



## Extraction of Surface-feature Lines on Meshes Using Normal Tensor Framework

Shoichi Tsuchie<sup>1</sup> and Masatake Higashi<sup>2</sup>

<sup>1</sup>Nihon Unisys, Ltd., shoichi.tsuchie@unisys.co.jp

<sup>2</sup>Toyota Technological Institute, higashi@toyota-ti.ac.jp

### ABSTRACT

This paper describes a novel method for extracting surface-feature lines, which are curves on surfaces and become  $C^1$  boundary curves as fillet edges and boundaries between convex and concave regions as well as sharp edges as  $C^0$  boundaries, on meshes. By detecting them, we can proceed high-quality mesh geometry processing such as feature-preserving mesh denoising and simplification, and underlying surface extraction (i.e., segmentation) required for the reverse engineering. Our method is based on the normal tensor framework, and detects both  $C^0$  and  $C^1$  boundary vertices according to a change of curvature values along the principal directions. Furthermore, our method can be applied to noisy scanned data by performing anisotropic smoothing for the normal tensor. We demonstrate effectiveness of our method by applying it to some CAD models and real-world scanned data.

**Keywords:** surface-feature line, fillet edge, normal tensor, tensor smoothing.

### 1. INTRODUCTION

Surface-feature lines are curves on surfaces, which are important for characterizing their shapes. They are lines which become  $C^1$  boundary curves as fillet edges and boundaries between convex and concave regions as well as sharp edges as  $C^0$  boundaries. In geometry processing as mesh denoising or simplification, feature lines should be preserved and may become a part of region boundaries in mesh segmentation. Various studies as [7,10,17,23,26] and also references therein have proposed to extract the feature lines which correspond to ridges/valleys (also called *crest lines*) defined on a smooth surface as the locus of points where the maximum/minimum principal curvatures take a positive-maximum/ negative-minimum along its curvature direction [6].

*Crest line* becomes useful in various applications such as CG models and 3-D medical imaging, but it is insufficient for characterizing a shape in the industrial/mechanical objects as shown in Fig. 1(left). In those models, the high-curvature region is usually composed of smooth transition surface between two underlying surfaces, and the feature lines are the boundaries of this transition surface (called fillet) that should be extracted as shown in Fig. 1(right).

Detecting the surface-feature lines, we can get high-quality mesh segmentation in which each region has a similar geometrical property such as

monotonically varying curvature. Várady et al. [22] discussed the extraction of fillets from scanned meshes for the purpose of reverse engineering [21] in detail. First, they divide the mesh into highly curved and relatively flat parts with *Morse complex* segmentation. In this stage, the boundaries of highly curved region pass through the middle of fillet, then thickening the boundary curves, they obtain a pair of boundary curves as fillet edges. Although our objectives are almost the same as their studies, their approach is strongly affected by the quality of the *Morse complex* processing, in which some small fine structures may be extinguished.

In this paper, using the normal-voting tensor [15,18] (simply normal tensor below) whose eigenvalue analysis is a powerful tool for classifying feature saliencies: surface, crease, and corner, and its eigenvector is used to determine the ridge direction, we address the problem of extracting the surface-feature lines on meshes even when they include noise, because satisfactory solutions to them have not been obtained although its importance in reverse engineering is widely recognized.

#### 1.1. Related Works

Although there exist many studies on extracting the feature lines using such methods as geometric snakes

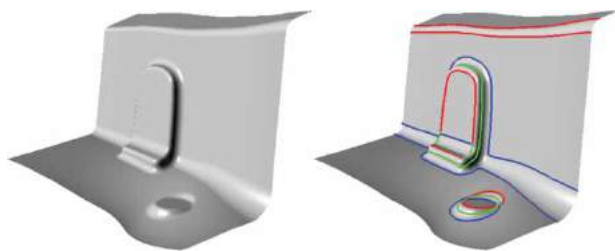


Fig. 1: Industrial/mechanical object. In the right figure, surface-feature lines are colored in red (convex fillet edges), blue (concave ones), and green (boundaries between convex and concave regions).

(or active contour) [1,3,13], *Morse theory* [19,22], topological method [24] in addition to the crest line, we focus on the methods based on the normal tensor framework. The reasons are as follows: this framework is more effective to detect sharp features than the above methods, provides important geometry information such as surface normals and principal directions, and executes geometry processing as mesh denoising [20] in the same framework, but there exist only a few studies as Kim et al. [9] and Jiao and Bayyana [8] which apply it to surface-feature lines extraction. We show their details below.

#### 1.1.1. Clustering Method

Kim et al. [9] obtained the feature lines similar to Fig. 2 by clustering the eigenvalues of normal tensor using the *K-means* algorithm. But their results are lack of (C) in Fig. 2, which is due to the small difference of eigenvalues in the neighborhood. Similar to [9], Lavoué et al. [11] used the discrete curvature values instead of the normal tensor eigenvalues. In both cases, the results strongly depend on the number of

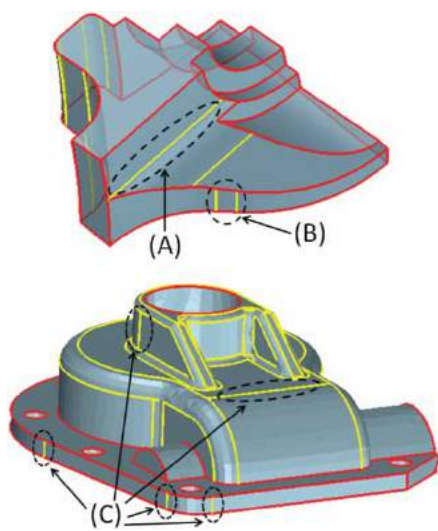


Fig. 2:  $C^0/C^1$  boundaries (our results).  $C^0/C^1$  boundaries are colored in red/yellow.

clusters  $K$ . In order to detect fine features in this approach, we have to use large  $K$ , but a high-quality merging process for the over-segmentation problem is instead needed [11,12]. Their issue is summarized as follows:

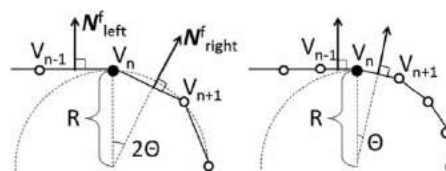
- In the clustering-based methods, processing equivalent to high-quality segmentation is required in order to extract the surface-feature lines.

#### 1.1.2. Dihedral Angle vs. Curvature

As a direct method to extract the lines without clustering process, Jiao and Bayyana [8] treated the detection of  $C^1$  boundary edges based on their normal tensor with several angle parameters and ridge direction given by the eigenvector corresponding to the smallest eigenvalue. But their method requires the small dihedral angle  $\theta = 1$  degree for detecting the  $C^1$  boundary edges while 10 degree is sufficient for the sharp features as their default setting. Their basic idea is to first extract sharp edges and then identify the  $C^1$  continuity point in the edge curves, and traverse the boundary edge from the detected point. Therefore, it is difficult to apply their method to the models which have no sharp edge as shown in Fig. 1. In their result, the line (A) in Fig. 2 is not shown in [8].

A fundamental characteristics of the angle-based method is as follows:

- It depends on the mesh size, i.e., the dihedral angle  $\theta$  at  $V_n$  tends to become large in a coarse mesh and small in a fine mesh irrespective of constant curvature radius  $R$  shown in the figure below.



Tab. 1 summarizes issues in the existing studies mentioned above from some viewpoints.

Yamakawa and Shimada [25] used both the dihedral angle and curvature gap across the boundary between two adjacent regions in their *polygon crawling* method. The gap is defined by the ratio of two curvatures, and their default threshold value is 2, while dihedral angle is set to 5 degree. The above two methods [8,25] have the following common problem:

- It is difficult to apply their methods to noisy scanned data, since the threshold of dihedral angle is too small.

Method	Vertex clustering & region growing	Feature detection		
		(A)	(B)	(C)
Kim [9]	necessary	O	?	X
Lavoué [11]	necessary	O	?	No data
Jiao [8]	Not necessary	X	O	No data
Ours	Not necessary	O	O	O

Tab. 1: Issues of the existing methods. The symbols: O, X, and ?, indicate “detected”, “not detected”, and “not shown in the paper”, respectively.

## 1.2. Contributions

Contributions of our method for extracting the surface-feature lines on meshes are as follows:

- based on the normal tensor framework, we directly detect both  $C^0$  and  $C^1$  boundary vertices according to a change of curvature values and the principal directions without clustering process as [9] and without using some critical angular values as [8];
- in order to be able to apply our method to noisy scanned data, we perform an anisotropic smoothing for the normal tensor. Hence, we can estimate the principal directions and their curvature values robustly to noises.

The rest of the paper is organized as follows: the principle of normal tensor is explained in Section 2. In Section 3, we introduce our method, and our results are shown in Section 4, and finally we conclude the paper in Section 5.

## 2. NORMAL TENSOR FRAMEWORK

### 2.1. Normal Tensor

We construct normal tensor  $A_n$  as a weighted sum of a covariance matrix of the facet normals  $N_{n'}^f$  ( $n' \in \Lambda^f(n)$ ) where  $\Lambda^f(n)$  indicates facet indices connecting to a vertex  $V_n$ :  $A_n = \sum_{n' \in \Lambda^f(n)} w_{n'} N_{n'}^f N_{n'}^{fT}$ . Here we use the Nelson Max’s weighting method [14] for  $w_{n'}$ . We let  $\sigma_1 \geq \sigma_2 \geq \sigma_3 (\geq 0)$  be the eigenvalues of  $A_n$ , and  $E_i$  ( $i = 1, 2, 3$ ) be the corresponding eigenvectors (see Fig. 3(left)). Then,  $A_n$  can be factored as the production of a rotation matrix  $R$  and a diagonal scaling matrix  $\Sigma$  as follows:

$$A_n = R \Sigma R^T = (E_1 \ E_2 \ E_3) \begin{pmatrix} \sigma_1 & 0 & 0 \\ 0 & \sigma_2 & 0 \\ 0 & 0 & \sigma_3 \end{pmatrix} (E_1 \ E_2 \ E_3)^T. \quad (1)$$

### 2.2. Vertex Classification for its Feature Saliency

Medioni et al. [15] defined saliency maps from the eigenvalues of normal tensor:  $S_{Su} = \sigma_1 - \sigma_2$  (surface),  $S_{Cr} = \sigma_2 - \sigma_3$  (crease), and  $S_{Co} = \sigma_3$  (corner). Thereafter, Page et al. [18] suggested the maximum of these three saliencies determines how we classify the feature saliency at a vertex  $V_n$  as follows:

$$\max\{S_{Su}, \varepsilon S_{Cr}, \varepsilon S_{Co}\} = \begin{cases} S_{Su} & : \text{surface, normal } N_n = E_1, \\ \varepsilon S_{Cr} & : \text{crease, tangent } T_n = E_3, \\ \varepsilon S_{Co} & : \text{no orientation} \end{cases} \quad (2)$$

where  $\varepsilon \geq 0$  is a constant parameter which controls the relative significance of the feature saliencies as shown in Fig. 3(right). This parameter should be fixed considering a level of noise. As a rule of thumb, Page et al. [18] proposed the formula to design for a specific crease angle  $\phi$  (see Fig. 3(right)):

$$\phi = 2 \arctan\left((\varepsilon + 1)^{-1/2}\right). \quad (3)$$

For example,  $\varepsilon \approx 100$  is correspondent to having set a crease angle to 10 degrees, and Fig. 4 shows the crease and corner vertices corresponding to three different angular values.

## 3. OUR PROPOSAL

Based on the normal tensor framework, we develop a method to detect the surface-feature lines according to a change of curvature values and enhance the normal tensor framework against noise. First, we introduce a principal curvature equation in Section 3.1. Next, we explain our method to detect points on the lines in Section 3.2 and to create the lines robustly against noisy data in Section 3.3.

### 3.1. Principal Curvatures Estimation

We introduce a principal curvature equation in the normal tensor framework. The curvatures are very important descriptors of surface-feature and play an important role in constructing our algorithm below. Our method can robustly estimate the principal

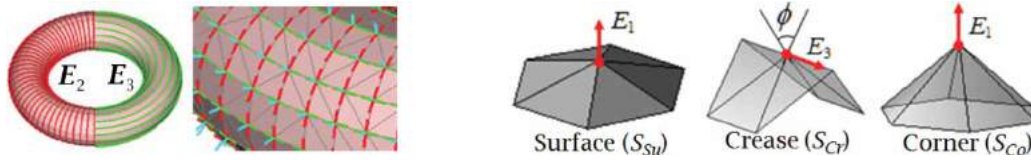


Fig. 3: Visualization of eigenvectors (left) and classification of feature saliencies (right).

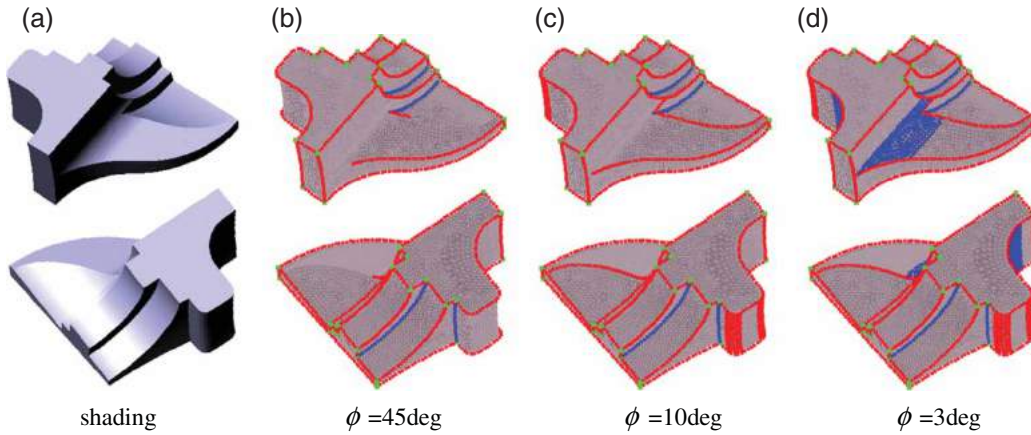


Fig. 4: Angle dependency for feature detection. Convex and concave crease, and corner vertices are colored in red, blue, and green, respectively.

curvatures via the surface normal filtering, whereas discrete approximations of the second order derivatives such as cotangent formula for mean curvature and angle deficit method for Gaussian curvature [16] are very error-sensitive.

The principal curvatures are given by the eigenvalues of  $2 \times 2$  shape operator matrix [5]:  $S = (TB)^T(\nabla N^T)(TB)$ , where  $\nabla$  denotes the gradient,  $N$  and  $(TB)$  are surface normal and any two orthogonal basis to  $N$ , respectively. On the other hand, as shown in Fig. 3(left), the eigenvectors  $E_i (i = 1, 2, 3)$  of the normal tensor  $A_n$  are mutually orthogonal and  $E_1$  indicates the surface normal,  $E_2/E_3$  are the directions of the principal curvatures  $k_{\max}/k_{\min}$ . Hence, using these eigenvectors, we obtain the following shape operator matrix whose eigenvalues denote the principal curvatures:

$$S = (E_2 E_3)^T(\nabla E_1^T)(E_2 E_3). \quad (4)$$

In our study, the gradient of the normal vector at  $V_n$  is calculated by  $\nabla N_n^T = RW^T(WW^T)^{-1}$  proposed in [4], where  $R$  and  $W$  are the matrix form of the normalized difference vector  $V_{n'} - V_n$  projected on the tangent space at  $V_n$  and  $N_{n'} - N_n$  including additional condition, respectively. (See [4] for details.)

Here we note that Eq. (4) is governed by the normal tensor eigenvectors. Therefore, we can obtain the principal curvatures robustly to large noise in scanned meshes via  $S$  by smoothing the normal tensor which includes information of not only normal vectors but also curvature values and principal directions. Furthermore, we can obtain the normal tensor corresponding to shape features by the anisotropic

smoothing [20]:

$$A_n^{(t+1)} \leftarrow A_n^{(t)} + \frac{1}{\sum_{n' \in \Lambda(n)} w_{nn'}^A} \sum_{n' \in \Lambda(n)} w_{nn'}^A (A_{n'}^{(t)} - A_n^{(t)}). \quad (5)$$

Here,  $A_n^{(t)}$  is the normal tensor at vertex  $V_n$  in the  $t$ -th step and  $w_{nn'}^A$  is the anisotropic weight:

$$w_{nn'}^A = \frac{l_{nn'} l_{nn'}^\perp}{l_{nn'}^A}, \quad l_{nn'}^A \equiv \sqrt{(V_{n'} - V_n)^T \frac{A_{n'} + A_n}{2} (V_{n'} - V_n)},$$

where  $l_{nn'} = |V_{n'} - V_n|$  and  $l_{nn'}^\perp$  is the projected distance of the edge  $V_n V_{n'}$  on the tangent plane at  $V_n$ .

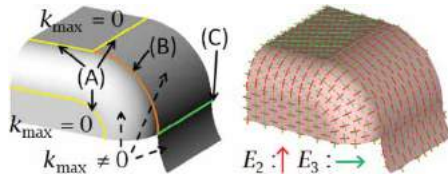
In our experiments with noisy data, we performed the normal tensor smoothing by 10 times iterations in 2-ring neighborhood as default parameters. Through the experiments we have studied that 2-ring neighborhood should be taken for robust estimations such as local surface fitting and mesh smoothing against noisy data and that the number of iterations is enough for the error convergence for all the data.

### 3.2. Detection of Points on Surface-feature Lines

Once we have obtained the principal curvature  $k_{\max}$  by the maximum eigenvalue of Eq. (4) and the principal directions  $E_2$  and  $E_3$  by the eigenvectors of smoothed normal tensor Eq. (5) for noisy data, we can extract the surface-feature lines whose control points are interpolated by the vertices having the following properties: a vertex  $V_n$  is the point if



- C0) its feature saliency defined by Eq. (2) is not surface saliency,  $S_{Su}$ ;
- C1) the gap of the curvature  $k_{max}$  exists in its principal direction  $E_2$  in the neighborhood of  $V_n$ . For example, the vertices on the yellow lines (A) in the below figure satisfy this condition;
- C2) the principal directions in non-flat regions differ (mutually orthogonal) as shown in (B);
- C3) the sign of curvature changes in the direction  $E_2$  as shown in (C).



Note that condition C0) is mainly used for detecting the  $C^0$  boundary vertices, and the other conditions, especially C1), is designed for extracting  $C^1$  boundary ones as shown in Fig. 5(left). We explain the details below.

First, owing to the condition C0), we detect the vertices on sharp edges and corners. In our study, we set the angle parameter  $\phi = 45$  degree in Eq. (3). We denote this function by **IsSharpVtx**( $V_n$ ).

Second, in condition C1), we define the curvature gap as follows: given two vertices  $V_n$  and its neighbor vertex  $V_{n'}$ ,

$$\left| \frac{k_{max} \text{ at } V_n}{k_{max} \text{ at } V_{n'}} \right| > e \quad \text{or} \quad \left| \frac{k_{max} \text{ at } V_n}{k_{max} \text{ at } V_{n'}} \right| < \frac{1}{e},$$

where  $e$  is a threshold value and is set to 2 as a default in our experiments. Our default parameters  $e = 2$  and  $\phi = 45$  degree in C1) are the same with those in *polygon crawling* method [25]. Then, at first, we verify

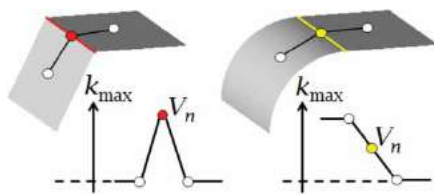


Fig. 5:  $C^0$  and  $C^1$  boundary vertex [12] (left) and vertex neighborhood (right).

the gaps of curvature at  $V_n$  for two adjacent vertices  $V_{n,F}$  and  $V_{n,B}$ , which are the nearest neighbor vertices of  $V_n$  in the direction  $E_2$  and  $-E_2$  respectively (see Fig. 5 right). If there exists the gap at least in one side and there exists no curvature gap for the next vertex along the direction, then  $V_n$  is regarded as the point on surface-feature lines. We denote C1) function by **HasCurvatureGap**( $V_n$ ).

C2) and C3) are also easily formulated in a similar manner, and we denote them as the Boolean functions: **IsOrthogonal**( $V_n$ ) and **IsConvexConcave**( $V_n$ ), respectively.

Finally, in order to avoid erroneous decision due to some irregularities and small noises in meshes, we check the vertices  $V_{n,R}$  or  $V_{n,L}$ , which are the nearest neighbor vertices of  $V_n$  in the direction  $E_3$  and  $-E_3$  respectively, also satisfy the conditions from C0) to C3).

### 3.3. Creation of Surface-feature Lines for Noisy Data

Different from CAD models, there exist mesh irregularities in scanned data accompanied by noises, and mesh edges are not arranged along the feature lines as shown in the dashed line of Fig. 6(a). In those data, we search adjacent vertices  $V_{n,F}$ ,  $V_{n,B}$ ,  $V_{n,R}$ , and  $V_{n,L}$  in 2-ring neighborhood. We can robustly obtain the principal directions if normal tensor is smoothed for 2-ring neighborhood.

Once we obtained the characteristic points as shown in Fig. 6(b), then we select the representative points, each of which is the projected point of the centroid of its 2- or 3-ring neighborhood points onto the mesh, trace the representative points in the direction of  $E_3$  as shown in Fig. 6(c), and finally create a line by interpolating them.

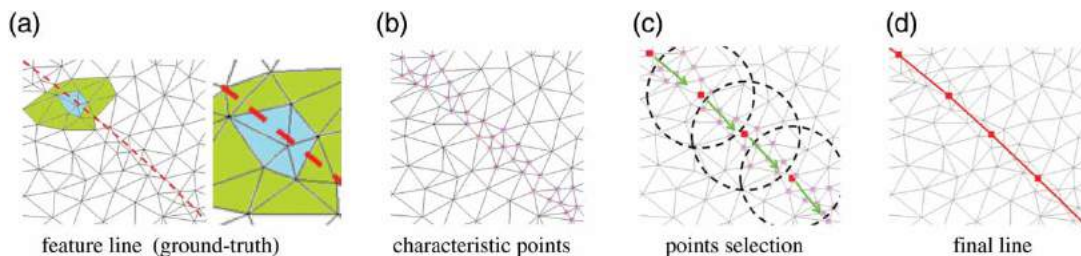
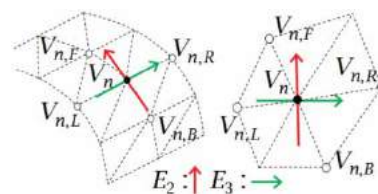


Fig. 6: Line creation process for noisy data. (a) indicates a part of wheel arch (= typical character-line of automobile as shown in Fig. 12), and dashed line colored in red is the ground-truth of feature line. (b) shows the characteristic points extracted by our method. (c) and (d) explain the process of line creation.

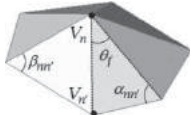
## 4. RESULTS

We have implemented our proposed algorithm described in Section 3 using MSVC++ 2012 on mobile Intel/Core i7 3520M (2.9GHz) CPU computer. All meshes are rendered with flat shading to show faceting. To demonstrate the capabilities of our method, we conducted some experiments for both CAD models and real-world scanned data.

### 4.1. Curvature Estimation

Using torus data, we show the accuracy and robustness of our curvature estimation compared with a representative method by Meyer et al. [16], in which the mean curvature vector  $\mathbf{K}(V_n)$  and Gaussian curvature  $\kappa_G(V_n)$  are formulated by the following equations:

$$\mathbf{K}(V_n) = \frac{1}{2A_{mixed}} \sum_{n' \in \Lambda(n)} (\cot \alpha_{nn'} + \cot \beta_{nn'}) (V_n - V_{n'}),$$

$$\kappa_G(V_n) = \frac{1}{A_{mixed}} (2\pi - \sum_{f \in \Lambda_f(n)} \theta_f),$$


where  $A_{mixed}$  is a facet area in the neighborhood of  $V_n$ , and  $\Lambda(n)/\Lambda_f(n)$  are the vertex/facet indices in one-neighborhood. (See [16] for details.)

First, in the case of clean mesh created from CAD surface, two methods have small errors as shown

	Clean mesh		Noisy Mesh	
	Mean curvature	Gaussian curvature	Mean curvature	Gaussian curvature
Meyer et al. [16]	0.0318	0.147	9.539	57.15
Ours	0.0572	0.359	0.585	1.267

Tab. 2: Comparison of the relative errors defined by Eq. (6) of curvature values for clean and noisy meshes. Clean mesh is created from CAD surface (Fig. 7(a)) and random noise with maximum length  $0.1\hat{e}$  ( $\hat{e}$  is the average length of mesh edges) are added to clean mesh in noisy one (Fig. 7(b)).

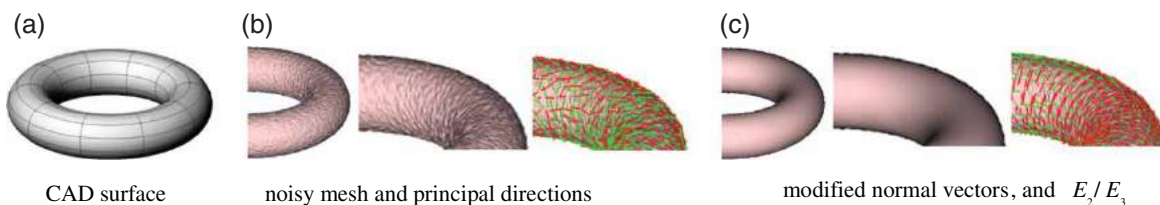


Fig. 7: Torus data and their appearances. In figure (c), only normal vectors are replaced with the eigenvector of smoothed normal tensor in noisy mesh (b). Furthermore, the principal directions are shown by red ( $E_2, k_{max}$ ) and green ( $E_3, k_{min}$ ) lines, respectively, in the same as Fig. 3(left).

in the left side of Tab. 2, in which each value indicates the maximum values of relative errors defined as follows:

$$\max \left\{ \frac{|x_n - \tilde{x}_n|}{|\max \{\tilde{x}_n\}_{n=1}^N|} \right\}_{n=1}^N, \quad (6)$$

where  $x_n$  is the curvatures (mean or Gaussian), and  $\tilde{x}_n$  is the correct value calculated by the CAD data. Here, we set the denominator to the absolute maximum curvature value of the CAD data.

Next, in the case of noisy mesh, which is created by adding random noise with maximum length  $0.1\hat{e}$  ( $\hat{e}$  is the average length of mesh edges), our result has less error, while the representative method produces erroneous result.

In our method, smoothing the normal tensor by Eq. (5), we have obtained correct normal vectors along with the principal directions from its eigenvectors. Fig. 7(c) shows the validity of the normal vectors by replacing the ones in noisy mesh shown in Fig 7(b) with the eigenvector  $E_1$  of normal tensor without vertex relocation, and shows the correctness of the estimation of the principal directions. Note that in Fig. 7(c), we make the appearance smooth by changing normal vectors in noisy data, whereas *bump map* [2] in CG area makes it noisy by changing them. Since our method is not accompanied by the vertex relocation, we can efficiently estimate the normals and curvature values.

Fig. 8 shows the influence of different parameters with respect to the number of iterations and neighborhoods in normal tensor smoothing by Eq. (5).

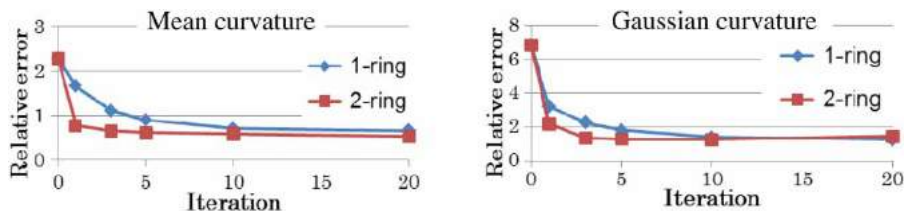


Fig. 8: Influence of the number of iterations and neighborhoods in normal tensor smoothing by Eq. (5).

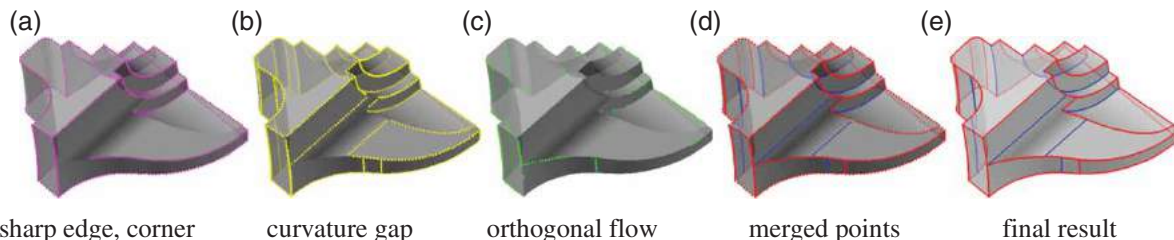


Fig. 9: Result for Fandisk model. In the figures (d) and (e), red and blue colors express the features which exist on the convex and concave parts, respectively.

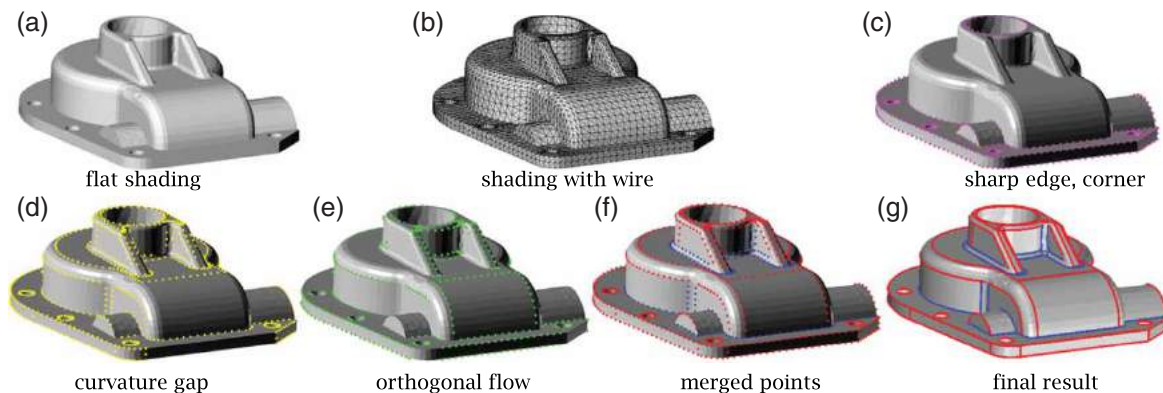


Fig. 10: Result for Casting model. In the figures (f) and (g), red and blue colors express the features which exist on the convex and concave parts, respectively.

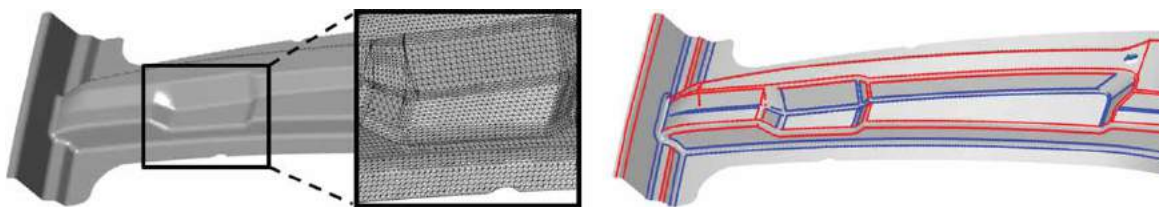


Fig. 11: Result for a typical automobile mechanical part (center pillar). Surface-feature lines on convex parts are colored in red and in blue on concave ones.

10 iterations are enough for convergence and 2-ring neighborhood gives a more accurate result than 1-ring one.

**4.2. Experiment in CAD Models**

First, we show the results of each function in our algorithm using Fandisk shown in Fig. 9. Fig. 9(a) shows the result with the function *IsSharpVtx()* only. The functions *HasCurvatureGap()* and *IsOrthogonal()*

do extract the boundary vertices, which are not extracted as the sharp features, as shown in Fig. 9(b), (c). Fig. 9(d) is the result which is the union of (a), (b) and (c), and is distinguished by different colors; the vertices in convex part are colored in red and concave in blue. Fig. 9(e) shows the final surface-feature lines whose control points are interpolated by the characteristic points in Fig. 9(d), and shows effectiveness of our method. This is the same in the Casting model shown in Fig. 10.



As we mentioned in Section 1 using Fig. 2 and Tab. 1, our results shown in Fig. 9(e) and Fig. 10(g) have well extracted surface-feature lines in both models compared with the existing methods [8,9].

Second, Fig. 11 shows another example of our results, and many fillet edges are well extracted. Furthermore, the surface-feature lines represent the

underlying surface very well, which is important for mesh segmentation.

#### 4.3. Experiment in Scanned data

In order to demonstrate effectiveness of our proposed method accompanied by normal tensor smoothing

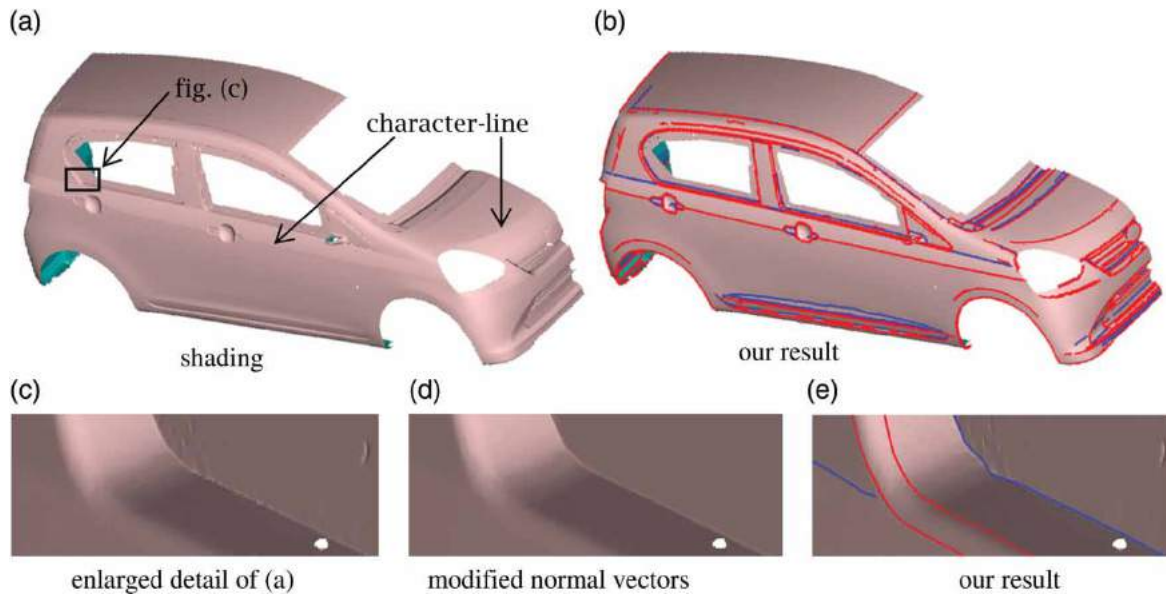


Fig. 12: Result for scanned data of clay model. In figures (b) and (e), red and blue colors express the features which exist on the convex and concave parts, respectively. In figure (d), only normal vectors are replaced with the eigenvector of smoothed normal tensor in the raw data (c).

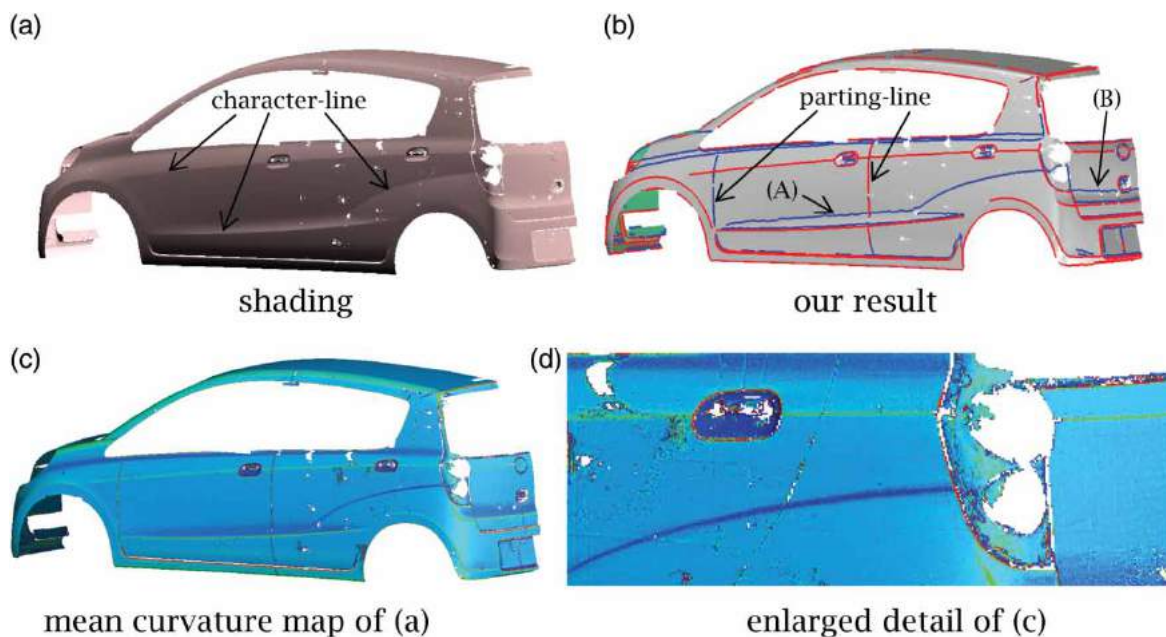


Fig. 13: Result for scanned data of a car in the market. (a) and (b) show the flat shading of the model, and our extraction of surface-feature lines colored in red (convex parts) and blue (concave ones), respectively, (c) shows mean curvature map for raw data whose noise level can be seen in (d).



Model	Facets	Tensor Smoothing	Run-time [sec]	Details of run-time [sec]		
				Smoothing / Curvature calc.	Points detection	Line creation
Fig. 9 FanDisk	12,946	No	0.17	0.078	0.047	0.047
Fig. 10 Casting	10,224	No	0.18	0.078	0.047	0.062
Fig. 11 C-Pillar	121,080	No	1.12	0.54	0.32	0.26
Fig. 12 Exterior1	2,386,918	Yes	106	45	14	47
Fig. 13 Exterior2	2,164,666	Yes	95	43	12	40

Tab. 3: Model size, parameters and run-time in our method. Run-time is measured on the mobile Intel/Core i7 3520M (2.9GHz) CPU computer. Tensor smoothing was done by Eq. (5).

given by Eq. (5), we show two examples which were scanned from industrial style-design objects; a clay model and a car in the market.

First, Fig. 12 shows the result for the clay model. In Fig. 12(b), the important character-lines shown in Fig. 12(a) are well extracted. Fig. 12(c) indicates the enlarged detail of (a) and shows two geometrical shapes: one is a convex-fillet and the other is a sharp edge. In our result shown in Fig. 12(e), both shapes are detected as surface-feature lines colored in red and blue, respectively as well as the extraction of important character-lines in car-styling design. In this experiment, using our normal tensor smoothing by Eq. (5) the same as the noisy torus case mentioned above, we have obtained correct normal vectors as shown in Fig. 12(d). We can see that our tensor smoothing technique makes data smooth preserving the sharp edges well.

Next, Fig. 13 shows the result for another automobile exterior model. Compared with Fig. 12, there exist a lot of important character-lines in this data accompanied by parting-lines in door parts, which are formed by concave grooves or lacking the data since this model was measured on a car in the market. In our experiment, we have detected these character-lines, and region boundaries indicated by blue lines (A) and (B).

#### 4.4. Performance

Tab. 3 lists the model size, run-time and its details to create surface-feature lines for each data.

The great portion of our total processing time for scanned data is spent on smoothing of normal tensor in order to estimate the principal curvatures and their directions robustly against noises and on creating the feature lines. Hence the processing time that detects feature points based on the local and non-iterative operations is faster than non-local or iterative approaches.

## 5. CONCLUSION

Compared with the existing results shown in [8,9], our method have extracted complete surface-feature

lines as shown in (A), (B), and (C) of Fig. 2. Furthermore, owing to our anisotropic smoothing of normal tensor without vertex relocation, we have obtained the smoothed surface normal and principal directions robustly and efficiently, and have estimated the principal curvatures. Then, we have extracted the lines as shown in Fig. 12 and 13, which become an important clue to realizing a high-quality mesh segmentation which is our main objectives. Consequently, our method can extract surface-feature lines without relying on critical values of dihedral angles [8] and vertex clustering [9].

Future research includes applying our method to mesh segmentation, reconstruction, and so on.

## ACKNOWLEDGEMENTS

The Casting model shown in Fig. 2 and Fig. 10 is courtesy of the AIM@SHAPE Shape Repository, and the scanned data shown in Fig. 12 and Fig. 13 are provided by Daihatsu Motor Co., Ltd.

## REFERENCES

- [1] Bischoff, S.; Weyand, T.; Kobbelt, L.: Snakes on Triangle Meshes, *Bildverarbeitung für die Medizin*, 2005, 208-212
- [2] Blinn, J. F.: Simulation of wrinkled surfaces, *Computer Graphics*, 12(3), 1978, 286-292.
- [3] Clements, A.; Zhang, H.: Minimum Ratio Contours on Surface Meshes, *Proceedings of the IEEE International Conference on Shape Modeling and Applications 2006*, 26-37.
- [4] Grana, C.; Cucchiara, R.: Performance of the mpeg-7 shape spectrum descriptor for 3d objects retrieval, in: *Proceedings of the Second Italian Research Conference on Digital Library Management Systems (IRCDL 2006)*, Padova, Italy, 2006, 11-14.
- [5] Hadwiger, M.; Sigg, C.; Scharsach, H.; Bühler, K.; Gross, M.: Realtime ray-casting and advanced shading of discrete isosurfaces, *Computer Graphics Forum*, 24(3), 2005, 303-312.
- [6] Higashi, M.; Saitoh, T.; Watanabe, Y.; Watanabe, Y.: Analysis of aesthetic free-form surfaces by

- surface edges, Proceedings of the Third Pacific Conference on Computer Graphics and Applications, 1995, 294-305.
- [7] Hildebrandt, K.; Polthier, K.; Wardetzky, M.: Smooth feature lines on surface meshes, Eurographics Symposium on Geometry Processing, 2005, 85-90.
- [8] Jiao, X.; Bayyana, N. R.: Identification of C1 and C2 discontinuities for surface meshes in cad, Computer-Aided Design, 40(2), 2008, 160-175.
- [9] Kim, H. S.; Choi, H. K.; Lee, K. H.: Feature detection of triangular meshes based on tensor voting theory, Computer-Aided Design, 41(1), 2009, 47-58.
- [10] Kim, S.-K.; Kim, C.-H.: Finding ridges and valleys in a discrete surface using a modified mls approximation, Computer-Aided Design, 37(14), 2005, 1533-1542.
- [11] Lavoué, G.; Dupont, F.; Baskurt, A.: Constant curvature region decomposition of 3d-meshes by a mixed approach vertex-triangle, Journal of WSCG 12(2), 2004, 245-252.
- [12] Lavoué, G.; Dupont, F.; Baskurt, A.: A new CAD mesh segmentation method based on curvature tensor analysis, Computer-Aided Design, 37(10), 2005, 975-987.
- [13] Lee, Y.; Lee, S.; Geometric Snakes for Triangular Meshes, Computer Graphics Forum, 21(3), 2002, 229-238
- [14] Max, N.: Weights for computing vertex normals from facet normals, Journal of Graphics Tools, 4(2), 1999, 1-6.
- [15] Medioni, G.; Lee, M.-S.; Tang, C.-K.: A Computational Framework for Segmentation and Grouping, Elsevier, Amsterdam, 2000.
- [16] Meyer, M.; Desbrun, M.; Schröder, P.; Barr, A.H.: Discrete Differential-Geometry Operators for Triangulated 2-Manifolds, Proceedings of Visualization and Mathematics, 2003, 35-57.
- [17] Ohtake, Y.; Belyaev, A.; Seidel, H.-P.: Ridge-valley lines on meshes via implicit surface fitting, ACM Transactions on Graphics (Proc. of ACM SIGGRAPH 2004), 23(3), 2004, 609-612.
- [18] Page, D. L.; Koschan, A. F.; Sun, Y.; Paik, J. K.; Abidi, M. A.: Robust crease detection and curvature estimation of piecewise smooth surfaces from triangle mesh approximations using normal voting, Proceedings of the International Conference on Computer Vision and Pattern Recognition, 1, 2001, 162-167.
- [19] Sahner, J.; Weber, B.; Prohaska, S.; Lamecker, H.: Extraction of feature lines on surface meshes based on discrete Morse theory, Computer Graphics Forum, 27(3), 2008, 735-742.
- [20] Tsuchie, S.; Higashi, M.: Surface mesh denoising with normal tensor framework, Graphical Models, 74(4), 2012, 130-139.
- [21] Várady, T.; Martin, R.: Reverse Engineering, chapter 26, 651-681, North-Holland, 2002.
- [22] Várady, T.; Facello, M. A.; Ter'ek, Z.: Automatic extraction of surface structures in digital shape reconstruction, Computer-Aided Design, 39(5), 2007, 379-388.
- [23] Vidal, V.; Wolf, C.; Dupont, F.: Robust feature line extraction on CAD triangular meshes, Proceedings of the International Conference on Computer Graphics Theory and Application, 2011, 106-122.
- [24] Weinkauff, T.; Günther, D.: Separatrix Persistence: Extraction of salient edges on surfaces using topological methods, Computer Graphics Forum, 28(5), 2009, 1519-1528
- [25] Yamakawa, S.; Shimada, K.: Polygon crawling: Feature-edge extraction from a general polygonal surface for mesh generation, Proceedings of 14th International Meshing Roundtable, 2005, 257-274.
- [26] Yoshizawa, S.; Belyaev, A.; Yokota, H.; Seidel, H.-P.: Fast and faithful geometric algorithm for detecting crest lines on meshes, Pacific Conference on Computer Graphics and Applications, 2007, 232-237.