, China
2The Chinese University of Hong Kong, Hong Kong, China
Email: zhan.song@ siat.ac.cn

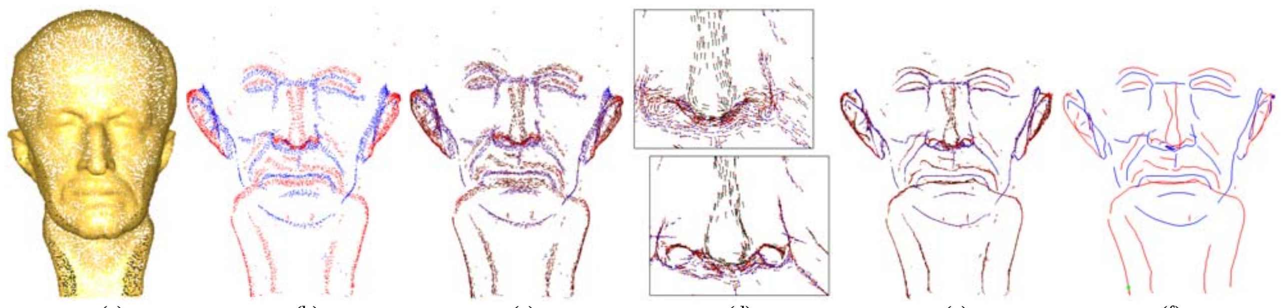


Figure 1. Figure 1. Over view of our feature extraction pipeline: 1(a) original point cloud with computed normals; 1(b) potential feature points identified to be close to possible features; (c) principal directions with potential feature points (d) top: the zoon in part of potential feature points on nose; bottom: the zoon in part of the smoothed points on nose which presented in the next stage. (e) smoothed feature points; (f) reconstructed polylines that identify the sharp features.

***Abstract*—We present an effective algorithm for detecting feature curves on point sets. Based on the local surface fitting method, our algorithm first compute the curvatures and principal directions of each point of point sets. The algorithm then extracts potential feature points according to the biggest principal curvature of the point, and evaluates the principal directions of the detected points. By projecting the points onto the principal axes of their neighborhoods, the potential feature points are smoothed. Using the principal directions with each optimized point, feature curves are generated by polyline growing along the principal directions of feature points. The results indicate that our algorithm is sensitive to both sharp and smooth feature curves of point set, and it supports multi-resolution extraction of features.**

***Index Terms*—ridges, valleys, valley-ridge extraction, MLS surface fitting, feature curves**

I. Introduction

Multiple techniques have investigated the identification of feature curves on point set models and polygonal models. The previous work on feature lines extraction can be roughly divided into three categories: normal deviation based, covariance analysis based, projection procedure based.

Normal deviation based method proposed in paper [1] first estimates the normal vectors by the PCA analysis of these 1-ring neighbors, as explained in [2], and then the segmentation is performed on point cloud based on the variation of the normals. Once the connected graph where each edge connects two segments is constructed, the feature curves can be obtained after pruning and smoothing the graph.

Covariance analysis based methods [3] [4] classify the points of point cloud according to the likelihood or unlikelihood that they belong to a feature, which is computed from principal component analysis on local neighborhoods. In paper [3], by changing the local neighbor size, multi-scale estimation is achieved. Both [3] and [4] build minimal spanning trees over the potential feature points, and then curve lines are computed to appropriate the features. However, the above methods require high quality of point cloud, and their sensitivity to the smooth features is very limited.

Projection procedure based algorithm estimates the feature curves by approximating the point set surface. In paper [5], a robust algorithm that identifies sharp features in a point cloud by returning a set of smooth curves aligned along the edges. This feature extraction is a multi-step refinement method that leverages the concept of Robust Moving Least Squares (RMLS) [6] to locally fit surfaces to potential features. By projecting the points to the intersections of multiple surfaces, polylines are expanded along the projected points. After resolving gaps, connecting corners, and relaxing the results, the algorithm returns a set of complete and smooth curves that define the features. Since the RMLS is computationally expensive, this algorithm is limited by time constraints.

In this paper, we focus on extracting of valley-ridge lines from scatted point sets. The proposed algorithm falls into the third category of feature extraction algorithms, since it's based on a projection procedure for local fitting. After approximating the neighborhoods of each point with a local MLS polynomial [7], the curvatures and curvature tensors are computed based on the first and second fundamental forms [8][9] of the surface. Points whose absolute value of principal curvatures is bigger than a threshold are defined as a potential feature point. By performing principal component analysis over the potential feature points, they are smoothed. Then feature curves are achieved by growing along the principal directions of the smoothed points.

The point clouds used in our experiments may contain connection information in originals, we only use their coordinate information.

Given an unorganized point cloud without normal and any connectivity information

$$P=\{p_i\},\ p_i\in R^3, i\in\{1,...,N\},$$

the valley-ridge curves on point cloud are achieved by performing the following four steps, see Figure 1.

(1) Approximating the neighborhoods of each point with a local polynomial, the principal curvatures of each point can be computed. Then the potential feature points are extracted according to the curvature values of each point.

(2) Based on Fundamental forms of fitting surface, the principal directions for each potential point are computed.

(3) Using cross-correlation coefficient analysis of local points and principal component analysis, we smooth the projected points.

(4) Create initial sets of valleys and ridges by growing polylines along the principal directions of the smoothed points, and then analyze the end-points of feature polylines to complete gaps.

In the following sections, we'll discuss each of the four steps of the algorithm, in details.

II. Potential Valley-ridge points extraction

A. MLS fitting

The proposed algorithm employs moving least square (MLS) method to fit a smooth patch for ω radius neighborhood of each point,

$NBHD(p_i)=\{p_j\},\|p_j-p_i\|\le\omega, j=0,...,k$, which works as follows.

Build a covariance matrix B of $NBHD(p_i)$,

$$B=\sum_{p_j\in NBDH(p_i)}(p_j-o_i)(p_j-o_i)^T.$$

where

$$o_i=\tfrac{1}{k}\sum_{j=1}^{k}p_j.$$

Calculating the eigenvalues and their corresponding eigenvectors of B, a reference domain H_i is defined by eigenvectors which are corresponding to the biggest eigenvalue and the second biggest eigenvalue, respectively. The surface normal n_i of H_i is set to be the eigenvector corresponding to the smallest eigenvalue, $\|n_i\|=1$. Then build a local coordinate system on H_i, o_i is the origin and n_i is the Z-axis.

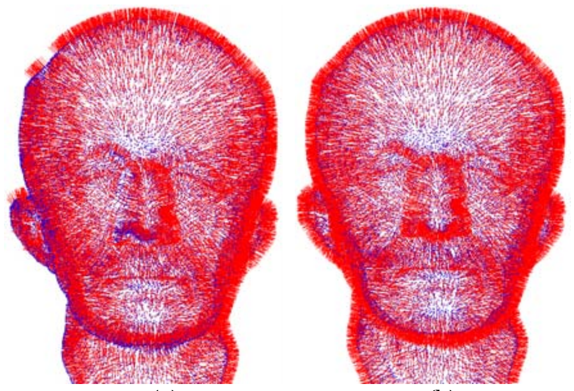
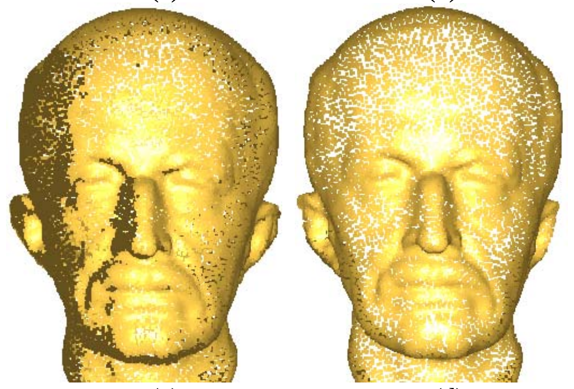

(a) (b)

(c) (d)

Figure 2. Normals of Maxplanck model before adjustment and after adjustment. (a) Normals before adjustment (b) Normals after adjustment (c) point cloud with unadjusted normals (d) points set with adjusted normals.

The experiment shows that the Z-axes of local coordinate frames can point to the different sides of point set 'surface', see Figure 2(a). For the purpose of obtaining right valley-ridge information, consistently outward normal directions are adjusted by applying normal adjustment operation [10]. And each $+Z$ axis of local coordinate system has the same direction as the adjusted n_i, see Figure 2(b). Then $NBHD(p_i)$ is approximated with a local MLS polynomial g_i which satisfies (1).

$$\min\sum_{p_j\in NBHD(p_i)}<p_j-p_g,n_i>^2 \qquad (1)$$

p_g is the projection of p_j onto g_i along n_i. g_i is a polynomial in local coordinate (o_i,u,v,n_i), i.e.

$$g_i(u,v) = a + bu + cv + duv + eu^2 + fv^2, (u,v) \in [-h,h]^2 \quad (2)$$

Formula (1) can be changed to

$$\min \sum_{p_j \in NBHD(p_i)} ((p_j - o_i)\cdot n_i - g(u_j,v_j))^2 \quad (3)$$

here, (u_j, v_j) is the representation of p_j about a local coordinate system in H_i. For any point q in H_i, its projection on g_i can be calculated out as $o_i + (u_q, v_q, g(u_q, v_q))$.

Since the resulted MLS approximation relies on the number of neighbor points used in local fitting, we set the fitting radius $\omega = 2.0 * \kappa$, where κ denote the average station distance of the point cloud.

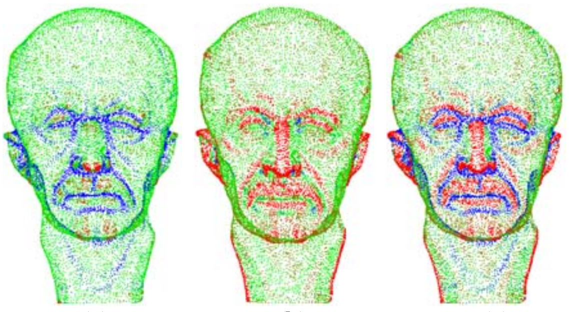

(a) (b) (c)

Figure 3. Point set marked according to the value of principal curvatures. (a) Maximal principal curvatures. (b) Minimal principal curvatures. (c) The biggish principal curvature values.

B. *Curvatures computation*

Since the local fitting g_i has been achieved, curvatures of each point p_i on point set surface can be calculated based on computing the first and second fundamental quality of the surface [8] [9]: L, M, N, E, F, G. The main curvatures of p_i are denoted as $K_{\max}, K_{\min}$. Based on the analysis of the experiment results, we found that better results are achieved by using biggish principal curvature, see Figure 3.

C. *Potential feature points detection*

k_i is the biggish principal curvature of p_i, τ is a user defined threshold which related to the average absolute curvature of the whole point set. If $k_i < 0$ and $|k_i| > \tau$, p_i is added into valley-point set V, if $k_i > \tau$, p_i is added into ridge-point set R, see Figure 1(b).

As a matter of convenience, we use F representing V or R in the following paper.

III. Computing principal direction of minimum curvature

Based on the fundamental forms of fitting surface, the principal directions responding to $K_{\max}, K_{\min}$ can also be calculated as follows:

$$T_{\max} = ((-1)*(N - K_{\max}G), M - K_{\max}F)$$

$$T_{\min} = ((-1)*(N - K_{\min}G), M - K_{\min}F),$$

By computing the principal directions of each point in point set, curvature tensor on point set surface can be determined, as shown in Figure 4. Since $T_{\max}$ and $T_{\min}$ are everywhere orthogonal (expecting for umbilical region), it only needs to compute one of the curvature direction fields, and the other direction field can be achieved by computing its gradient field.

In our feature curve extraction process, to speed up the whole procedure, only principal directions of detected potential feature points are calculated, as seen in Figure 1(c).

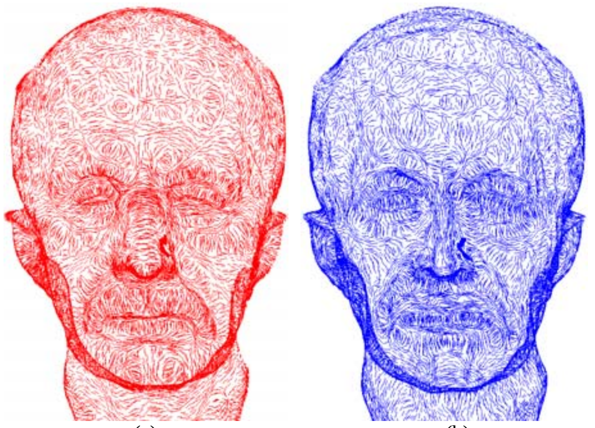

(a) (b)

Figure 4. Principal directions $T_{\max}$, $T_{\min}$ estimated on point set surface. (a) Maximal principal directions. (b) Minimal principal directions.

IV. Smoothing the feature points

The feature points are smoothed by projecting them onto their principal axis of neighborhoods, so the over all smoothing performance is determined by the neighborhood size δ, as seen in Figure 5. In order to obtain stable smoothing effect, adaptive neighborhood growth scheme, which based on a statistical analysis of point distributions on the approximate plane, is adopted [5].

For each potential feature point $p_i \in F$, the covariance matrix C of the neighborhood

$$NBHD(p_i) = \{p_j \mid (\| p_j - p_i \| \le \delta, j = 0,...,k)\}$$

is computed,

$$C = (p_1 - \bar{p}, ..., p_k - \bar{p}) \cdot (p_1 - \bar{p}, ..., p_k - \bar{p})^T$$

where

$$\bar{p} = \frac{1}{k}\sum_{j=1}^{k} p_j .$$

Calculate the eigenvalues and their corresponding eigenvectors, let $\lambda_0 \ge \lambda_1 \ge \lambda_2$, then V_0, V_1, V_2 are corresponding eigenvectors of $\lambda_0, \lambda_1, \lambda_2$. Project feature points onto the plane defined by two eigenvectors V_0, V_1 and $\bar{p}$. Then the projected points are translated into the local coordinate, and the projected points are interpreted in terms of a distribution of two random variables, X and Y, which can be analyzed by cross-correlation coefficient defined by

$$\rho_{XY} = \frac{\mathrm{cov}(X,Y)}{\sqrt{D(X)}\sqrt{D(Y)}}$$

where, ρ_{XY} has value between -1 and +1 representing the degree of linear dependence between X and Y, meaning that the points in $NBHD(p_i)$ are distributed along a feature line.

δ will be increased if the cross-correlation coefficient ρ over points in δ neighborhood is not sufficiently large enough, for our models we use $\rho \geq 0.6$. If δ is bigger than a predefined threshold, the point p_i will be abandoned as outlier or noise.

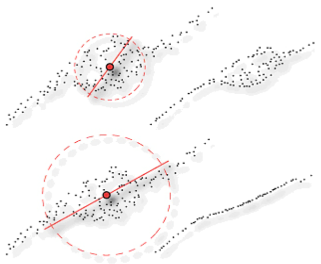


Figure 5. Smoothing result in different neighborhood size [5] [11].

After getting a suitable δ, the principle axis of new $NBHD(p_i)$ is obtained by employing PCA operation, and then a smoothed point $\dot{p}_i$ is yielded by projecting p_i onto the principle axis.

The smoothed points are added into the point set of $F(S)$.

V. Feature Polyline Propagation

Feature polylines of valleys or ridges are respectively generated by growing them along the maximal principal curvature directions of $V(S)$ and minimal principal curvature directions of $R(S)$.

A. *Growing of a feature curve*

For a new line of feature curve, seed point is traced in each principal direction D as:

$$D \in \{T_{\max}, -T_{\max}\} or \{T_{\min}, -T_{\min}\}$$

Suppose $L = \{l_1, ..., l_n\}$ be the current generating curve, where l_k is the k-th sample point. The new point on current curve is generated as:

$$l_k = l_{k-1} + s \cdot d_{k-1}$$

where s indicates a user-specified step size, and d_{k-1} is the principal direction of l_{k-1}. Since l_k probably not exactly belongs to the point set surface, it needs to be optimized. The neighbor points in radius r are collected and the barycenter p_c is computed. Arranging the concerned principal directions of neighbor points as the same sign with the direction vector of l_{k-1}, the principal direction of p_c is computed as the unity vector of the neighbor's, which normalized and denoted as T_c. Then l_k is set to be the barycenter p_c and its principal directions is set to be T_c.

Once the new updated sampled point is obtained, the neighbors of the previous sampled point in radius s are tagged and will not be concerned in the following tracing operation. However, for the starting seed point, not all its neighbors in radius are tagged, but only the points in the semi-sphere of the current tracing direction, as shown in Figure. 6.

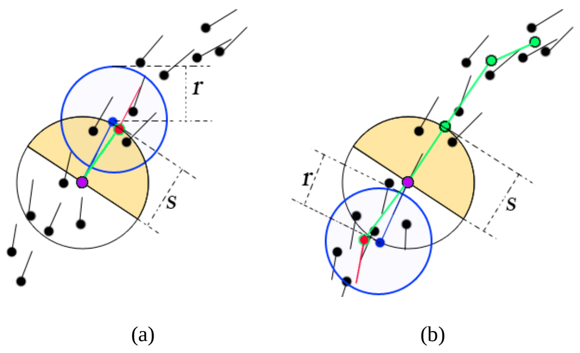


Figure 6. The lines of curvature tracing process. The Purple point is the starting seed, the blue point is the new sampled point, and the red point is the optimized new sampled point. When a new sampled point is detected, the neighborhoods of the previous sampled point are flagged. (a) The tracing process in one direction of the seed point. (b) The tracing process in the other direction of the seed point.

Each current feature curve stops, if any one of the following conditions is satisfied:

(1) If the neighbors of current sampled point in radius r has been tagged.

(2) If a feature line reaches the boundary of the point set, i.e., there are no neighbors of the current sampled point in radius r.

The two parameters (s and r) used in the tracing process are both related to the average station distance of the points cloud κ. Our experiments indicate that $s = 1.2 * \kappa$ and $r = 0.8 * \kappa$ lead to good results.

B. *Lines of valleys and ridges extracton*

The feature curves of valleys and ridges are extracted independently. And the sampled points of feature curves are separately added into the point set $R(L)$ and $V(L)$. Since the ridges and valleys turn into each other as surface orientation is changed, without loss of generality, only the lines of ridge curves are considered, which are extracted by performing the following steps:

(1) All points in $R(S)$ are added into a priority queue, the points with higher ratio between maximal principal curvature and minimal principal curvature are assigned higher priority.

(2) Pop a new seed point $\boldsymbol{p_i}$ from the priority queue and add it into $R(L)$ as l_0. Then a tracing procedure is separately started from l_0 in two different directions $T_{\min}$, $-T_{\min}$.

(3) During the tracing process, for each new sampled point l_{k+1}, if none of the stopping conditions are satisfied, the new sampled point l_{k+1} is added into $R(L)$. And then the local neighbors of the previous sampled point l_k in radius s are flagged and removed from the priority queue. Otherwise, the tracing operator of the current direction needs to be stopped.

By repeating steps (2) and (3) until the priority queue is empty, the feature curves of ridges are obtained, as shown in Figure 1(f).

The feature curves of valleys can be calculated based on the same scheme as tracing feature curves of ridges.

C. *Optimization of Feature Curves*

Then gaps between polylines which are caused by poor sample quality are completed by connecting each end-point of feature polyline to other feature points within the cone formed by the tangent vector T at the end-point and a predefined aperture angle μ [5].

Finally, we smooth polylines by applying uniformity scheme on interior points for 1 or 2 passes. Given a feature polyline $R=\{q_0,...,q_k\}$, firstly fix q_0 and q_2, then do the following work on the plane which defined by (q_0,q_1,q_2): move q_1 to the vertical of the line segment (q_0,q_2) with no change of the area of triangle $\Delta q_0,q_1,q_2$. Repeat this until all internal vertices are adjusted, see Figure 7.

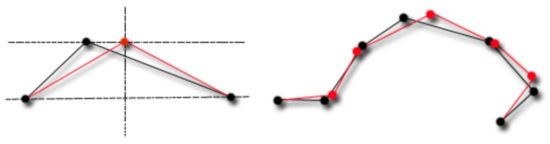


Figure 7. Smoothing feature polylines; fix the end-points of the polylines and adjust the internal vertices with restriction that the area of any triangle enclosed by three contiguous points keeping invariant on their own defined plane.

As seen in Figure 8, the valley and ridge feature curves are extracted from Sper-dragon models.

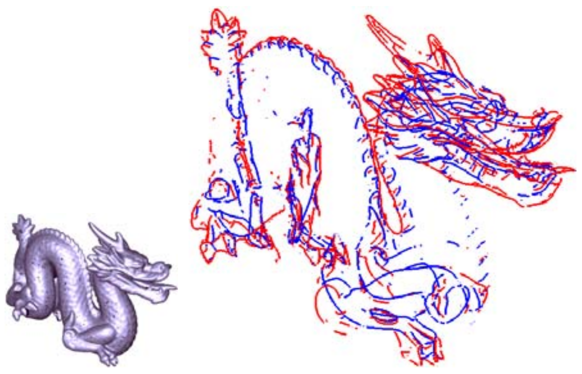


Figure 8. Valley-ridge feature curves on Sper-dragon model, blue lines denote valleys and red lines denote ridge curves.

VI. Experiments Results

The performance of the proposed method to its run time efficiency and robustness to noise is discussed in this section. The proposed algorithm carried out on a 2.7GHz AMD Athlon 2.7GHz processor with 2GB memory.

A. *Run Time Efficiency*

Since the experimental data used in our algorithm are raw point clouds without any normals and connection information, the preprocessing operators including normal computation and orientation arrangements are essential parts of the feature curve extraction. The moving least squares surface fitting is a fundamental operation for detecting potential feature points and computing principal directions of each feature points. Smoothing potential feature points and generating feature curves are the last two key procedures for the proposed algorithm.

As shown in Table I, we technically divide the whole running time of the algorithm into four parts: (1) cost for pre-processing including normal calculation and average station distance. (2) Cost for MLS fitting, curvature estimation, and also includes the cost of detecting potential feature points. (3) Smoothing feature points, and feature curve propagation and completion. The experiments indicate that the most time-consuming part of our implementation is the preprocessing (1) and MLS surface fitting (2). Since the time required to construct the polylines of feature curves is determined by the number of potential feature points in point cloud, the cost of (3) increase when models have complicated details.

B. *Robustness to Noise*

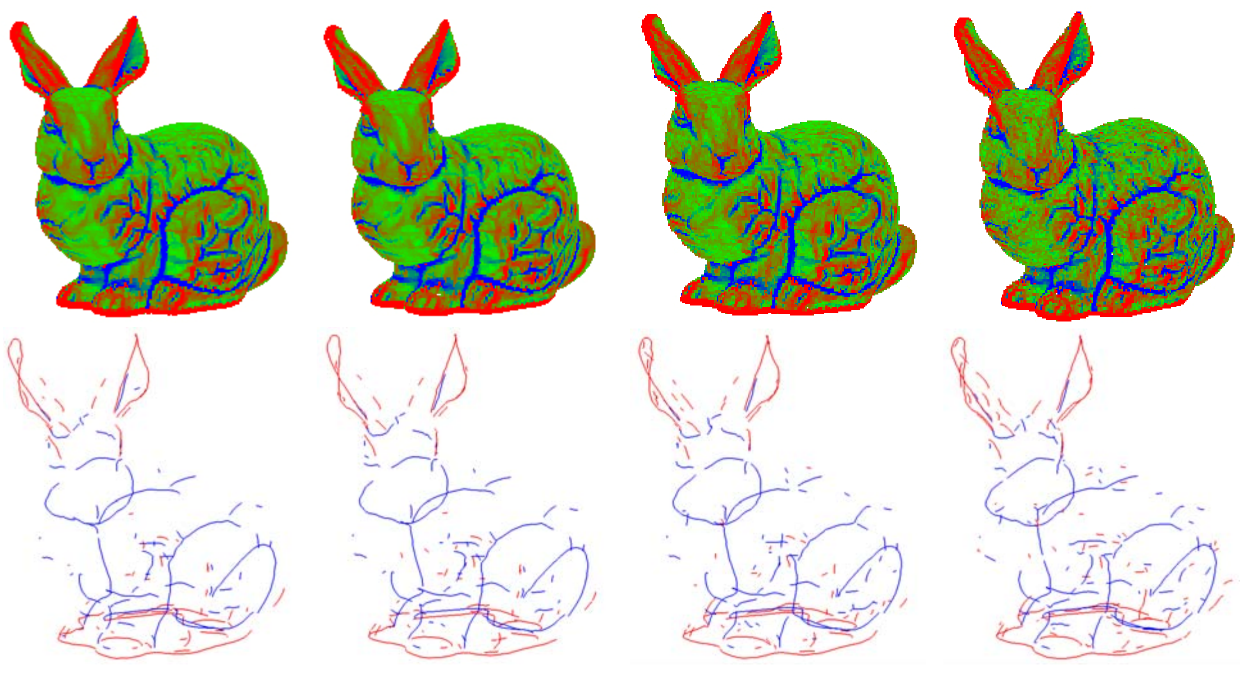
To validate the robustness of the extraction of feature curve in handing noisy point sets, three sets of noisy data are generated by adding Gaussian noise with different amplitudes of standard deviations (i.e. 0.0001 and 0.001) to each point cloud model. Table II shows that an increase in the noise level leads to a slight increase in the computational costs, which can be explained by the fact that, for models with higher noise level, the same curvature threshold and MLS radius leads more potential feature points, which consequently increase the cost of (3). The graphic illustrations in Figure 9 also indicate that the proposed algorithm is very stable against noise.

TABLE I.
THE RUN TIME PERFORMANCE OF THE PROPOSED ALGORITHM WHICH COMPUTED WITH $\omega=2.0\kappa$, $\tau=16$. [$\kappa=$ AVERAGE STATION DISTANCE, $\omega=$ MLS RADIUS, $\tau=$ CURVATURE THRESHOLD FOR DETECTING FEATURE POINTS]

Model	Scale	κ	Feature	Run time (ms)			
	[Point]	$[10^{-2}]$	Points	(1)	(2)	(3)	Total
Maxplanck	25445	1.6130	5771	1046	625	204	1875
Bunny	35947	1.7479	4434	1437	860	156	2453
Venus	134359	0.9094	25215	5141	3359	1688	10188
Balljoint	137073	0.6924	46911	1985	3718	4954	10657

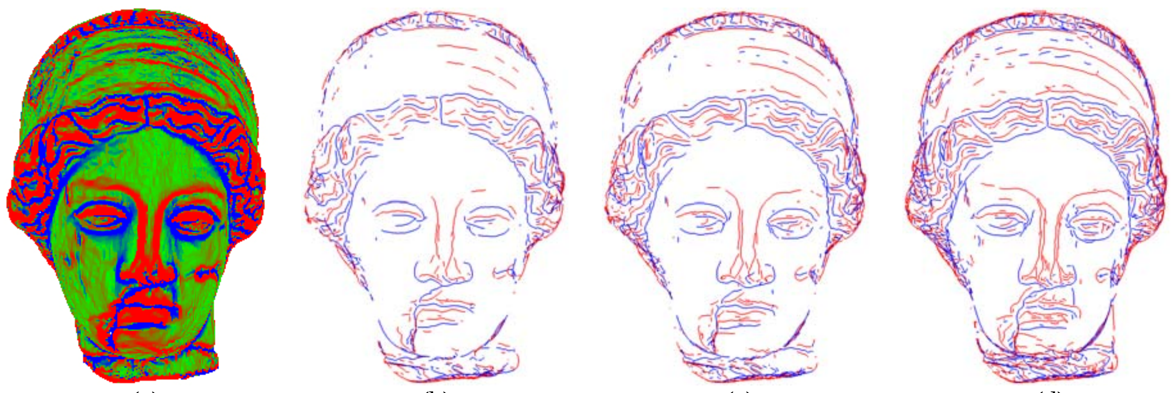
TABLE II.
THE RUN TIME PERFORMANCE OF THE PROPOSED ALGORITHM PERFORMING ON DIFFERENT MODELS AT VARYING NOISE LEVELS WITH $\omega=2.0\kappa$, $\tau=16$. [κ = AVERAGE STATION DISTANCE, ω=MLS RADIUS, τ = CURVATURE THRESHOLD FOR DETECTING FEATURE POINTS]

Model	Scale [Point]	Noise	κ [10^{-2}]	ω [κ]	τ	Feature Points	Run time (ms)			
							(1)	(2)	(3)	Total
Maxplanck	25445	0.0001	1.6134	2.0	16	5803	1047	625	234	1906
		0.0005	1.6131	2.0	16	5972	1047	625	235	1907
		0.001	1.6113	2.0	16	6597	1047	609	281	1937
Bunny	35947	0.0001	1.7468	2.0	16	4452	1438	859	141	2438
		0.0005	1.7455	2.0	16	4508	1438	859	141	2438
		0.001	1.7426	2.0	16	4802	1438	859	172	2469
Venus	134359	0.0001	0.9090	2.0	16	25274	5094	3359	1641	10094
		0.0005	0.9094	2.0	16	30153	5125	3344	2359	10828
		0.001	0.9063	2.0	16	30153	5125	3328	2407	10860



(a) 0.0 (b) 0.0001 (c) 0.0005 (d) 0.001

Figure 9. The feature curves extracted on bunny models witch is added with different level of Gaussian noise. Blue lines denote valleys and red lines denote ridges. The bunny models with different noise are color marked according to the biggish principal curvatures on the top line. The feature curves are respectively depicted on the bottom line. (a) Models with no noise. (b) Models with Gaussian noise of 0.0001. (c) Models with Gaussian noise of 0.0005. (d) Models with Gaussian noise of 0.001.



(a) (b) (c) (d)

Figure 10. Multi-level valley-ridges curves extraction on Venus models, where blue lines indicate valleys and red lines indicated ridges. (a) Venus model colored according to the biggish principal curvatures. (b) Features computed with curvature threshold $\tau=16$. (c) $\tau=13$. (d) $\tau=10$.

C. Multi-resolution Extraction and Error analysis

Multi-resolution valley-ridge features are achieved by applying different radius in local surface fitting, and using different curvature threshold in valley-ridge point detection. As seen in Figure 10, multi-resolution feature curves are achieved on Venus models computed with $\omega = 2.0\kappa$ and with varying curvature threshold of $\tau = 16$, $\tau = 13$, $\tau = 10$.

The valley-ridge feature points extracted by our algorithm may not be the points in point set, but the points projected onto the local MLS fit, so the error of our implication is the error of MLS surface fitting [12].

VII. Conclution and Future Works

This paper present a new algorithm for extracting valley-ridge curves from raw point cloud, the experiments indicate that the proposed algorithm produces viable results on irregular point clouds (see Figure 1) and point data with different level of noise(see Figure 8, 9) in high efficiency, and meanwhile, the multi-resolution feature curve can also be achieved (see Figure 10).

As future work, we plan on improving the computing efficiency of detecting potential feature points, and investigating the smoothing and sampling methods to preserve both sharp features and smooth features.

Acknowledgment

The work in this paper was supported by National Natural Science Foundation of China (Grant No. 61002040 and Grant No. 60873175).

References

[1] K. Demarsin, D. Vanderstraeten, T. Volodine, D. Roose, “Detection of closed sharp feature lines in point clouds for reverse engineering applications”, Report TW 458, Department of Computer Science, K.U. Leuven, Belgium, 2006.

[2] K. Hormann, “Theory and Applications of Parameterizing Triangulations”, PhD thesis, Department of Computer Science, University of Erlangen, 2001.

[3] M. Pauly, R Keiser, M.Gross, “Multi-scale feature extraction on point-sampled surfaces”, *In: Proceeding of Eurographics.* ETH Zurich, Sweden, Blackwell, vol. 22, no. 3, pp. 281-290, 2003.

[4] S. Gumhold, X. Wang, R. McLeod, “Feature Extraction from Point Clouds”, *In: Proceedings of the 10th international meshing roundtable.* Newport Beach, California, Springer, pp. 293-305, 2001.

[5] I. J. Daniels, L. K. Ha, T. Ochotta, C. T. Silva, “Robust smooth feature extraction from point clouds”, *In: Proceedings of IEEE International Conference on Shape Modeling and Applications.* Lyon, France, IEEE Computer Society Press, 2007: 123-133.

[6] S. Fleishman, O. D. Cohen, C. T. Silva, “Robust moving least-squares fitting with sharp features”, *In: Proceeding of ACM Transactions on Graphics,* Los Angeles, USA, New York: ACM Press, vol. 24, no. 3, pp. 544-552, 2005.

[7] G. Gaël, G. Markus, “Algebraic point set surfaces”, *In: ACM SIGGRAPH 2007,* New York: ACM Press, pp. 23-31, 2007.

[8] T. Gatzke and C. M. Grimm, “Estimating curvature on triangular meshes”, *International Journal of Shape Modeling,* vol. 12, no. 1, pp.1-28, 2006.

[9] J. Goldfeather and V. Interrante, “A novel cubic-order algorithm for approximating principal direction vectors”, *Transaction on Graphics,* vol. 23, no. 1, pp. 45-63, 2004.

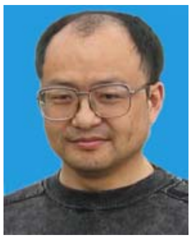
[10] H. Hoppe, T DeRose, T Duchamp, et al, “Surface reconstruction from unorganized points”, *In: Proceeding of Siggraph’92.* vol. 26, no. 2, pp. 71-78, 1992.

[11] I. K. Lee, “Curve reconstruction from unorganized points”, *Computer Aided Geometric Design,* vol. 17, no. 2, pp. 161-177, 2000.

[12] D. Levin, “The approximation power of moving least-squares”, *Mathematics of computation.* Vol. 67, no. 224, pp. 1517-1531, 1998.

X.F. Pang was born in Hunan, China in 1984. She received master degree in educational technology in Nanjing normal university, Nanjing, in 2010.

She is currently with the Shenzhen Institutes of Advanced Technology (SIAT), Chinese Academy of Sciences (CAS), as a research assistant. Her current research interests include computer graphics and Digital Geometry Processing.



M. Y. Pang was born in Anhui, China in 1968. He received his master and Ph.D. degrees from Institute of Computer Graphics Technology of Jiangsu University in 2002 and 2004, respectively.

He is associate professor at Department of Educational Technology, Nanjing Normal University. He ever worked as a postdoc in the State Key Lab of Novel Software Technology, Nanjing University, China. His research interest covers digital geometry processing, visualization of dynamic systems, and computer-aided geometric design.

Z. SONG was born in Shandong, China, in 1978. He received Ph.D. in Mechanical and Automation Engineering from the Chinese University of Hong Kong, Hong Kong, in 2008.

He is currently with the Shenzhen Institutes of Advanced Technology (SIAT), Chinese Academy of Sciences (CAS), as an assistant researcher. His current research interests include computer graphics, structured-light based sensing, image processing, 3-D face recognition, and human–computer interaction.