博士論文（要約）

Extraction of coherent and smooth feature lines from meshes

（破線や揺れの少ない特徴線の抽出）

高　琪琪

Acknowledgements

First, I owe my deepest gratitude to my Ph.D. advisor, Prof. Yasushi Yamaguchi, for his continuous support of my Ph.D. study and related research. Whenever I found myself stuck in a tunnel with no exit and was uncertain about how to move on, he always pointed me in the right direction. I am blessed to have such an advisor and mentor with patience, enthusiasm, and immense knowledge.

Besides my advisor, my special thanks go to Prof. Takashi Kanai and Prof. Tomohiro Tachi, who have constantly been offering me inspiring advice from different points of view at lab meetings over the years. I would also like to thank the rest of my thesis committee: Prof. Kazunori Yamaguchi and Prof. Akimasa Morihata, not only for their insightful comments but also for the questions they asked that motivated me to improve my research from various perspectives.

I would like to express my sincere gratitude to Prof. Yong-Liang Yang[1], Prof. Johannes Wallner[2], and Dr. Yu-Kun Lai[3], the co-authors of an important work related to this thesis. They shared their thoughts in detail with me, which helped me better understand their work. I also benefited vastly from the stimulating discussion with Prof. Otto Röschel[2] and Prof. Hans-Peter Schröcker[4] on my research.

[1]University of Bath
[2]Graz University of Technology
[3]Cardiff University
[4]University of Innsbruck

In particular, I am tremendously indebted to my previous lab members, Dr. Tatsuya Yatagawa[5], Dr. Hideki Todo[6], and Hiroki Sugita, for enlightening me in an early stage of my research. Without you, this work could never have been possible. I thank my labmates at both Yamaguchi lab and Kanai lab for their invaluable support, for the sleepless nights we worked together before deadlines, and for all the fun we have had in the last few years.

Very special thanks go to the financial support from the University of Tokyo Fellowship and the Integrated Human Sciences Program for Cultural Diversity (IHS) at the University of Tokyo. IHS gave me the opportunity to broaden my horizon with a boundary-breaking curriculum. I am especially grateful to Prof. Mariko Muramatsu, who made my life-changing experience in Italy possible, through which I realized how technology could be used in real archaeological contexts.

Finally, last but by no means least, I am grateful to my parents, who have always believed in me when I failed to do so. You are my mood stabilizers! I am also grateful to my friends both on and off campus who helped me resist the temptation to live like a "hermit" on foreign soil. Pursuing a Ph.D. is combating both internal and external uncertainty every single day. To all whom I unfortunately cannot mention one-by-one, thank you all for making me feel that I am not fighting this war alone.

[5]University of Tokyo
[6]Chuo Gakuin University

Table of contents

List of figures

List of tables

Chapter 1

Introduction

This chapter is part of an unpublished paper, and thus is not included in this abridged dissertation.

Chapter 2

Building blocks of feature-line extraction

2.1 Mechanism of feature-line extraction

This thesis focuses on view-independent ridge-valley-like feature lines. As mentioned previously, ridge-valley lines [26] are loci of points where principal curvatures attain local extrema along corresponding principal directions of curvature. At an arbitrary point on the surface illustrated in Fig. 2.1 (left) along one of two principal directions of curvature (indicated by a purple line), the principal curvature is always zero. The principal curvature along the other principal direction of curvature (indicated by a navy arrow) attains its maximum somewhere on a ridge line. Such local extrema are traced by locating zero-crossings of derivatives of curvatures, that is, third-order derivatives. When viewed from the top, the principal direction of curvature (indicated by a navy arrow) is clearly roughly perpendicular to the nearby ridge and can be oriented toward the nearby ridge line (see Fig. 2.1 right).

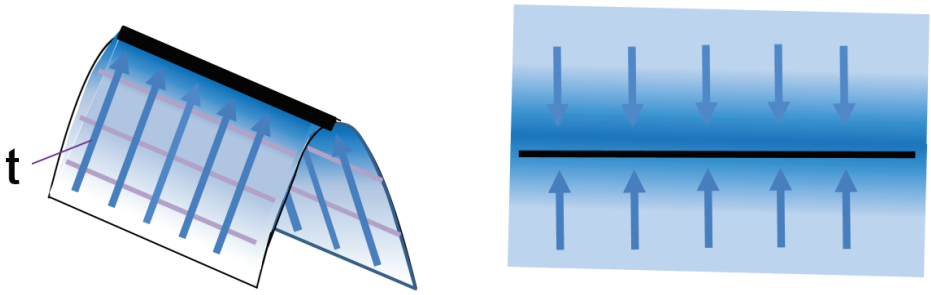

Fig. 2.1 Feature lines can be generalized as the loci of points where a certain geometric property I, such as principal curvature, attains local extrema in a corresponding critical direction $\mathbf{t}$, such as the principal direction of curvature. The geometric property I increases along critical directions. The value of I is indicated by the color intensity.

Similarly, apparent ridges [17] are the local extrema of view-dependent principal curvatures along corresponding view-dependent curvature directions, and suggestive contours [11] are the local extrema of radial curvatures along corresponding radial directions.

In a nutshell, the geometric properties exploited in these feature-line extraction algorithms may vary; however, they share an underlying logic. Given a certain form of geometric property I and a direction $\mathbf{t}$ that we refer to as "critical direction," both defined over mesh vertices, a set of feature lines can be defined as the loci of points where the geometric property I attains its local extrema (or other special values) in its corresponding critical direction $\mathbf{t}$. Similar to the relation between the principal direction of curvature, principal curvature, and ridge lines, such a direction should be roughly perpendicular to (i.e., it can be oriented toward) the nearby feature line, as illustrated in Fig. 2.1 (right). We can also see that the geometric property I increases along critical directions.

Throughout this thesis, we will refer to the geometric property I and the critical direction $\mathbf{t}$ as the two building blocks of feature-line extraction.

2.2 Acquisition of the two building blocks of feature-line extraction

To extract feature lines, that is, to extract points on feature lines over mesh edges, a scalar field and a direction field—directions that can be oriented toward (and are ideally roughly perpendicular to) potential nearby ridges or valleys—over the entire mesh are required (see Fig. 2.2). In this chapter, we explore acquisition of these two building blocks of feature-line extraction.

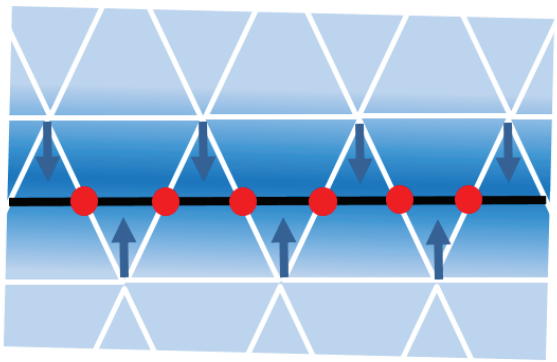

Fig. 2.2 Feature lines are determined by the sequence of points (highlighted in red) sampled over mesh edges. The locations of these points are decided jointly by critical directions $\mathbf{t}$'s (navy arrows) and geometric properties I's over mesh vertices. The value of I is indicated by the color intensity.

Most existing algorithms for extracting feature lines on meshes in object-space are based on the classical definition of curvature in a discrete differential geometry context. Noticeably, for most feature-line extraction algorithms proposed so far, both geometric properties and critical directions are derived from high-order derivatives. Typically, these differential invariants must be approximated using discrete differential geometry. By creating a hierarchy of meshes with different levels of detail and estimating differential invariants from them, many existing algorithms for feature-line extraction, including [26], can efficiently extract feature lines of increasingly large scales. They are relatively fast to compute but essentially sensitive to surface noise and mesh tessellation.

Effort has also been made to extract curvature information from the viewpoint of integral invariants. Alliez et al. [1] introduced an operator that integrates (averages) a curvature tensor over a neighborhood of a central point. To obtain a smooth curvature tensor field, this operator considers a relatively large neighborhood. Later, Yang et al. [35] and Pottmann et al. [27] proposed an integrals-based definition of both principal curvatures and principal directions of curvature at a given resolution level r. This method can be considered an extension of [23], which exploited integral invariants to estimate the curvature of planar curves. Given neighborhoods constructed by kernel balls or spheres centered on a given surface with radius r, curvature information can be estimated via principal component analysis (PCA) of such neighborhoods. At the same time, geometric properties, such as volume or spherical area bounded by the kernel ball or sphere and the surface's interior, are related to mean curvature and thus have specific geometric implications. Notably, [12] presented methods for curvature estimation directly from point clouds by constructing and analyzing various covariance matrices defined over local neighborhoods.

For the rest of this chapter, we will review several differential invariants and integral invariants from which both the geometric property and the critical direction can be derived.

2.2.1 Differential invariants

Without loss of generality, we focus on curvature, the most commonly employed differential invariant in the context of feature-line extraction. Defining curvature on discrete surfaces is never trivial. According to [29], existing methods for estimating curvature can be roughly categorized into three groups:

(a) Patch-fitting methods, which essentially fit parabolas to surface samples (vertices);

(b) Normal curvature-based methods, which essentially fit circles to surface samples (outgoing mesh edges of a vertex);

(c) Tensor-averaging methods, which calculate the average of curvature tensors over a tiny surface patch.

While (a) and (b) could work on 1-ring neighborhoods of mesh vertices, they proved to become unstable on meshes with degenerate configuration. Rusinkiewicz [29] proposed a more robust method (see Appendix A.2.2). Directional derivatives of surface normals along the directions of mesh edges are turned into a set of linear constraints on the elements of the per-facet curvature tensors **II** (the matrix of the second fundamental form; see Appendix A). These elements are computed using least squares. The per-vertex curvature tensors are then estimated as a weighted average of the per-facet curvature tensors of its adjacent facets. Since a per-vertex normal is estimated as the weighted average of its adjacent facet normals, it turns out that reliable estimation of curvature requires geometric information of at least 2-ring neighborhoods of mesh vertices.

For the vertex in the center (highlighted in red) of Fig. 2.3, its 2-ring neighborhood includes all the facets present in the image. As a result of curvature tensors averaging, the original surface patch could be considered "smoothed" to the shape indicated by the dashed lines, and consequently only one feature line in the center would be extracted instead of two on both sides of the central vertex. Each of these two feature lines is defined over only two strips of mesh elements, that is, on a scale smaller than the neighborhood size intrinsically specified by the geometric property.

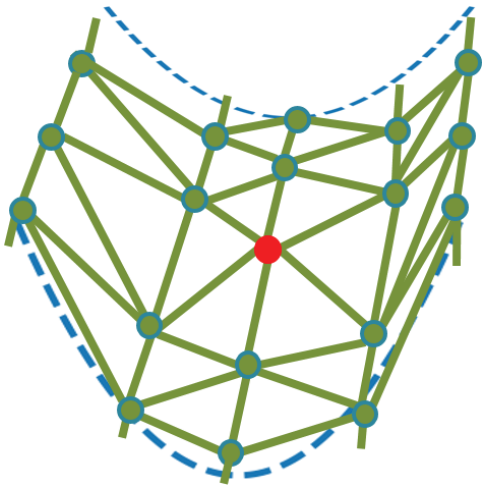

Fig. 2.3 Feature lines of comparatively small scales are "smoothed out" when extracted with curvature.

2.2.2 Integral invariants

Following the notation of [35], we begin by considering an oriented surface Φ and the 3D domain D bounded by Φ.

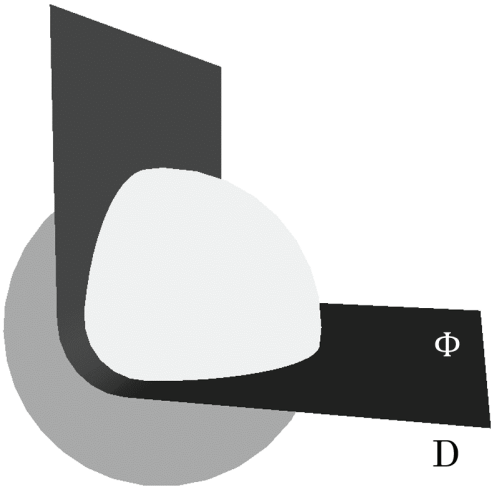

Fig. 2.4 A 3D illustration of a kernel ball centered on surface Φ.

For a specific point $\mathbf{p} \in \Phi$, a kernel ball $B_r(\mathbf{p})$ and a kernel sphere $S_r(\mathbf{p}) = \partial B_r(\mathbf{p})$, both centered on $\mathbf{p}$ with radius r, can be constructed. $N_b^r(\mathbf{p}) = D \cap B_r(\mathbf{p})$ and $N_s^r(\mathbf{p}) = D \cap S_r(\mathbf{p})$ are referred to as the *ball neighborhood* and the *sphere neighborhood* (see Fig. 2.5), respectively. For an arbitrary integration domain Π, we denote its indicator function by $1_\Pi(\mathbf{x})$, that is, $1_\Pi(\mathbf{x}) = 1$ if $\mathbf{x} \in \Pi$ and $1_\Pi(\mathbf{x}) = 0$ otherwise. The geometric properties, that is, the integral invariants obtained by integrating the indicator function over $N_s^r(\mathbf{p})$ and $N_b^r(\mathbf{p})$, are referred to as the *surface area descriptor* and the *volume descriptor*, respectively [28].

The critical directions corresponding to such geometric properties are the principal directions of curvature, which according to [35], can be derived by PCA of the sphere or ball neighborhoods. PCA of the ball or sphere neighborhoods centered on point $\mathbf{p}$ generates corresponding covariance matrices, eigenvectors of which can serve as principal directions of

curvature and the surface normal vector at **p**. Indeed, telling which is which among the three can be difficult.

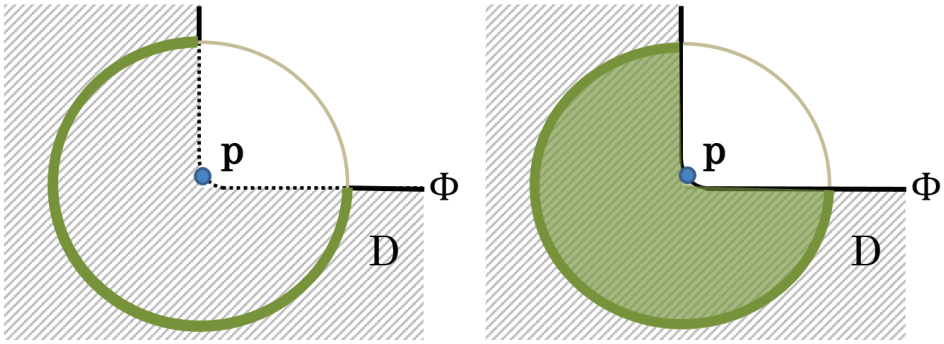

Fig. 2.5 A 2D conceptual illustration of a sphere neighborhood (left) and a ball neighborhood (right) centered at point **p** on surface Φ (see Fig. 2.4). Both are highlighted in green.

Apart from integral invariants defined over sphere or ball neighborhoods, which are reported to exhibit robustness and be consistent with classic settings of curvature [35], we consider *ambient occlusion* [22, 5], an integral invariant defined from the viewpoint of "visibility."

Ambient occlusion is a technique for simulating diffuse indirect illumination in a 3D scene, a simplified version of "obscurance" [39]. Ambient occlusion at a point on a surface is usually defined as the cosine-weighted percentage of the hemisphere where incoming ambient light cannot reach that point. Alternatively, instead of the entire scene, we can consider ambient light in a bounded region within distance r from the point when computing its ambient occlusion [4, 21]. Bavoil et al. [4] borrowed the concept of *horizon mapping* [25], which was previously defined to create realistic shadows, and proposed an image-space horizon-based approach to ambient occlusion approximation. A *horizon map* in this context is a precomputed table of elevation angles that lists, for a point on the mesh, the elevation of the horizon in a sampled collection of directions (denoted by azimuthal angles) when viewed from that point. As a horizon map and ambient occlusion are closely related, [36] defined a so-called "direction of maximum occlusion" based on the intuition that a nearby valley is more likely to appear in the direction with higher elevation angle, that is, a nearby "mountain." It is the

direction with maximal elevation angle and possibly maximal occlusion over the horizon map.

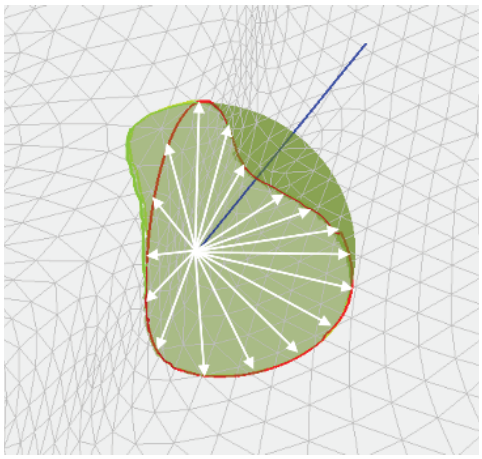

Fig. 2.6 The horizon viewed from a mesh vertex. The horizon is drawn in red. The intersection curve of the surface and the hemisphere is highlighted in light green. The surface normal vector at the vertex is drawn with a blue line. The (visible part of the) kernel ball is filled in with olive green. Reproduced from [16].

As we have mentioned, the feature lines of comparatively small scales are usually "smoothed out" when extracted with curvature. Fortunately, as integral invariants can be defined over smaller neighborhoods than differential invariants, we can resort to them for extracting feature lines of small scales, that is, to derive the geometric property and critical direction required for feature-line extraction from the viewpoint of integral invariants.

We focus on ambient occlusion or "horizon-based" occlusion and its corresponding critical direction, that is, the direction of maximum occlusion. They prove more reliable than surface area or volume descriptors and corresponding critical directions defined over ball or sphere neighborhoods (see Appendix B.2). A practical definition of the amount of ambient occlusion $A(\mathbf{p})$ at a point $\mathbf{p}$ with a surface normal vector $\mathbf{n}(\mathbf{p})$ usually takes the form

$$A(\mathbf{p}) = \frac{1}{\pi} \int_{\Omega} (1 - V(\mathbf{p}, \omega))(\mathbf{n}(\mathbf{p}) \cdot \boldsymbol{\omega}) d\omega.$$

Here, V is the visibility function over the normal-oriented unit hemisphere Ω. It returns 0 if a ray starting from the point $\mathbf{p}$ in direction $\boldsymbol{\omega}$ intersects with the surface itself and returns 1 otherwise. Nevertheless, we do not have to adhere to cosine-weighted ambient occlusion for our application. Inspired by the concept of ambient occlusion and in order to distinguish it from its original definition in physics, we define the *occlusion* at the point $\mathbf{p}$ on the mesh as the ratio of the normal-oriented hemisphere with a radius r occluded by its surroundings. Instead of ray casting, we take a purely geometric approach, as the horizon is essentially determined jointly by the intersection curve of the surface and the normal-oriented hemisphere, and the central projection of visible contour edges onto the hemisphere (see Fig. 2.6), whichever has a larger elevation angle when viewed from $\mathbf{p}$ at a certain azimuthal angle. The horizon map h at $\mathbf{p}$ becomes a function that associates an azimuthal angle with an elevation angle. Letting elevation angle at azimuthal angle θ be $h_{\mathbf{p}}(\theta)$, the amount of occlusion at $\mathbf{p}$ can be obtained by integrating the horizon map:

$$Occlusion = \frac{1}{2\pi}\int_0^{2\pi} \sin(h_{\mathbf{p}}(\theta))d\theta.$$

A discrete horizon map can be created by sampling from the horizon at N_h azimuthal directions. We can approximate the amount of occlusion at $\mathbf{p}$ from its horizon map as follows:

$$Occlusion = \frac{1}{N_h}\sum_{i=1}^{N_h} \sin(h_{\mathbf{p}}(\frac{2\pi i}{N_h})). \tag{2.1}$$

Meanwhile, the direction of maximum occlusion can be estimated by locating the direction with maximum elevation angle from the "smoothed" horizon map. The original horizon map may consist of a series of sharp peaks or even plateaus from which it would be extremely difficult to deduce a meaningful critical direction. Therefore, we have to smooth out those peaks to figure out the roughly average direction of a cluster of summits. In particular, we smooth horizon maps by discrete Fourier transform: after discrete Fourier transform, terms of high frequencies (noises) are removed and a "smoothed" horizon map (see Fig. 2.7) can be obtained by applying inverse discrete Fourier transform.

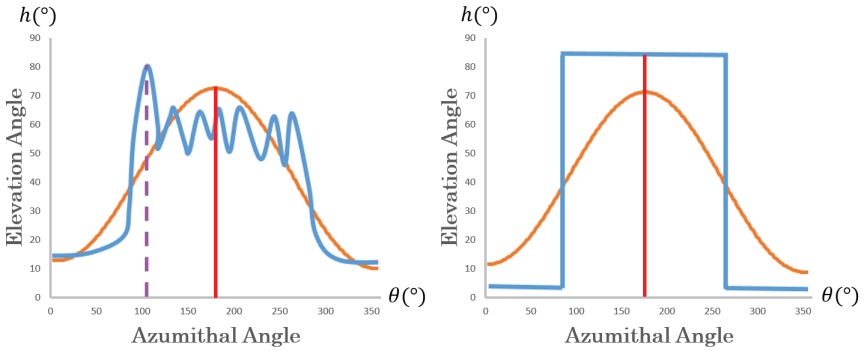

Fig. 2.7 Horizon maps before (the blue curves) and after (the orange curves) "smoothing." The direction of maximum occlusion (indicated by the red lines) is then defined as the direction with the maximum elevation angle over the "smoothed" horizon map. Note its difference from the single direction with the maximum elevation angle in the original horizon map (the dashed line in magenta). Reproduced from [16].

To extract feature lines with occlusion, for a vertex (a green dot in Fig. 2.8) adjacent to a sharp crease, its neighborhood size should be large enough to reach its nearest, most "salient" adjacent edge (highlighted in orange in Fig. 2.8) to ensure that the critical direction (a red arrow) at the vertex points to the edge. A local neighborhood size smaller than that, such as that indicated by the purple circle around the vertex in Fig. 2.8, would leave the vertex without a meaningful critical direction or point to some unpredictable direction as a result of trivial perturbation of surface normals. We refer to neighborhood size determined in this way as the *minimum local neighborhood size*.

Appendix B.1 presents a framework for defining critical directions from the viewpoint of integral invariants.

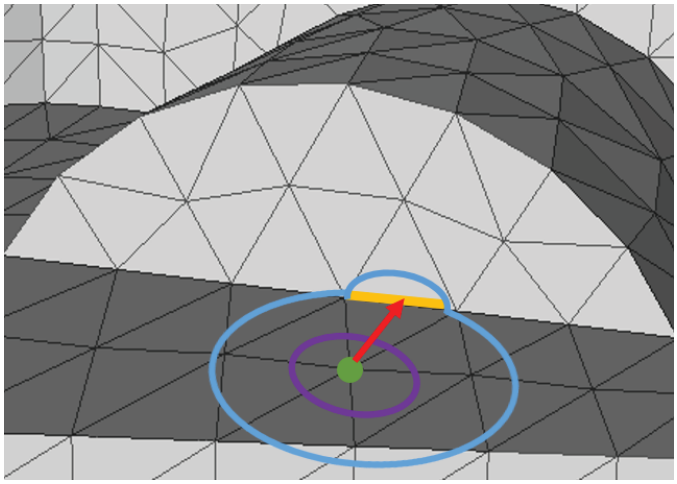

Fig. 2.8 The minimum local neighborhood size (the blue circle) ensures the definition of a meaningful critical direction at the vertex (highlighted in green) adjacent to a sharp crease and consequently enables extraction of feature lines over the crease. Reproduced from [16].

Chapter 3

A novel line-tracing strategy

3.1 Line-tracing in related work

As mentioned in Section 2.1, a common method [11, 26, 17] for locating the local extrema of a certain geometric property I (e.g. principal curvatures) over mesh edges is locating the zero-crossings of the directional derivatives of the property along corresponding critical directions $\mathbf{t}$'s (e.g. principal directions of curvature).

3.1.1 Their mechanism

Intuitively, if critical directions $\mathbf{t}'$s along a mesh edge undergo an arbitrarily radical change, it is impossible to predict how $\mathbf{t}$ behaves over the edge or if the edge contains a local maximum. Also, as critical directions at mesh vertices are supposed to be roughly perpendicular to nearby feature lines (see Fig. 2.1), if a mesh edge is crossed by a roughly "straight" feature line, critical directions at both end vertices of the edge should be perpendicular to it (see Fig. 2.2), that is, the critical directions at both end vertices should be parallel.

For this reason, we limit the scope of our discussion for the moment to cases in which critical directions at both end vertices and anywhere between them on an edge are relatively constant

(ignoring vector orientation). Ideally, we assume that the critical directions $\mathbf{t}(\mathbf{u})$ and $\mathbf{t}(\mathbf{v})$ at both end vertices $\mathbf{u}$, $\mathbf{v}$ of an edge $[\mathbf{u}, \mathbf{v}]$ are roughly parallel, pointing in opposite directions: $\mathbf{t}(\mathbf{u})$, $\mathbf{t}(\mathbf{v})$ and the edge $[\mathbf{u}, \mathbf{v}]$ lie approximately in the same plane, which is also an unspoken starting premise of the aforementioned existing methods.

There can be no more than one maximum over an edge. Typically, detecting a local maximum over the edge $[\mathbf{u}, \mathbf{v}]$ is a two-step process:

(i) Detecting the zero-crossing of the directional derivative $D_{\mathbf{t}}I$,

(ii) Verifying if the extremum is a maximum.

Note that the critical directions $\mathbf{t}'$s at $\mathbf{u}$ and $\mathbf{v}$ are oriented to point to directions with positive derivatives along which the geometric property is supposed to increase. For (i) and (ii), Ohtake et al. [26] accordingly adopted a 2-step maxima test:

- $\mathbf{t}(\mathbf{u}) \cdot \mathbf{t}(\mathbf{v}) < 0$

Intuitively, if an edge $[\mathbf{u}, \mathbf{v}]$ contains a local maximum or, equivalently, a feature line crosses the edge, $\mathbf{t}(\mathbf{u})$ and $\mathbf{t}(\mathbf{v})$ should "point to" each other (see Fig. 3.1 left), entailing that they point in opposite directions simply because, if they point in roughly the same direction (see 3.1 right), the nearby feature line (which is expected to lie within the shaded area) is unlikely to cross the edge $[\mathbf{u}, \mathbf{v}]$ in the first place.

- $\mathbf{t}(\mathbf{u}) \cdot (\mathbf{v} - \mathbf{u}) > 0$ and $\mathbf{t}(\mathbf{v}) \cdot (\mathbf{u} - \mathbf{v}) > 0$

"Pointing to" each other also implies that the angle between the critical direction and the edge at both end vertices, α and β, is acute. This test ensures that the extremum attained over the edge $[\mathbf{u}, \mathbf{v}]$ is indeed a maximum, ruling out cases when $\mathbf{t}(\mathbf{u})$ and $\mathbf{t}(\mathbf{v})$ point to different nearby feature lines (see Fig. 3.2 right), in which there are clearly no intersections between feature lines and the edge.

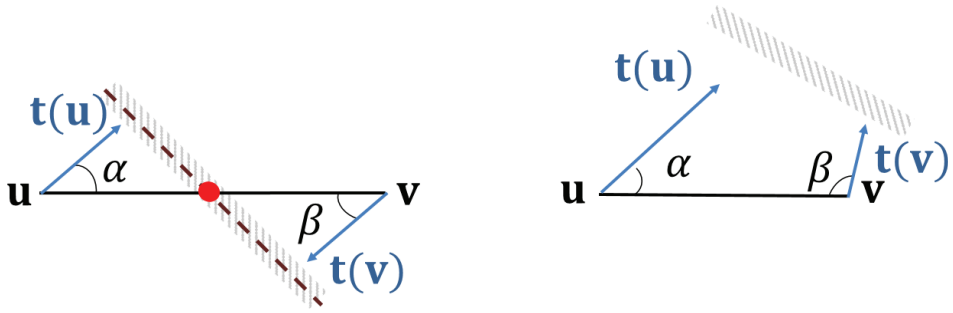

Fig. 3.1 A test for detecting the zero-crossing of $D_{\mathbf{t}}I$ shared by both [26] and [17], which checks if critical directions at both end vertices are pointing in opposite directions.

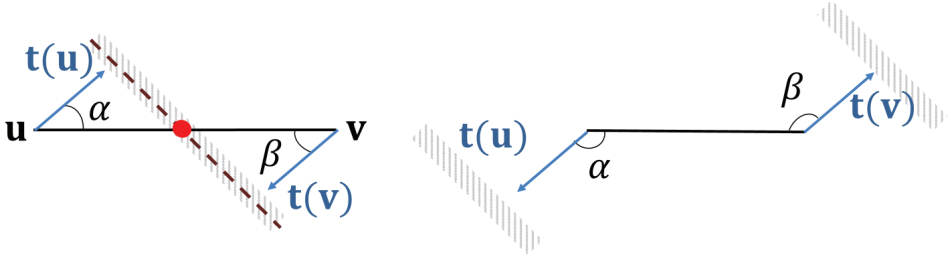

Fig. 3.2 A test for maxima suggested in [26] that checks if both α and β are acute.

Note that in both [26] and [17], for (i), the existence of a zero-crossing of $D_{\mathbf{t}}I$, the directional derivative of the geometric property I along the critical direction $\mathbf{t}$, is confirmed by checking if the angle between $\mathbf{t}(\mathbf{u})$ and $\mathbf{t}(\mathbf{v})$ is obtuse. After that, Judd et al. [17] devised a slightly different test that eliminates minima after zero-crossing lines are drawn. They dropped a perpendicular from each vertex to the zero-crossing line (a line connecting two zero-crossings in a mesh triangle) and checked if the angle between the critical direction to the perpendicular at each vertex was acute (see Fig. 3.3 down). This test proves more robust than the one suggested by Ohtake et al. [26], which in effect substitutes the edge direction for the perpendicular.

The location of the zero-crossing is then interpolated using the magnitude of the directional derivatives at both vertices:

$$\mathbf{p} = \frac{\left\|D_{\mathbf{t}}I(\mathbf{u})\right\|\mathbf{v} + \left\|D_{\mathbf{t}}I(\mathbf{v})\right\|\mathbf{u}}{\left\|D_{\mathbf{t}}I(\mathbf{u})\right\| + \left\|D_{\mathbf{t}}I(\mathbf{v})\right\|}. \tag{3.1}$$

3.1.2 Their defects

Evidently, when the angles (or their supplementary angles) between the critical direction and the edge at both end vertices **u** and **v** are relatively small, the aforementioned maxima tests can be trusted to make the right decision about the existence of a local maximum over the edge [**u**, **v**] (see Fig. 3.3 top).

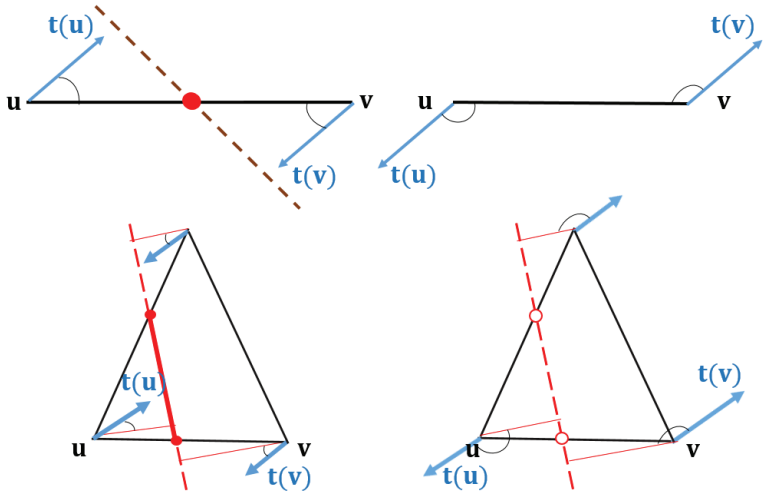

Fig. 3.3 Tests for maxima: [26] checked if both angles between critical directions and the edge were acute (top); [17] checked if the angle between the critical direction and the corresponding perpendicular was acute at all three vertices of a mesh triangle (down). Both tests make the right decision when these angles (or their supplementary angles) are relatively small.

Cause of fragmentation

However, ridge-valley lines fragment when at least one of the two angles between the edge and the critical directions at the end vertices is obtuse (see Fig. 3.3 top right). The maxima test of Ohtake et al. [26] becomes unreliable when the critical directions at the end vertices are almost perpendicular to the edge direction. This can be verified in Fig. 3.4 left, in which a potential maximum over [**u**, **v**] was missed. The maxima test of Judd et al. [17] is more robust. Nevertheless, line fluctuation causes failure in the maxima test, which eventually

leads to line fragmentation. Note that in Fig. 3.4 right, with a local maximum over [**u**, **v**] "pulled" away from where a potential feature line should lie (the shaded region), a zero-crossing line almost parallel to **t**(**u**) is generated and later eliminated due to the obtuse angle between the critical direction and the corresponding perpendicular at **u**, which results in maxima test failure.

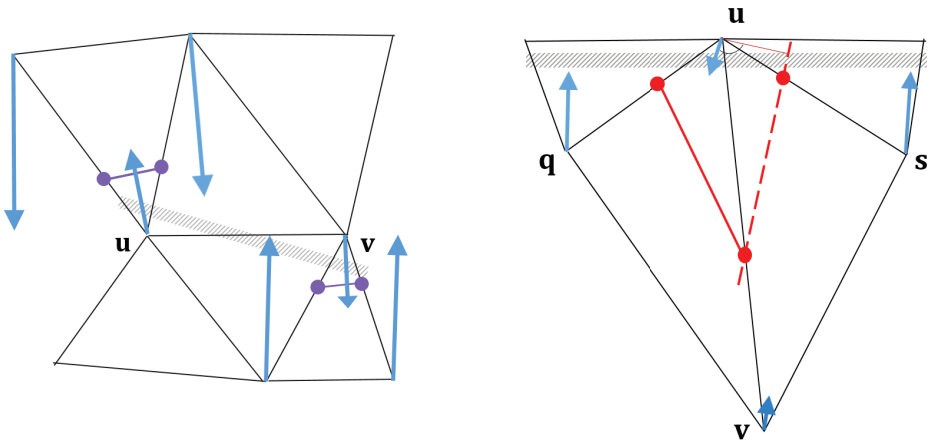

Fig. 3.4 Maxima tests of [26] and [17] may result in fragmentation (left and right) and fluctuation (right) of lines.

Cause of fluctuation

So far, we have discussed how existing algorithms determine the existence of local maxima over mesh edges, which tends to result in line fragmentation. The location of such a local maximum over an edge [**u**, **v**], however, is completely based on the magnitude of directional derivatives $D_{\mathbf{t}}I$ at both end vertices (see Eq. 3.1), which can be defined as the amount of change in I per unit of distance along **t**.

Clearly, Eq. 3.1 is based on the unstated assumption that the change of $D_{\mathbf{t}}I$ is approximately linear over the entire edge [**u**, **v**], implying that critical directions $\mathbf{t}'$s along an edge are relatively constant (ignoring vector orientation) (see Section 3.1.1).

The red curves in Fig. 3.5 and Fig. 3.6 illustrate how the geometric property I changes along a mesh edge [**u**, **v**]. If I attains its local maximum somewhere along the curve (corresponding to the red dot), the directional derivative $D_{\mathbf{t}}I$ at that point should be zero. We assume that the change of the magnitude of $D_{\mathbf{t}}I$ is approximately linear over the entire edge [**u**, **v**]. Then, according to Eq. 3.1, the local maximum should be located closer to the vertex with a smaller $D_{\mathbf{t}}I$ (in terms of magnitude).

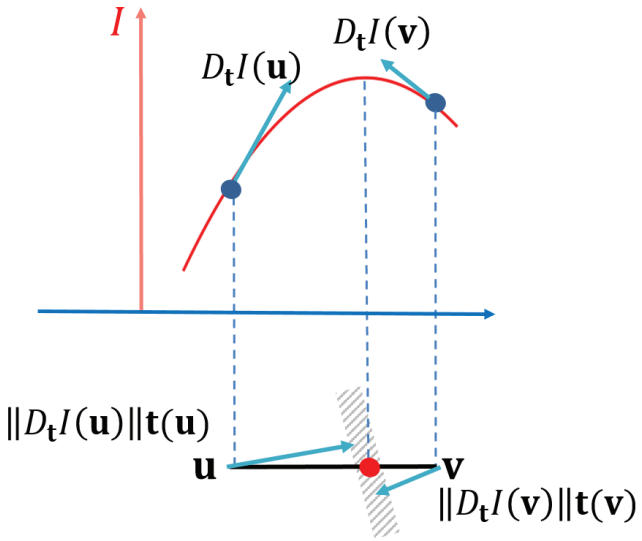

Fig. 3.5 When the unstated assumption holds, the location of a local maximum over a mesh edge [**u**, **v**] can be estimated with Eq. 3.1.

Consider the case when a *potential feature line* (which is expected to lie roughly in the shaded region in Fig. 3.4 right) is almost parallel to the mesh edges.

On both edges [**p**, **q**] and [**p**, **s**], $D_{\mathbf{t}}I$ at **p** is smaller than $D_{\mathbf{t}}I$ at the other end vertex. As a result, the local maxima over these two edges should be located closer to **p**. Nonetheless, the location of the maximum over [**u**, **v**] may be "pulled" toward **v** when **v** is associated with a directional derivative smaller (in terms of magnitude) than both **q** and **s**. This could occur when **v** is far enough from the potential feature line that the change of $D_{\mathbf{t}}I$ in the vicinity of **v** cannot be regarded as linear (see Fig. 3.6); that is, when the aforementioned assumption does not hold any more.

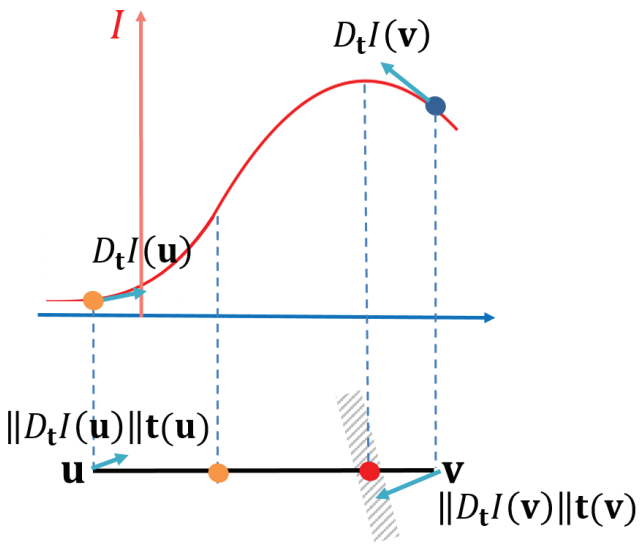

Fig. 3.6 When the unstated assumption does not hold, estimation of a local maximum over a mesh edge [**u**, **v**] based on Eq. 3.1 can no longer be trusted. Dots in orange and red correspond to the estimated and expected locations of the local maximum, respectively.

Noticeably, in both maxima tests, the existence of local maxima depends solely on critical directions, and the locations of local maxima rely solely on the magnitudes of directional derivatives. While the critical direction **t** indicates the direction along which the geometric property I is supposed to increase in value, the directional derivative yields the rate of that increase along **t**. Observation of frequent fluctuation and fragmentation suggests the necessity of incorporating another indicator into the line-tracing strategy.

3.1.3 An attempt to reduce fragmentation

Yoshizawa et al. [37] pointed out that typical ridge-valley line-extraction algorithms [26] tend to result in line fragmentation. They offer a "gap-jumping" strategy for connecting close, disconnected lines in situations similar to those demonstrated in Fig. 3.7, when the angles between end-segments and the segment connecting the two end-points are within a certain range ($\alpha \leq \frac{\pi}{3}, \beta \leq \frac{\pi}{3}, \gamma \leq \frac{\pi}{2}$).

This requirement is likely to be satisfied when two end-segments are roughly parallel. However, not only are these thresholds for angles determined empirically, but nearby line segments are connected almost exclusively according to their positions.

That is to say, although this strategy may be able to produce pleasing results on some occasions, it fails to deal with the root cause that gives rise to fragmentation, which we have covered in depth, and hence lacks the theoretical ground to be a fundamental cure.

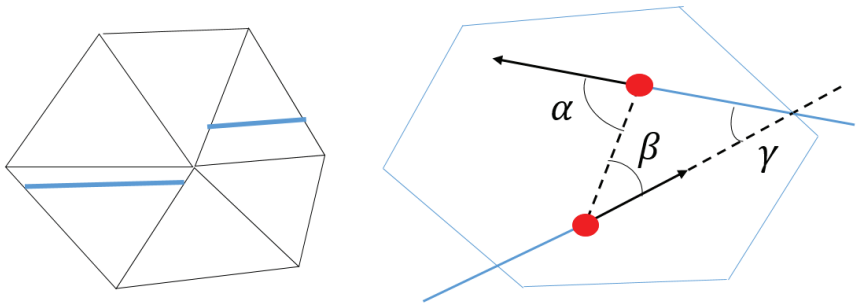

Fig. 3.7 Left: a situation in which it is desirable to connect two nearby lines. Right: angles α, β, γ are used to measure when gap-jumping is necessary. Reproduced from Fig. 4 of [37].

3.2 Proposed method

Since ridge-like lines turn into valley-like lines by reversing the surface normals, without loss of generality, the discussion hereafter focuses on extracting valley-like feature lines with different sets of geometric properties and critical directions. Following the mechanism of existing ridge-valley-like line extraction [26, 17], the feature lines of interest can be defined as the loci of points where certain geometric properties attain their maxima.

3.2.1 The idea

If a mesh edge [**u**, **v**] contains a local maximum—that is, a feature line crosses the edge—it is self-evident that a feature line (segment) exists in the 1-ring neighborhood of both end

vertices. Recall that by definition, the critical direction **t** in the ideal case is perpendicular to the nearby feature line [15]. In other words, starting from the end vertex **v**, somewhere along **t**(**v**), the value of I should attain a maximum as we reach the nearby feature line at location $\mathbf{p_v}$. Needless to say, $\mathbf{p_v}$ is the *nearest maximum* accessible from **v** and is located in a small-enough neighborhood centered on **v**. Similarly, for vertex **u**, its nearest maximum $\mathbf{p_u}$ should be somewhere along **t**(**u**) (see Fig. 3.8 left).

Since both $\mathbf{p_u}$ and $\mathbf{p_v}$ are located on the nearby feature line, the line passing through them can be considered the approximate feature line in the vicinity of the mesh edge [**u**, **v**]. Once again, assuming **t**(**u**) and **t**(**v**) are roughly parallel, whether edge [**u**, **v**] contains a local maximum now becomes whether edge [**u**, **v**] intersects with the line passing through $\mathbf{p_u}$ and $\mathbf{p_v}$.

Therefore, instead of the linear interpolation of directional derivatives, we consider locating maxima over mesh edges via the linear interpolation of the nearest maxima of mesh vertices. Our proposed method is premised on the following assumption:

- For a mesh vertex **v**, the nearest maximum $\mathbf{p_v}$ in its vicinity along its critical direction **t**(**v**) can be located.

Since critical directions **t**'s along edge [**u**, **v**] are roughly parallel, $\mathbf{p_u}$, $\mathbf{p_v}$ and the edge can be considered approximately coplanar. If $\mathbf{p_u}$ and $\mathbf{p_v}$ are not on the same side of the edge [**u**, **v**] (see Fig. 3.8 left), the local maximum over the edge can then be located at the intersection of the line segment [$\mathbf{p_u}$, $\mathbf{p_v}$] and the edge [**u**, **v**]. If, on the contrary, $\mathbf{p_u}$ and $\mathbf{p_v}$ are located on the same side of [**u**, **v**], the only possible intersection of the edge [**u**, **v**] and the line passing through $\mathbf{p_u}$ and $\mathbf{p_v}$ falls on the extension of the edge (see Fig. 3.8 right); that is, the edge [**u**, **v**] does not contain a local maximum.

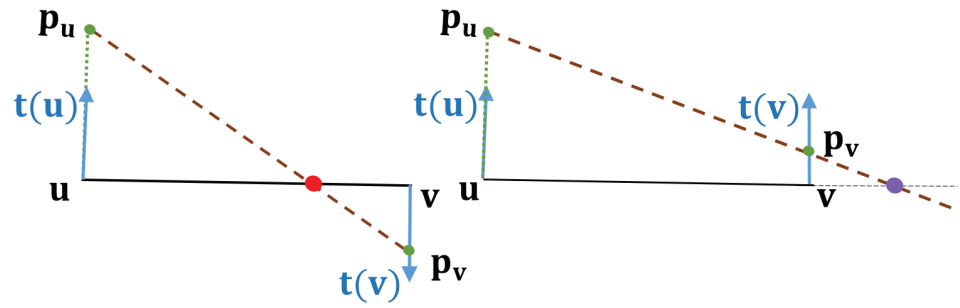

Fig. 3.8 Detecting and locating local maxima over a mesh edge via interpolating the nearest maxima of vertices when critical directions at end vertices are parallel. A local maximum exists over the edge [**u**, **v**] when $\mathbf{p_u}$ and $\mathbf{p_v}$ are located on different sides of the edge (left). Otherwise, the edge [**u**, **v**] does not contain a local maximum (right).

By introducing the locations of the nearest maxima for mesh vertices, our proposed method for line-tracing is expected to generate lines with less fragmentation (see Fig. 3.9 right) and fluctuation (see Fig. 3.10 right) than existing algorithms.

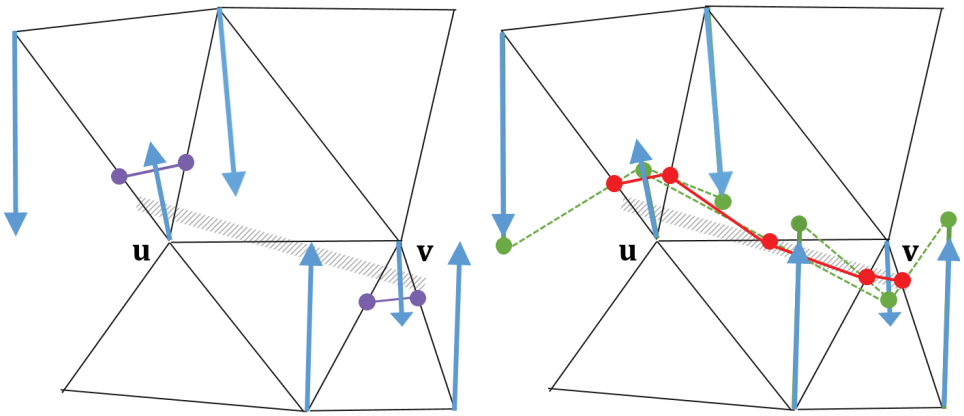

Fig. 3.9 Left: Failure in the maxima test of Ohtake et al. resulted in fragmentation of lines. Right: Our proposed method is expected to accurately detect and locate the local maximum over edge [**u**, **v**]. The shaded region indicates where a potential feature line is expected to lie. The critical directions at mesh vertices are illustrated with blue arrows. The length of a blue arrow is proportionate to the magnitude of $D_{\mathbf{t}}I$ at that vertex. Green dots indicate the estimated nearest maxima of mesh vertices.

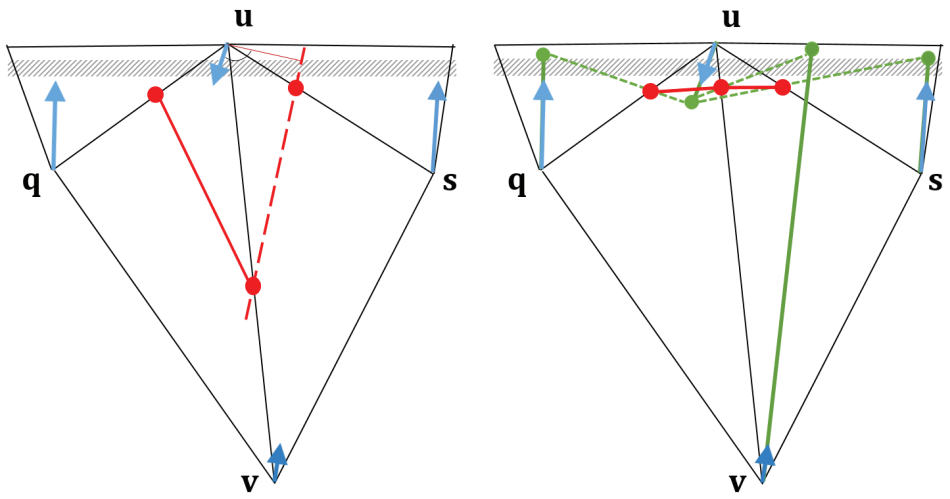

Fig. 3.10 Red lines specify feature lines generated with the strategy of Judd et al.[17] (left) and our proposed method (right).

3.2.2 The algorithm for our line-tracing strategy

Fig. 3.11 demonstrates the pipeline of our line-tracing strategy. We now address the details of the algorithm.

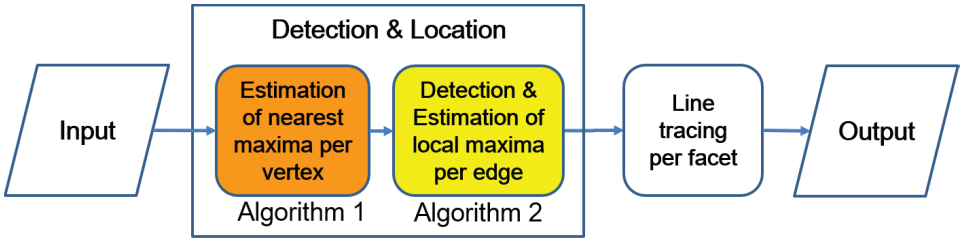

Fig. 3.11 Pipeline of our proposed method.

Estimation of the nearest maxima for mesh vertices

By definition, if a feature line exists in the vicinity of a mesh vertex $\mathbf{v}$ along the critical direction $\mathbf{t}(\mathbf{v})$, the geometric property I should increase (in terms of absolute value) until

it reaches a peak value at the point closest to $\mathbf{v}$ on the feature line and decrease thereafter. Therefore, it is intuitive to fit a quadratic function $f(s)$ to such a geometric property along the critical direction $\mathbf{t}$ over the vicinity of $\mathbf{v}$. Assuming $\mathbf{t}(\mathbf{v})$ is a unit vector, for N_f vertices $\mathbf{v}_1$,$\mathbf{v}_2$,...,$\mathbf{v}_{N_f}$ in the neighborhood of $\mathbf{v}$ and itself, fit $f(s)$ to the geometric property I by minimizing the following cost function:

$$\sum_{i=0}^{N_f} w_i \left(I(\mathbf{v}_i) - f(s_i)\right)^2, \tag{3.2}$$

where $\mathbf{v}_0 = \mathbf{v}$, $s_i = (\mathbf{v}_i - \mathbf{v}) \cdot \mathbf{t}(\mathbf{v})$, $w_i = e^{-\frac{\|\mathbf{v}_i - \mathbf{v}\|}{\sigma}}$ (see Fig. 3.12).

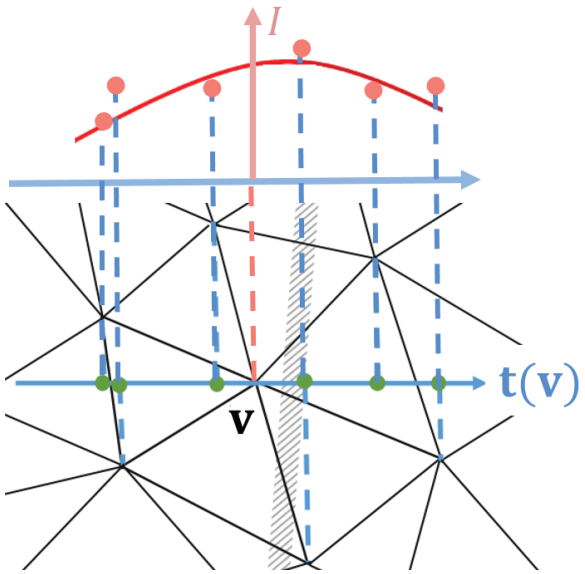

Fig. 3.12 Local fitting of $f(s)$ to the geometric property I in the vicinity of the mesh vertex $\mathbf{v}$ along $\mathbf{t}(\mathbf{v})$.

After this local fitting of $f(s)$ to the geometric property, we can locate the nearest maximum $\mathbf{p}_\mathbf{v}$ along $\mathbf{t}(\mathbf{v})$, which maximizes $f(s)$, $s \in [s_{min}, s_{max}]$, where s_{min} and s_{max} are the minimum and maximum of s_i ($i = 0, ..., N_f$), respectively. In our implementation, the size of the neighborhood over which such fitting is performed depends on the type of geometric property utilized in feature-line extraction. In other words, we plug into 1-ring and 2-ring

neighborhoods when occlusion and curvature are employed, respectively. By default, σ takes the value of the Euclidean distance between the mesh vertex and its nearest adjacent vertex.

Algorithm 1 presents the details of the procedure for estimating the nearest maxima for mesh vertices.

Algorithm 1: Estimating the nearest maxima for mesh vertices

input : A polygonal mesh, geometric properties I's, and critical directions $\mathbf{t}$'s defined at all mesh vertices
output: Nearest maxima of all mesh vertices

foreach *vertex* $\mathbf{v}$ *of the mesh* **do**
  $X \leftarrow \{\mathbf{v}_0, \mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_{N_f}\}$;
  Fit $f(s)$ to I over X (quadratically) by minimizing Eq. 3.2;
  if f *is convex* **then**
    Fit $f(s)$ to I over X (linearly) by minimizing Eq. 3.2;
  Find and store the nearest maximum $\mathbf{p}_\mathbf{v}$ of $\mathbf{v}$ along $\mathbf{t}(\mathbf{v})$ over X where $f(s)$, $s \in [s_{min}, s_{max}]$, attains its maximum;

Detection & estimation of local maxima over mesh edges

Similar to existing methods, our proposed method is based on the assumption that critical directions $\mathbf{t}$'s are roughly parallel along an edge $[\mathbf{u}, \mathbf{v}]$; that is, $\mathbf{t}(\mathbf{u})$ and $\mathbf{t}(\mathbf{v})$ are parallel so that the line segment $[\mathbf{p_u}, \mathbf{p_v}]$ and the mesh edge $[\mathbf{u}, \mathbf{v}]$ are approximately coplanar.

Nevertheless, $[\mathbf{p_u}, \mathbf{p_v}]$ and $[\mathbf{u}, \mathbf{v}]$ often do not lie exactly in the same plane, ruling out the possibility of any intersection between the two. Assuming that both $\mathbf{t}(\mathbf{u})$ and $\mathbf{t}(\mathbf{v})$ are unit vectors, we rotate critical directions $\mathbf{t}(\mathbf{u})$ and $\mathbf{t}(\mathbf{v})$ slightly by the same amount until they become

$$\mathbf{c} = \frac{\mathbf{t}(\mathbf{u}) - \mathbf{t}(\mathbf{v})}{\|\mathbf{t}(\mathbf{u}) - \mathbf{t}(\mathbf{v})\|}, \quad \mathbf{d} = \frac{\mathbf{t}(\mathbf{v}) - \mathbf{t}(\mathbf{u})}{\|\mathbf{t}(\mathbf{v}) - \mathbf{t}(\mathbf{u})\|} = \text{-}\,\mathbf{c}, \tag{3.3}$$

respectively. Now **c** and **d** become parallel, pointing in opposite directions (see Fig. 3.13 right). The points $\mathbf{p_u}'$ and $\mathbf{p_v}'$, the projection of $\mathbf{p_u}$ and $\mathbf{p_v}$ onto **c** and **d**, respectively, are checked to see if the mesh edge [**u**, **v**] contains a local maximum (see Fig. 3.13 left).

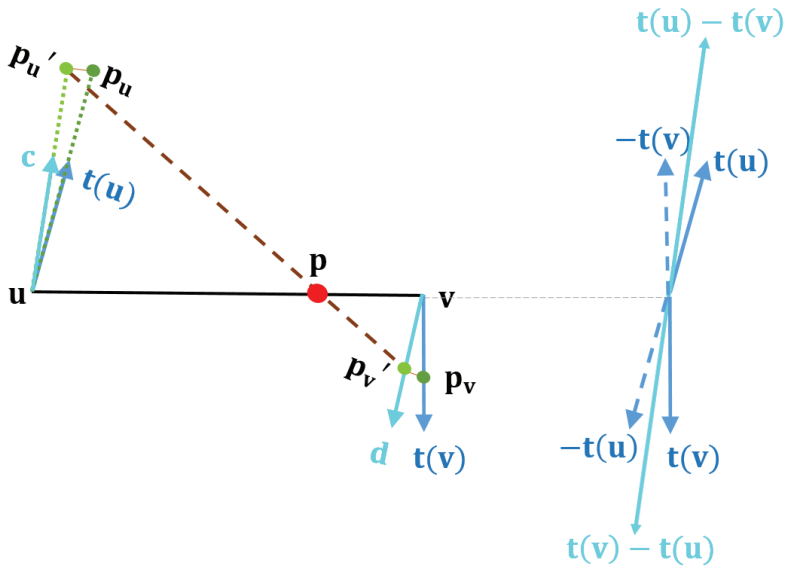

Fig. 3.13 When $\mathbf{p_u}$, $\mathbf{p_v}$ and the mesh edge [**u**, **v**] are not coplanar, find a local maximum over the edge [**u**, **v**] at the intersection of [**u**, **v**] and [$\mathbf{p_u}'$, $\mathbf{p_v}'$] (left). **t**(**u**) and **t**(**v**) are rotated slightly to a common direction, which is equivalent to the average of **t**(**u**) and **- t**(**v**) (right).

A local maximum over the edge [**u**, **v**] is then detected and determined as illustrated in Fig. 3.8. If the edge contains a local maximum **p**, obviously $\triangle \mathbf{p_u}'\mathbf{pu} \sim \triangle \mathbf{p_v}'\mathbf{pv}$. The similarity of these two triangles leads to

$$\frac{\|\mathbf{p}-\mathbf{u}\|}{\|\mathbf{p}-\mathbf{v}\|} = \frac{\|\mathbf{p_u}'-\mathbf{u}\|}{\|\mathbf{p_v}'-\mathbf{v}\|} = \lambda.$$

We can then approximate the location of **p** with linear interpolation.

$$\mathbf{p} = \frac{\lambda}{1+\lambda}\mathbf{v} + \frac{1}{1+\lambda}\mathbf{u} = \frac{\|\mathbf{p_u}'-\mathbf{u}\|\,\mathbf{v} + \|\mathbf{p_v}'-\mathbf{v}\|\,\mathbf{u}}{\|\mathbf{p_u}'-\mathbf{u}\| + \|\mathbf{p_v}'-\mathbf{v}\|}. \tag{3.4}$$

Algorithm 2 specifies the procedure of estimating local maxima over mesh edges in detail.

Algorithm 2: Estimating local maxima over mesh edges

input : A polygonal mesh, geometric properties I's, and critical directions $\mathbf{t}$'s defined at all mesh vertices and the estimated nearest maxima of all mesh vertices

output: Local maxima over mesh edges

foreach *edge* $[\mathbf{u}, \mathbf{v}]$ *of the mesh* **do**
- $\mathbf{p_u} \leftarrow$ location of the nearest maximum of $\mathbf{u}$;
- $\mathbf{p_v} \leftarrow$ location of the nearest maximum of $\mathbf{v}$;
- Rotate $\mathbf{t}(\mathbf{u})$ and $\mathbf{t}(\mathbf{v})$ to $\mathbf{c}$ and $\mathbf{d}$, respectively, based on Eq. 3.3 so that they become strictly parallel;
- $\mathbf{p_u}' \leftarrow$ Projection of $\mathbf{p_u}$ onto $\mathbf{c}$;
- $\mathbf{p_v}' \leftarrow$ Projection of $\mathbf{p_v}$ onto $\mathbf{d}$;
- **if** *line segment* $[\mathbf{p_u}', \mathbf{p_v}']$ *intersects the edge* $[\mathbf{u}, \mathbf{v}]$ **then**
  - Compute and store the location of the intersection based on Eq. 3.4 as the local maximum over the edge;
- **else**
  - There is no local maximum over the edge;

Tracing lines per mesh facet

We iterate over each mesh facet to connect the local maxima over its edges, forming one line segment of a feature line. This process creates a "network" of feature lines.

For a mesh edge to contain a local maximum, the critical directions at both its end vertices should be roughly parallel, and the nearest maxima of both end vertices should lie on different sides of the edge. Therefore, notably, triangles with one local maximum over each edge are theoretically non-existent. However, the assumption of parallel critical directions along a mesh edge is often violated in the vicinity of "junctions." The topic of tracing lines in such cases will be revisited in Section 3.3.4.

3.3 Experiments

3.3.1 Evaluation criteria

A set of extracted feature lines can be regarded as a graph whose vertices correspond to points sampled over mesh edges. A single feature line corresponds to one connected component in the graph. Therefore, given such a graph, we apply the following qualitative criteria to the evaluation of feature lines:

- Average length of lines
  - The average length of all connected components in the graph, which can easily be obtained given the total length and number of lines. Intuitively, line fragmentation creates more lines and consequently a shorter average length. Longer average length indicates less fragmentation, in other words.
- Average fluctuation of lines
  - The average deviation of extracted feature lines from straight lines. Specifically, it is defined as $\frac{\sum_{i=1}^{J}(\pi-\theta_i)}{J}$, where θ_i is the angle between each pair of connected edges in the graph and J is the total number of such angles. Frequently, "wiggling" lines are obviously with larger deviation of lines on average; that is, smaller average deviation indicates less line fluctuation.

3.3.2 Rendering with geometric properties

As the term "feature-line extraction" may be used in a broad sense (see Fig. **??**) from Chapter 4, we use "rendering" instead to refer specifically to the process from detection and location of local maxima over mesh edges to line tracing over mesh facets (see Fig. 3.11).

Recall that the two building blocks of feature-line extraction are the geometric property and the critical direction. The following terms require clarification, as they will appear repeatedly in experiments starting from Chapter 3.

- Rendering with curvature
  - Rendering with the principal curvature and the principal direction of curvature;
  - Feature lines to be extracted are the loci of the points where the principal curvature attains local extrema along the principal direction of curvature.
- Rendering with occlusion
  - Rendering with the occlusion and the direction of maximum occlusion;
  - Feature lines to be extracted are the loci of the points where the occlusion attains local maxima along the direction of maximum occlusion.

3.3.3 Results

Fig. 3.14 and Fig. 3.15 compare two strategies for tracing valley lines. A model is rendered with valley lines based on the same pair of geometric property and critical direction but with different strategies for tracing local maxima over mesh edges.

Fig. 3.14 employs a differential invariant, the principal curvature, and the principal direction of curvature, while Fig. 3.15 is based on the occlusion and the direction of maximum occlusion. In both figures, the results generated by tracing the zero-crossings of directional derivatives are shown on the left, and lines traced via our proposed method—interpolating the nearest maxima of mesh vertices—are shown on the right. Directional derivatives of occlusion are defined similarly to the directional derivatives of view-dependent curvature in [17].

Our proposed method can generate coherent lines without fragmentation (outlined with red boxes) or fluctuation (outlined with blue boxes) in areas where typical ridge-valley line-extraction algorithms, such as that of [26], fail. Specifically, our proposed method is effective under unfavorable conditions, such as when critical directions are almost perpendicular to mesh edges (see Fig. 3.14 right).

Fig. 3.14 The casting model rendered with the principal curvature and the principal direction of curvature: valley lines generated with rtsc by connecting zero-crossings of derivatives of curvature (left); valley-like feature lines generated by interpolating the nearest maxima (right).

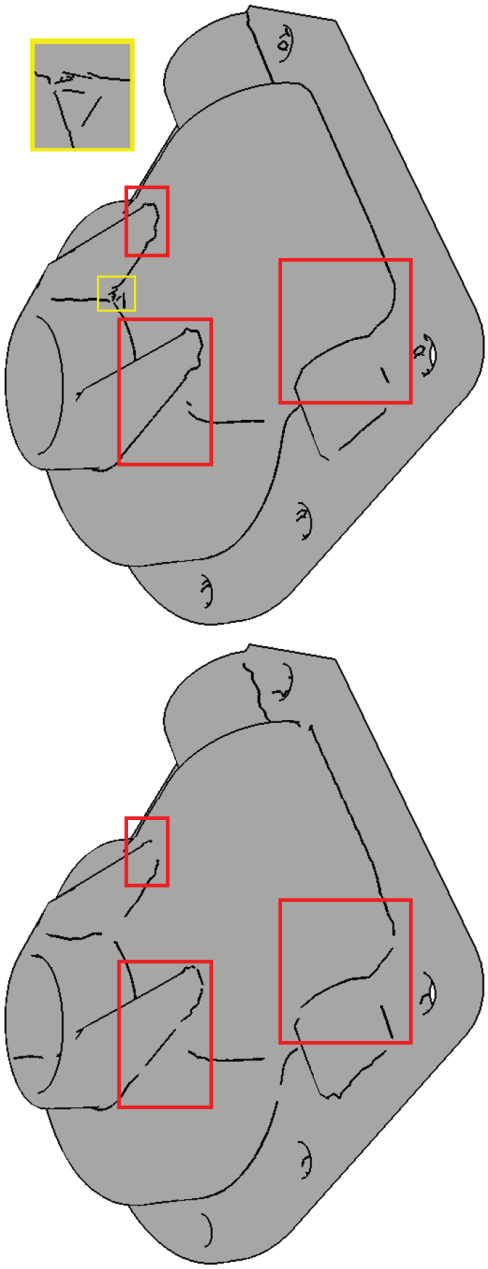

Fig. 3.15 The casting model rendered with the occlusion and the direction of maximum occlusion: valley lines generated by locating zero-crossings of directional derivatives over edges (left); valley-like feature lines generated by interpolating the nearest maxima (right). Artifacts in the vicinity of a "junction" (yellow box) will be discussed in Section 3.3.4.

Fig. 3.16, Fig. 3.17, and Fig. 3.18 illustrate a few more comparison examples [1]. Our proposed method generally outperforms existing methods when expected feature lines almost coincide with mesh edges (see the red boxes in all the corresponding figures), that is, when critical directions are almost perpendicular to mesh edges. As Table. 3.1 demonstrates, in most cases, it generates longer lines with less fluctuation on average.

Table 3.1 Comparison of line-tracing strategies

Models	Number of lines		Average length		Average Fluctuation	
	rtsc	ours	rtsc	ours	rtsc	ours
casting (Fig. 3.14)	46	32	0.305	0.503	0.271	0.239
jack o'lantern (Fig. 3.16)	138	81	8.651	15.201	0.125	0.111
pedestal (Fig. 3.17)	34	29	19.206	23.524	0.0668	0.0720
moai (Fig. 3.18)	71	24	23.921	72.566	0.173	0.160

3.3.4 Discussion

An evident limitation of this strategy is its assumption that critical directions at both end vertices are roughly parallel. As visible in Fig. 3.16–Fig. 3.19, in which edges violating the assumption are highlighted in orange, such edges are mostly located near "junctions" of multiple valleys or randomly over almost flat regions.

Naturally, our proposed line-tracing strategy is not applicable to cases of critical directions changing radically along an edge, a limitation shared by existing feature-line extraction algorithms. This could occur when the edge is in the vicinity of a potential "junction"; that is, there are actually more than one nearby valleys. Our strategy may result in undesirable artifacts, as we can observe in Fig. 3.15 right (bounded by a yellow box).

[1] The models are from https://free3d.com (jack o'lantern, moai) and https://www.turbosquid.com (Roman pedestal).

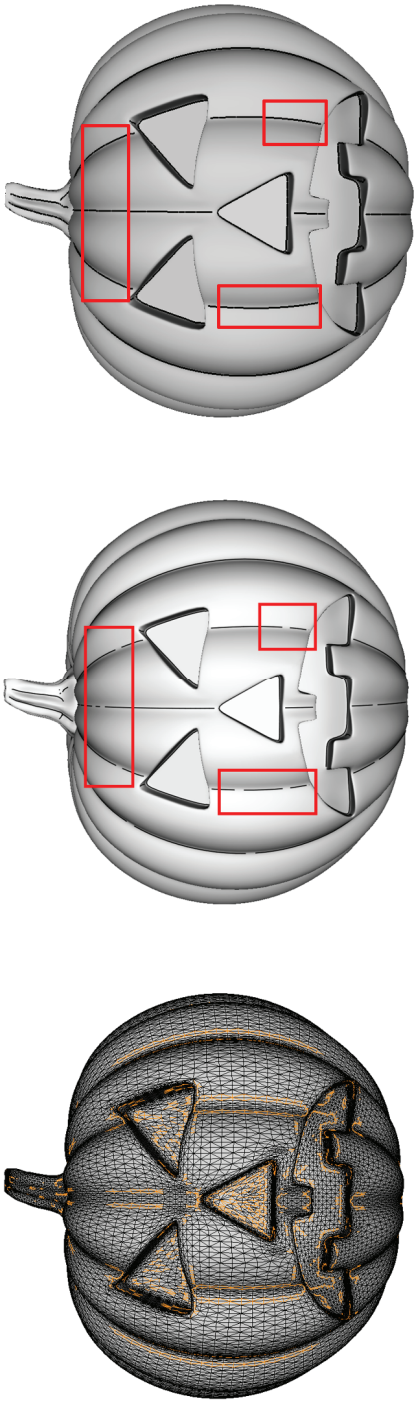

Fig. 3.16 The jack o'lantern model (left) rendered with the principal curvature and its direction: valley lines generated with rtsc by connecting zero-crossings of derivatives of curvature (middle); valley-like feature lines generated by interpolating the nearest maxima (right). Mesh edges whose critical directions at end vertices are far from being parallel (with a difference in critical directions larger than $\frac{\pi}{3}$) are highlighted in orange (left).

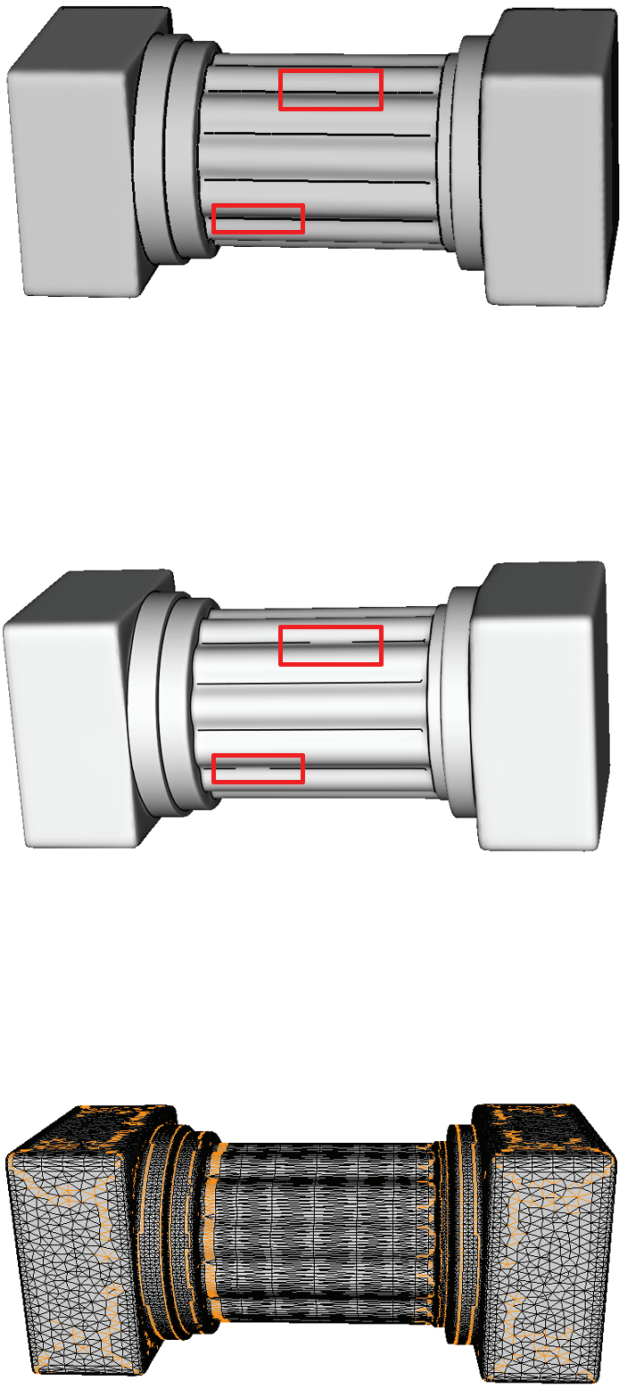

Fig. 3.17 The Roman pedestal model (left) rendered with the principal curvature and its direction: valley lines generated with rtsc by connecting zero-crossings of derivatives of curvature (middle); valley-like feature lines generated by interpolating the nearest maxima (right). Mesh edges whose critical directions at end vertices are far from being parallel (with a difference in critical directions larger than $\frac{\pi}{3}$) are highlighted in orange (left).

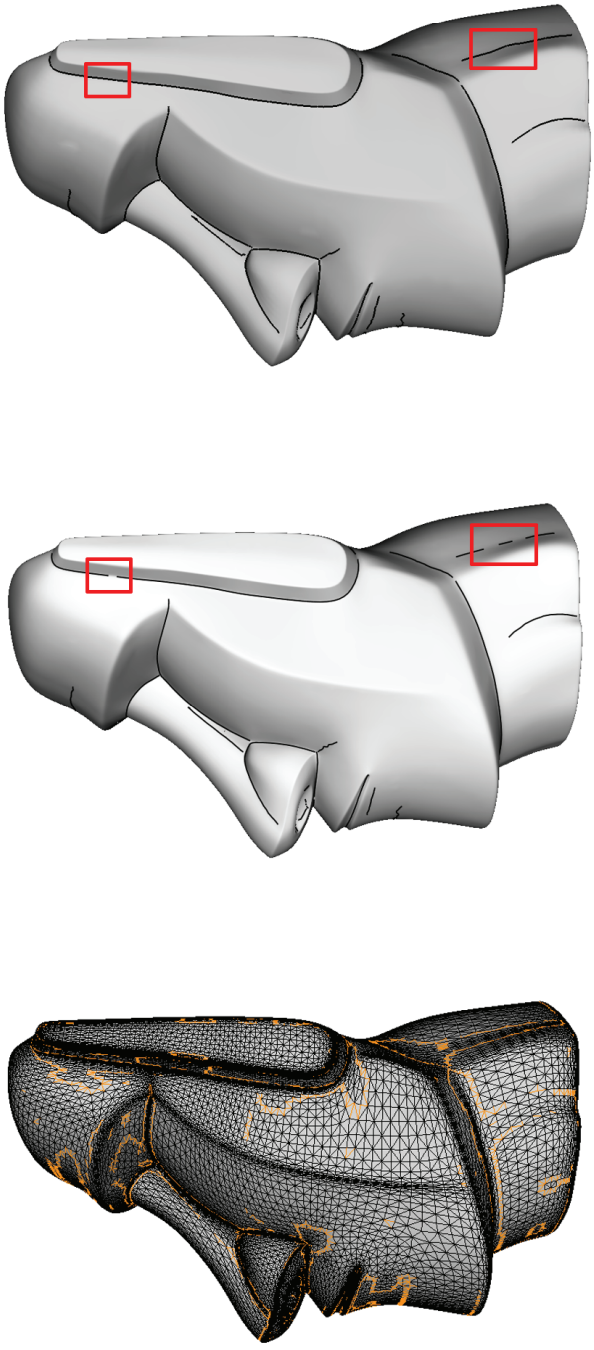

Fig. 3.18 The moai model (left) rendered with the principal curvature and its direction: valley lines generated with rtsc by connecting zero-crossings of derivatives of curvature (middle); valley-like feature lines generated by interpolating the nearest maxima (right). Mesh edges whose critical directions at end vertices are far from being parallel (with a difference in critical directions larger than $\frac{\pi}{3}$) are highlighted in orange (left).

Consider a parallel test that checks if critical directions at both end vertices of a mesh edge are far from parallel and excludes edges that violate the assumption. Fig. 3.19 compares feature lines extracted with and without the parallel test. Lines happen to join near "junctions" (see the orange boxes in Fig. 3.19) without the parallel test, with longer length on average but also more fluctuation.

Table 3.2 Comparison of feature lines in Fig. 3.19

Parallel test	w/o	w/
Number of lines	6	29
Average length	115.116	23.524
Average fluctuation	0.0978	0.0720

We now examine how well the assumption works in reality. Consider a rotated cosine wave surface patch whose ground truth for feature lines is a circle of radius π that is explicitly computable (see Fig. 3.20 left). As shown in Table 3.3 and Table 3.4, of the mesh edges crossing the feature line, which is an approximation of the ground truth with a mean squared error of 0.00688, 100.00% have differences in critical directions less than $\frac{2\pi}{9}$, 92.68% have differences less than $\frac{7\pi}{36}$, and 73.17% have differences less than $\frac{\pi}{6}$. A smaller tolerance to the differences in critical directions gives rise to more fragmentation of lines. For this surface patch, a tolerance of $\frac{2\pi}{9}$ is large enough to ensure extraction of lines without fragmentation. In our implementation, we find a tolerance of $\frac{\pi}{3}$ generates reasonably good results in terms of average length and average fluctuation for models listed in Table 3.1.

Table 3.3 Verification of the assumption for the rotated cosine wave model with low resolution (see Fig. 3.20 left)

Tolerance	$\frac{\pi}{4}$	$\frac{2\pi}{9}$	$\frac{7\pi}{36}$	$\frac{\pi}{6}$
Ratio of mesh edges satisfying the assumption among edges crossing the feature line (%)	100.00	100.00	92.68	73.17

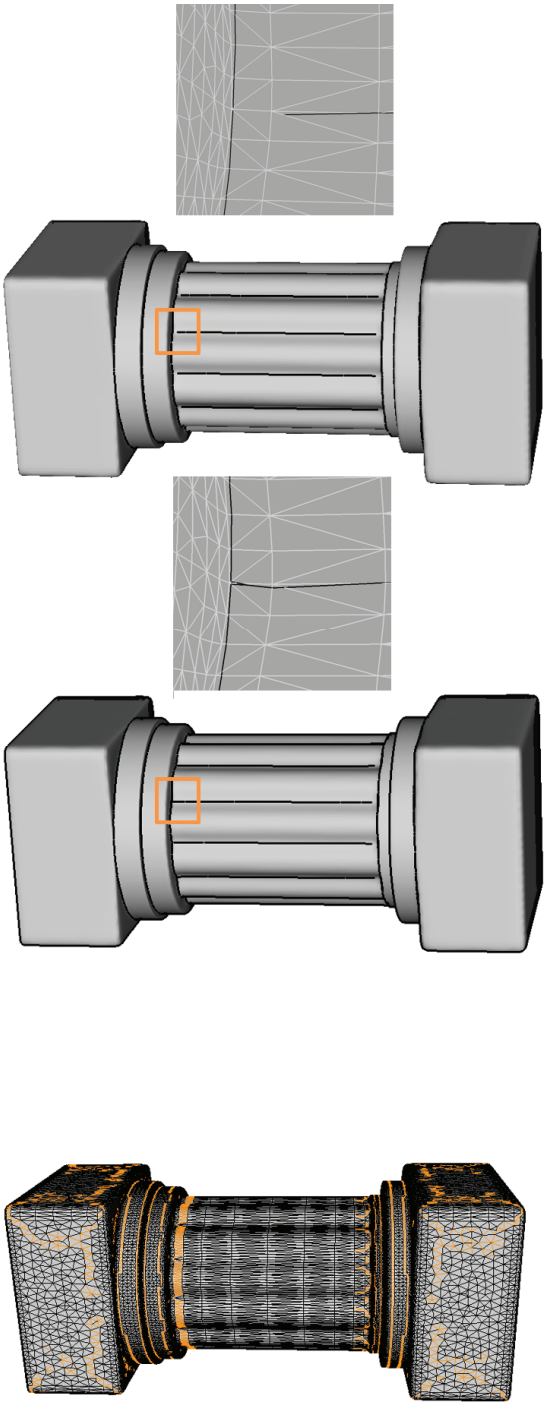

Fig. 3.19 Valley lines extracted from the Roman pedestal model (left) rendered with the principal curvature and its direction: (i) valley-like feature lines generated by interpolating the nearest maxima without the parallel test (middle); (ii) valley-like feature lines generated by interpolating the nearest maxima when edges violating the assumption are excluded (right). Mesh edges violating the assumption (with a difference in critical directions larger than $\frac{\pi}{3}$) are highlighted in orange (left).

Ground truth is usually absent in the context of feature-line extraction, so the concepts of average length and average fluctuation are our major evaluation criteria for feature lines. For the following simple analytical surfaces (see Fig. 3.20, Fig. 3.21, and Fig. 3.22), however, the ground truth of feature lines is explicitly computable.

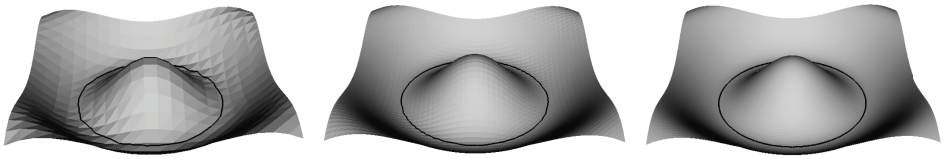

Fig. 3.20 A rotated cosine wave model of different resolutions rendered with valley lines: low (left), medium (middle), and high resolution (right). Short random lines are filtered by length. Both the total and average length of the ground truth are 19.739 ($2\pi^2$).

Both Table 3.4 and Table 3.6 demonstrate that feature lines with smaller mean squared error, which are better approximations of ground truth, generally have smaller average fluctuation and longer average length.

Table 3.4 Comparison of extracted feature lines in Fig. 3.20

	Low Res	Medium Res	High Res
MSE	0.00688	0.000230	4.565×10^{-5}
Number of lines	1	1	1
Average length	19.581	19.719	19.713
Avg fluctuation	0.325	0.131	0.0580

When short random lines are not removed completely (see Fig. 3.21), as the number of lines increases, their average length decreases considerably. The average length of lines proves to be a meaningful indicator of line fragmentation.

In other words, the quality of feature lines can be improved by removing shorter lines or lines with more fluctuation.

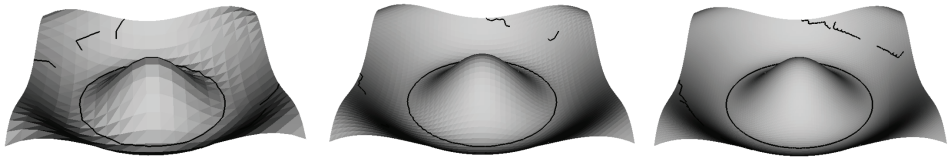

Fig. 3.21 A rotated cosine wave model of different resolutions rendered with valley lines: low (left), medium (middle), and high resolution (right). Short random lines are not filtered completely.

Table 3.5 Comparison of extracted feature lines in Fig. 3.21

	Low Res	Medium Res	High Res
MSE	1.00991	1.114	1.477
Number of lines	7	7	7
Average length	4.0139	4.267	4.635
Avg fluctuation	0.354	0.318	0.302

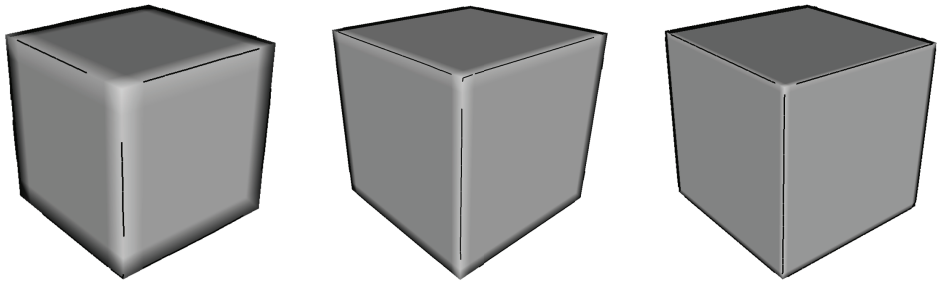

Fig. 3.22 A cube model of different resolutions rendered with ridge lines: low (left), medium (middle), and high resolution (right). The edge length of the cube is 2.

Table 3.6 Comparison of extracted feature lines in Fig. 3.22

	Low Res	Medium Res	High Res
MSE	3.755×10^{-5}	5.401×10^{-6}	9.002×10^{-7}
Number of lines	15	17	17
Total length	15.563	17.531	18.765
Average length	1.0375	1.0312	1.103
Avg fluctuation	0.135	0.0559	0.0253

Chapter 4

Feature-region classification & separation

This chapter is part of an unpublished paper, and thus is not included in this abridged dissertation.

Chapter 5

Overall comparison & Discussion

This chapter is part of an unpublished paper, and thus is not included in this abridged dissertation.

Chapter 6

Conclusion and future work

This chapter is part of an unpublished paper, and thus is not included in this abridged dissertation.

References

[1] P. Alliez, D. Cohen-Steiner, O. Devillers, B. Lévy, and M. Desbrun. Anisotropic polygonal remeshing. *ACM Trans. Graph.*, 22(3):485–493, 2003.

[2] E. Altantsetseg, Y. Muraki, K. Matsuyama, and K. Konno. Feature line extraction from unorganized noisy point clouds using truncated fourier series. *The Visual Computer*, 29:1432–2315, 2013. doi: 10.1007/s00371-013-0800-x.

[3] A. Bartesaghi and G. Sapiro. A system for the generation of curves on 3d brain images. *Human Brain Mapping*, 14(1):1–15, 2001.

[4] L. Bavoil, M. Sainz, and R. Dimitrov. Image-space horizon-based ambient occlusion. 2008.

[5] R. Bredow. Renderman in production. *SIGGRAPH Course notes*, 16, 2002.

[6] F. Buonamici, M. Carfagni, R. Furferi, L. Governi, A. Lapini, and Y Volpe. Reverse engineering of mechanical parts: A template-based approach. *J. Computational Design and Engineering*, 5:145–159, 2017.

[7] E. Catmull and J. Clark. Recursively generated b-spline surfaces on arbitrary topological meshes. *Computer-Aided Design*, 10:350–355, 1978.

[8] L. Charles. Smooth subdivision surfaces based on triangles. Master's thesis, Jan 1987.

[9] R. Cipolla and P. Giblin. *Visual Motion of Curves and Surfaces*. Cambridge University Press, New York, NY, USA, 2000. ISBN 0-521-63251-X.

[10] K. Crane, F. De Goes, M. Desbrun, and P. Schröder. Digital geometry processing with discrete exterior calculus. In *ACM SIGGRAPH 2013 courses*, SIGGRAPH '13, New York, NY, USA, 2013. ACM.

[11] D. DeCarlo, A. Finkelstein, S. Rusinkiewicz, and A. Santella. Suggestive contours for conveying shape. *ACM Trans. Graph.*, 22(3):848–855, 2003.

[12] J. Digne and JM. Morel. Numerical analysis of differential operators on raw point clouds. *Numerische Mathematik*, 127(2):255–289, 2014.

[13] D. Doo and M. Sabin. Behaviour of recursive division surfaces near extraordinary points. *Computer-Aided Design*, 10:356−360, 1978.

[14] A. Finkelstein, S. Rusinkiewicz, M. Burns, J. Klawe, D. DeCarlo, and A. Santella. Suggestive contours. https://gfx.cs.princeton.edu/proj/sugcon/, 2003.

[15] Q. Gao and Y. Yamaguchi. Detection of critical direction for feature line extraction on meshes based on local integral invariants. In L. Cocchiarella, editor, *ICGG 2018 - Proceedings of the 18th International Conference on Geometry and Graphics*, pages 252–264, Cham, 2019. Springer International Publishing.

[16] Q. Gao and Y. Yamaguchi. Extraction of coherent and smooth feature lines from meshes with fine details. *Computers & Graphics*, 82:222 – 231, 2019. doi: https://doi.org/10.1016/j.cag.2019.05.020.

[17] T. Judd, F. Durand, and E. Adelson. Apparent ridges for line drawing. *ACM Trans. Graph.*, 26(3), 2007.

[18] S. Kobayashi. *Differential geometry of curves and surfaces*. Shokabo, Tokyo, Japan, 1995.

[19] J. Koenderink. What does the occluding contour tell us about solid shape? *Perception*, 13(3):321–330, 1984.

[20] M. Kolomenkin, I. Shimshoni, and A. Tal. Demarcating curves for shape illustration. *ACM Trans. Graph.*, 27(5):157:1–157:9, 2008.

[21] S. Laine and T. Karras. Two methods for fast ray-cast ambient occlusion. In *Proceedings of the 21st Eurographics Conference on Rendering*, EGSR'10, pages 1325–1333, 2010.

[22] H. Landis. Production-ready global illumination. *Siggraph course notes*, 16, 2002.

[23] S. Manay, BW. Hong, J. Yezzi, A., and S. Soatto. Integral invariant signatures. In Tomás Pajdla and Jiří Matas, editors, *Computer Vision - ECCV 2004*, pages 87–99, Berlin, Heidelberg, 2004. Springer Berlin Heidelberg.

[24] Nelson L. Max. Weights for computing vertex normals from facet normals. *J. Graphics, GPU, & Game Tools*, 4:1–6, 1999.

[25] N.L. Max. Horizon mapping: shadows for bump-mapped surfaces. *The Visual Computer*, 4(2):109–117, 1988.

[26] Y. Ohtake, A. Belyaev, and HP. Seidel. Ridge-valley lines on meshes via implicit surface fitting. *ACM Trans. Graph.*, 23(3):609–612, 2004.

[27] H. Pottmann, W. Johannes, YL. Yang, YK. Lai, and SM. Hu. Principal curvatures from the integral invariant viewpoint. *Computer Aided Geometric Design*, 24(8):428–442, 2007.

[28] H. Pottmann, W. Johannes, QX. Huang, and YL. Yang. Integral invariants for robust geometry processing. *Computer Aided Geometric Design*, 26(1):37–60, 2009.

[29] S. Rusinkiewicz. Estimating curvatures and their derivatives on triangle meshes. In *Proceedings. 2nd International Symposium on 3D Data Processing, Visualization and Transmission, 2004. 3DPVT 2004.*, pages 486–493, 2004.

[30] G. Stylianou and G. Farin. Crest lines for surface segmentation and flattening. *IEEE Transactions on Visualization and Computer Graphics*, 10:536–544, 2004.

[31] G. Toussaint. Solving geometric problems with the rotating calipers. In *In Proceedings of IEEE MELECON'83*, volume 83, 1983.

[32] R. Vergne, D. Vanderhaeghe, J. Chen, P. Barla, X. Granier, and C. Schlick. Implicit brushes for stylized line-based rendering. *Computer Graphics Forum*, 30(2):513–522, 2011. doi: 10.1111/j.1467-8659.2011.01892.x.

[33] J. Wang, D. Gu, Z. Gao, Z. Yu, C. Tan, and L. Zhou. Feature-based solid model reconstruction. volume 13, 2013.

[34] A. Willis and B. Zhou. Ridge walking for 3d surface segmentation, 2019.

[35] YL. Yang, YK. Lai, SM. Hu, and H. Pottmann. Robust principal curvatures on multiple scales. In *Proceedings of the Fourth Eurographics Symposium on Geometry Processing*, SGP '06, pages 223–226, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association.

[36] Y. Yasuda and Y. Yamaguchi. Surrounded lines: robust line drawing based on quasi-local occlusion (in japanese). In *Proceedings of the 2010 Symposium on Visual Computing/Graphics and CAD*. The Institute of Image Electronics Engineers of Japan/IPSJ Special Interest Groups on Computer Graphics and Visual Informatics, 2010.

[37] S. Yoshizawa, A. Belyaev, and HP. Seidel. Fast and robust detection of crest lines on meshes. In *Proceedings of the 2005 ACM Symposium on Solid and Physical Modeling*, pages 227–232, New York, NY, USA, 2005. ACM.

[38] Y. Zhang, G. Geng, X. Wei, S. Zhang, and S. Li. A statistical approach for extraction of feature lines from point clouds. *Computers & Graphics*, 56:31–45, 2016.

[39] S. Zhukov, A. Iones, and G. Kronin. An ambient light illumination model. In *Rendering Techniques '98*, pages 45–55, Vienna, 1998. Springer Vienna.

Appendix A

Differential Geometry

As differential properties, including curvatures, are important indicators of surface geometry, we provide a brief overview of their definitions on 3D surfaces—either smooth surfaces or polygonal meshes—to facilitate understanding their implications in feature-line extraction.

Generally, the term "differential geometry" refers to the study of geometric problems using the techniques of differential calculus, integral calculus, linear algebra, etc. This section introduces the basics of differential geometry by referring mainly to [18], [9] and [10].

A.1 Curvature of planar curves

For a smooth planar curve γ among a myriad of circles through any point $\mathbf{p} \in \gamma$, there is a circle that best approximates it—that is, that shares the same tangent direction with γ at $\mathbf{p}$. This is known as the **osculating circle**, and its radius R is called the **radius of curvature**.

Intuitively, the reciprocal of R, $1/R$, indicates the curvedness of the curve γ, that is, how much it deviates from a straight line, so it is called the "curvature" of curve γ: the more the curve bends, the bigger the curvature.

Definition A.1 (**Osculating circle, radius of curvature, curvature**)**.** For an arbitrary point **p** on a planar curve γ, the circle through **p** that best approximates γ is called the **osculating circle**, its radius R the **radius of curvature**. The reciprocal of R is called **curvature**.

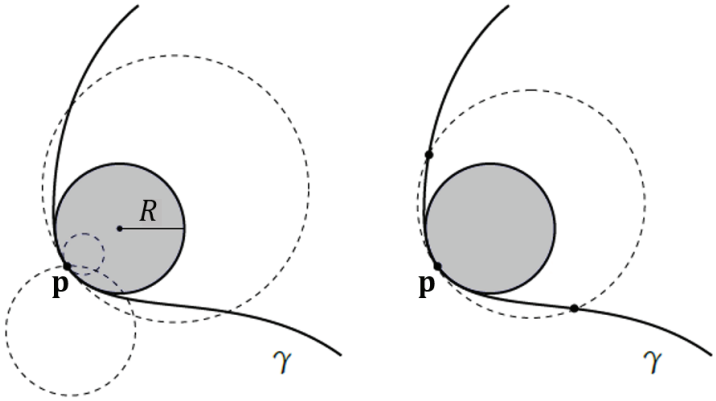

Fig. A.1 The osculating circle (filled in gray) of the curve γ approximates γ the best at **p**. Reproduced from [10].

A.2 The Geometry of Surfaces

Consider a parameter plane with coordinates (u, v) and a connected region D. A surface **S** can thus be parameterized in the following form:

$$\mathbf{S}(u,\ v) = (x(u,\ v),\ y(u,\ v), z(u,\ v)) \tag{A.1}$$

where x, y, and z are functions with infinite continuous derivatives. In other words, the geometry of the surface **S** can be described through a map f from D in the Euclidean plane $\mathbb{R}^2$ to $f(D)$, which is a subset of $\mathbb{R}^3$.

Note that for the Jacobian matrix of $\mathbf{S}$,

$$\begin{pmatrix} x_u & x_v \\ y_u & y_v \\ z_u & z_v \end{pmatrix}$$

or equivalently,

$$\begin{pmatrix} \frac{\partial x}{\partial u} & \frac{\partial x}{\partial v} \\ \frac{\partial y}{\partial u} & \frac{\partial y}{\partial v} \\ \frac{\partial z}{\partial u} & \frac{\partial z}{\partial v} \end{pmatrix}$$

Its two columns $\mathbf{S}_u$ and $\mathbf{S}_v$ or, interchangeably, $\frac{\partial S}{\partial u}$ and $\frac{\partial S}{\partial v}$ are both tangent vectors of the surface, and they are not necessarily perpendicular. That the plane spanned by $\mathbf{S}_u$ and $\mathbf{S}_v$ is a tangent plane of $\mathbf{S}$ is self-evident; that is, $\mathbf{S}_u$ and $\mathbf{S}_v$ form a vector basis for the tangent plane. Consequently, for every point (u, v) in the parameter plane, there is a corresponding tangent plane at $\mathbf{S}(u,\ v)$ of surface $\mathbf{S}$. Both the tangent plane and the surface share the same normal vector $\mathbf{n}$, which is determined by the cross product of $\mathbf{S}_u$ and $\mathbf{S}_v$ at the point $\mathbf{S}(u,\ v)$ (see Fig. A.2). As a result, the normal only depends on the first-order derivatives of the surface parameterization, and the tangent plane is often called the "first-order approximation" of the underlying surface around the point.

For an arbitrary curve γ on surface $\mathbf{S}$, there is a corresponding planar curve

$$u = u(t), v = v(t) \tag{A.2}$$

on the parameter plane with t as the curve parameter, which is not necessarily the arc length. The curve γ could be expressed in the form

$$\gamma(t) = \gamma(u(t),\ v(t)) \tag{A.3}$$

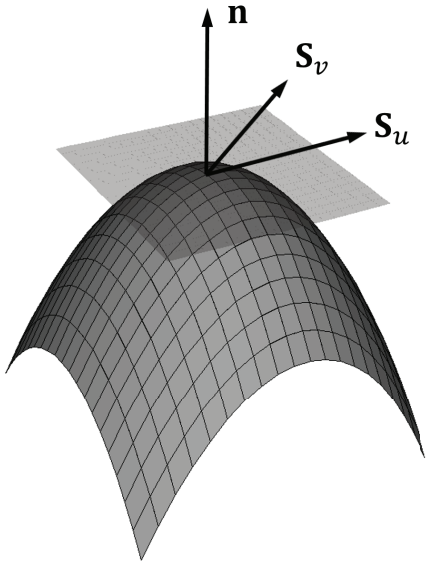

Fig. A.2 A tangent plane of the surface $\mathbf{S}$ spanned by tangent vectors $\mathbf{S}_u$ and $\mathbf{S}_v$, with surface normal $\mathbf{n}$. Reproduced from [9]

Consider the tangent vectors along γ. As $\gamma \subset \mathbf{S}$, they could be expressed based on the chain rule:

$$\gamma'(t) = \frac{d\gamma}{dt} = \frac{\partial\gamma}{\partial u}\frac{du}{dt} + \frac{\partial\gamma}{\partial v}\frac{dv}{dt} = \mathbf{S}_u u' + \mathbf{S}_v v' \tag{A.4}$$

To find the length of such a tangent vector of γ, we introduce the inner product.

$$\begin{aligned}
\langle \gamma'(t), \gamma(t) \rangle &= \|\gamma'(t)\|^2 \\
&= \begin{bmatrix} u' & v' \end{bmatrix} \begin{bmatrix} \mathbf{S}_u \cdot \mathbf{S}_u & \mathbf{S}_u \cdot \mathbf{S}_v \\ \mathbf{S}_v \cdot \mathbf{S}_u & \mathbf{S}_v \cdot \mathbf{S}_\mathbf{v} \end{bmatrix} \begin{bmatrix} u' \\ v' \end{bmatrix} \\
&= \begin{bmatrix} u' & v' \end{bmatrix} \begin{bmatrix} E & F \\ F & G \end{bmatrix} \begin{bmatrix} u' \\ v' \end{bmatrix} \\
&= Eu'^2 + 2Fu'v' + Gv'^2
\end{aligned}$$

in which

$$E = \mathbf{S}_u \cdot \mathbf{S}_u, F = \mathbf{S}_u \cdot \mathbf{S}_v, G = \mathbf{S}_v \cdot \mathbf{S}_v. \tag{A.5}$$

The length of the curve segment on γ when t moves from α to β is thus

$$\begin{aligned}\int_\alpha^\beta \sqrt{Eu'^2 + 2Fu'v' + Gv'^2}dt &= \int_\alpha^\beta \sqrt{E\Big(\frac{du}{dt}\Big)^2 + 2F\frac{du}{dt}\frac{dv}{dt} + G\Big(\frac{dv}{dt}\Big)^2}dt \\ &= \int_\alpha^\beta \sqrt{Edu^2 + 2Fududv + Gdv^2}dt\end{aligned}$$

Definition A.2 (**First fundamental form**)**.** The quadratic form

$$I = Eu'^2 + 2Fu'v' + Gv'^2 \quad (or \quad Edu^2 + 2Fdudv + Gdv^2) \tag{A.6}$$

is called the **first fundamental form** of the surface.

Furthermore, as mentioned previously, any tangent vector lying in the tangent plane of $\mathbf{S}$ at point $\mathbf{p}$ could be expressed as a linear combination of $\mathbf{S_u}$ and $\mathbf{S_v}$ at that point, such as $\xi\mathbf{S_u} + \eta\mathbf{S_v}$, where (ξ, η) is the coordinate of the vector on the tangent plane.

Suppose $\mathbf{a}_1$ and $\mathbf{a}_2$ are one pair of tangent vectors at $\mathbf{p}$, that is, $\mathbf{a}_1 = \xi_1\mathbf{S_u} + \eta_1\mathbf{S_v}$, $\mathbf{a}_2 = \xi_2\mathbf{S_u} + \eta_2\mathbf{S_v}$. Consider the inner product of $\mathbf{a}_1$ and $\mathbf{a}_2$:

$$\begin{aligned}\langle \mathbf{a}_1, \mathbf{a}_2 \rangle &= \begin{bmatrix}\xi_1 & \eta_1\end{bmatrix}\begin{bmatrix}\mathbf{S}_u \cdot \mathbf{S}_u & \mathbf{S}_u \cdot \mathbf{S}_v \\ \mathbf{S}_v \cdot \mathbf{S}_u & \mathbf{S}_v \cdot \mathbf{S}_v\end{bmatrix}\begin{bmatrix}\xi_2 \\ \eta_2\end{bmatrix} \\ &= \begin{bmatrix}\xi_1 & \eta_1\end{bmatrix}\begin{bmatrix}E & F \\ F & G\end{bmatrix}\begin{bmatrix}\xi_2 \\ \eta_2\end{bmatrix}\end{aligned}$$

For clarification, the 2 ×2 matrix

$$\begin{bmatrix}E & F \\ F & G\end{bmatrix} \tag{A.7}$$

is called the **matrix of the first fundamental form**. It is also expressed with **I** under some occasions.

Recall that the normal of a surface, that is, the normal of the tangent plane of the surface, only depends on the first-order derivatives of the surface parameterization. For second-order derivatives of the surface, once again, we return to the space curve $\gamma(u(t),\ v(t))$ on surface **S** with the aforementioned parameterization and further examine the behavior of the tangent plane of **S** (the normal **n** of **S**) when the point of tangency moves around a tiny surface patch.

Definition A.3 (**Second fundamental form**)**.** The quadratic form

$$II = Lu'^2 + 2Mu'v' + Nv'^2 \quad (or \quad Ldu^2 + 2Mdudv + Ndv^2) \tag{A.8}$$

is called the **second fundamental form** of the surface, in which

$$L = -\mathbf{S}_u \cdot \mathbf{n}_u, M = -\mathbf{S}_u \cdot \mathbf{n}_v = -\mathbf{S}_v \cdot \mathbf{n}_u, N = -\mathbf{S}_v \cdot \mathbf{n}_v. \tag{A.9}$$

Similarly, the 2 ×2 matrix

$$\begin{bmatrix} L & M \\ M & N \end{bmatrix} \tag{A.10}$$

is called the **matrix of the second fundamental form** or the **second fundamental tensor**. It is also sometimes expressed with **II**.

A.2.1 Curvature of smooth surfaces

Let us consider an arbitrary point **p** on a smooth, closed surface. Unlike the case of a curve, understanding the curvature of a surface can be much more complicated, as it can bend sharply along one direction while being completely flat along another, such as a cylinder.

In fact, the curvature of a surface is typically defined in terms of the curves contained in it. Consider a plane containing both **a**, an arbitrary tangent vector of the surface at **p**, and **n**, the normal of the surface at **p**. This plane intersects the surface, and the curvature κ of this

intersection curve (a **normal curve**) is called the "normal curvature" of the surface in the direction **a** (see Fig. A.3).

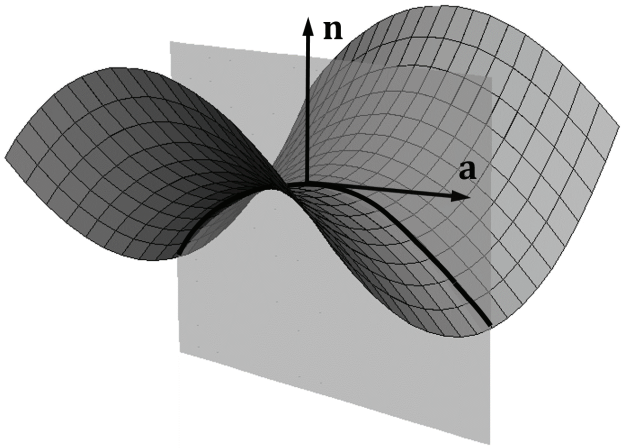

Fig. A.3 Intersection of a surface with a normal plane, the tangent to the intersection curve being **a**. Reproduced from [9]

Generally, multiplying the second fundamental tensor **II** by any vector in the tangent plane gives the derivative of the surface normal in that direction,

$$\mathbf{IIa} = D_{\mathbf{a}}\mathbf{n}, \tag{A.11}$$

which is a vector in the tangent plane.

As the normal curvature at any given point **p** could be regarded a function of the tangent direction at that point, we are interested in when this function gains its extrema, that is, when the surface bends the most. Suppose that unit vectors $\mathbf{X_1}$ and $\mathbf{X_2}$ indicate the directions along which we find, respectively, the maximum and minimum normal curvatures κ_1 and κ_2. These directions and their corresponding normal curvatures are thus called the "principal directions (of curvature)" and "principal curvatures" of the surface at **p**, respectively. Noticeably, the

normal curvature fluctuates smoothly with tangent direction on a smooth surface, ranging from κ_1 to κ_2.

The **mean curvature** and **Gaussian curvature** are closely related quantities (with especially meaningful interpretation on discrete surfaces). While the former is simply the arithmetic mean of two principal curvatures, the latter is the square of their geometric mean.

Definition A.4 (**Mean curvature, Gaussian curvature**)**.** For an arbitrary point $\mathbf{p}$ on a smooth surface, let κ_1 and κ_2 be principal curvatures of the surface at $\mathbf{p}$. Given

$$H = \frac{\kappa_1 + \kappa_2}{2}, \quad K = \kappa_1 \kappa_2 \tag{A.12}$$

, H and K are the **mean curvature** and **Gaussian curvature** of the surface at $\mathbf{p}$, respectively.

The Gaussian curvature measures the spherical spread of the surface normal, while the mean curvature can be considered the average of dihedral angles. Therefore, they are also called **intrinsic** and **extrinsic** curvatures, respectively.

A.2.2 Curvature of discrete surfaces

In a discrete differential geometry setting, reliably estimating per-vertex curvature [29] requires geometric information of at least 2-ring neighborhoods of mesh vertices.

First, per-vertex normals are estimated as the weighted averages of the per-facet normals of neighboring facets. The contribution of the facet $\triangle \mathbf{p}_0 \mathbf{p}_1 \mathbf{p}_2$ to vertex $\mathbf{p}_i$ (i = 0, 1, 2) (see Fig. A.4) is specified by weight,

$$w_{\mathbf{p}_i} = \frac{\sin \theta_i}{\|\mathbf{e}_{i+1}\| \|\mathbf{e}_{i+2}\|}, \tag{A.13}$$

with all indices taken mod 3. A coordinate system $(\mathbf{u}_\mathbf{p}, \mathbf{v}_\mathbf{p})$ in the tangent plane of each vertex $\mathbf{p}$ is then constructed.

Given Equation A.11, the directional derivatives of surface normals along the directions of mesh edges are turned into a set of linear constraints on the elements of the per-facet curvature tensor **II** (see Fig. A.4).

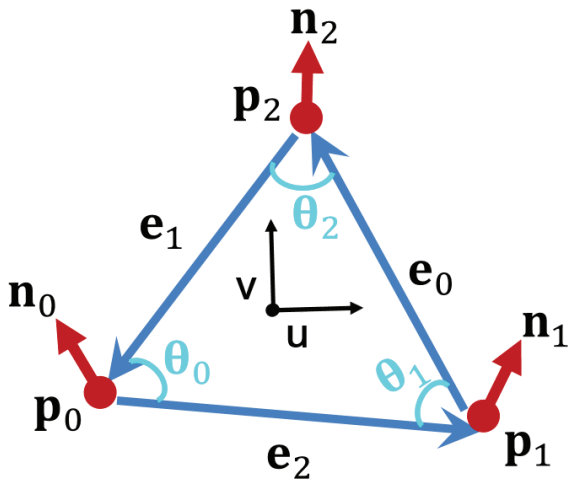

Fig. A.4 Directional derivatives of surface normals along directions of mesh edges turned into a set of linear constraints on the elements of **II**. Reproduced from [29]

$$\mathbf{II}\begin{bmatrix} \mathbf{e}_0 \cdot \mathbf{u} \\ \mathbf{e}_0 \cdot \mathbf{v} \end{bmatrix} = \begin{bmatrix} (\mathbf{n}_2 - \mathbf{n}_1) \cdot \mathbf{u} \\ (\mathbf{n}_2 - \mathbf{n}_1) \cdot \mathbf{v} \end{bmatrix} \tag{A.14}$$

$$\mathbf{II}\begin{bmatrix} \mathbf{e}_1 \cdot \mathbf{u} \\ \mathbf{e}_1 \cdot \mathbf{v} \end{bmatrix} = \begin{bmatrix} (\mathbf{n}_0 - \mathbf{n}_2) \cdot \mathbf{u} \\ (\mathbf{n}_0 - \mathbf{n}_2) \cdot \mathbf{v} \end{bmatrix} \tag{A.15}$$

$$\mathbf{II}\begin{bmatrix} \mathbf{e}_2 \cdot \mathbf{u} \\ \mathbf{e}_2 \cdot \mathbf{v} \end{bmatrix} = \begin{bmatrix} (\mathbf{n}_1 - \mathbf{n}_0) \cdot \mathbf{u} \\ (\mathbf{n}_1 - \mathbf{n}_0) \cdot \mathbf{v} \end{bmatrix}. \tag{A.16}$$

Here, $(\mathbf{u}, \mathbf{v})$ makes an orthonormal coordinate system on the plane determined by facet $\triangle \mathbf{p}_0\mathbf{p}_1\mathbf{p}_2$. The elements of **II**—L, M, and N—can then be solved using least squares.

For each vertex $\mathbf{p}$ (with a coordinate system of $(\mathbf{u_p}, \mathbf{v_p})$ for the tangent plane at $\mathbf{p}$) of a facet, assuming the facet and vertex normals are equal, re-express per-facet curvature tensor $\mathbf{II}$ in terms of $(\mathbf{u_p}, \mathbf{v_p})$ as

$$\begin{bmatrix} L_\mathbf{p} & M_\mathbf{p} \\ M_\mathbf{p} & N_\mathbf{p} \end{bmatrix} \tag{A.17}$$

, in which

$$L_\mathbf{p} = \mathbf{u_p}^T \mathbf{II} \mathbf{u_p}, M_\mathbf{p} = \mathbf{u_p}^T \mathbf{II} \mathbf{v_p}, N_\mathbf{p} = \mathbf{v_p}^T \mathbf{II} \mathbf{v_p}. \tag{A.18}$$

When the facet and vertex normals are unequal, first, one of the coordinate systems is rotated to be coplanar with the other, around the cross product of their normals.

Per-vertex curvature tensors are then defined as the weighted averages of the re-expressed per-facet curvature tensors (Eq. A.17) of neighboring facets with "Voronoi area" weighting, that is, the portion of the area of a neighboring facet that lies closest to a vertex.

Appendix B

Defining Critical Directions from Integral Invariants

B.1 A Framework for Defining Critical Directions from the viewpoint of Integral Invariants

Consider a local neighborhood $N^r(\mathbf{p})$ of size r for a specific point $\mathbf{p}$ on a surface, that is, a Euclidean disk centered on $\mathbf{p}$, which can be constructed by the intersection of the surface and a kernel ball or sphere with a radius r. In our previous work [15], we explored the possibility of leveraging $N^r(\mathbf{p})'$, the mapping of $N^r(\mathbf{p})$, for definition of critical directions. It aims to suggest a solution to the problem according to the nature of $N^r(\mathbf{p})'$. Considering the mapping $N^r(\mathbf{p})'$ of $N^r(\mathbf{p})$ onto other geometries (e.g. a plane), we need a "normal" vector of $N^r(\mathbf{p})$ to perform orthogonal projection. Intuitively, this "normal" can be defined as the normal vector of the plane, onto which the orthogonal projection of the disk or its boundary, that is, the intersection curve, attains its maximum area or perimeter, respectively. While Table B.1 names several types of spherical curves that can be derived from $N^r(\mathbf{p})$, Table B.2 summarizes the approaches that can be taken to determine critical directions over projections of these spherical curves.

Table B.1 A framework for defining critical directions over local neighborhoods

$N^r(\mathbf{p})'$ derived from the disk	Geometric implication of $N^r(\mathbf{p})'$	Approaches to obtain critical directions
(a) The surface patch (of the mesh) within the kernel ball	——	• PCA of the surface patch • PCA of the volume bounded by the surface patch and the kernel ball
(b) Spherical curves	(c) The intersection curve of the kernel sphere with the mesh (d) The horizon viewed from **p** (e) The boundary of the Gauss map of (a)	• PCA of the spherical curves • PCA of the spherical area bounded by the spherical curve • Find the critical direction over the spherical convex hull (SCH) of (e) • Find the critical direction from the circular sequence of elevation angles (in sampled directions de-noted by azimuthal angles), which is defined: 1. Based on the "normal" derived from the spherical curve itself 2. Based on the "normal" defined otherwise

(c) The intersection curve refers to a sequence of arcs, each of which can be defined exactly as the intersection of a triangle of the mesh with the kernel sphere

(d) The "horizon" for a given point **p** on a surface over $N^r(\mathbf{p})$ can be regarded as the boundary of the visible spherical area over a kernel sphere of radius r when viewed from **p**.

(e) To define the critical direction from the Gauss map of a disk, we focus on the spherical area bounded by the boundary of the Gauss map. We can construct the spherical convex hull of the spherical area, which is a spherical polygon whose sides are segments of great circles.

Table B.2 Defining critical directions from $N^r(\mathbf{p})'$ derived from (b)

$N^r(\mathbf{p})'$ derived from (b)	Geometric implication of $N^r(\mathbf{p})'$	Approaches to obtain critical directions
Projection of (b) along the direction of the "normal" onto the tangent plane at **p**	A planar curve on the tangent plane at **p**	• PCA of the planar curve generated by projection • PCA of the area bounded by the planar curve • Find the critical direction over the convex hull of the planar polygon • Find the critical direction from the circular sequence of distances (from the planar curve to **p**)

Inspired by the "rotating caliper" method [31], which is originally designed for planar polygons, we can define "diameter" and "width" for a spherical convex hull in a similar manner. In this case, we look for a pair of great circles that can enclose the spherical convex hull. In

the meantime, based on the "center" of the spherical convex hull, that is, the aforementioned "normal" of the disk, we can easily decide the direction with the largest/smallest expansion or aspect ratio over the spherical convex hull of $G(N^r(\mathbf{p}))$.

B.2 Comparison of Integral Invariants

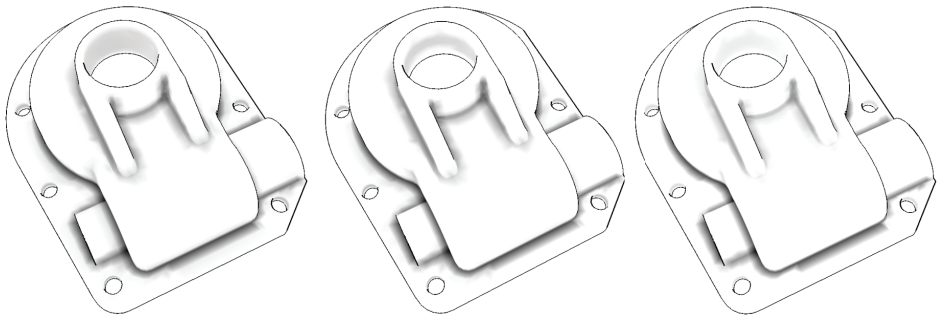

Fig. B.1 The casting model shaded with (from left to right): the occlusion (with occluding contours), the surface area descriptor, the volume descriptor (both the surface area descriptor and the volume descriptor are scaled to match the occlusion).

Three different integral invariants-the occlusion, the surface area descriptor, and the volume descriptor-are compared in Fig. B.1. Critical directions corresponding to the occlusion are almost perpendicular to the valley, hence generating the longest and smoothest line segment among the three over the same region (see Fig. B.2 left). Critical directions corresponding to the volume descriptor, however, fail to exhibit such desirable behavior (see Fig. B.2 right). This may be because the radius of the kernel ball, that is, the minimum local neighborhood size (see Section 2.2.2), is too small for it to achieve the accuracy required for feature-line extraction. Accuracy of the surface area descriptor in this sense lies somewhere between (see Fig. B.2 middle).

We can see that out of the three, the occlusion generates the most coherent and smoothest lines. It is also the easiest to compute and manipulate, as it can be estimated with sufficient accuracy with a much smaller neighborhood than the others.

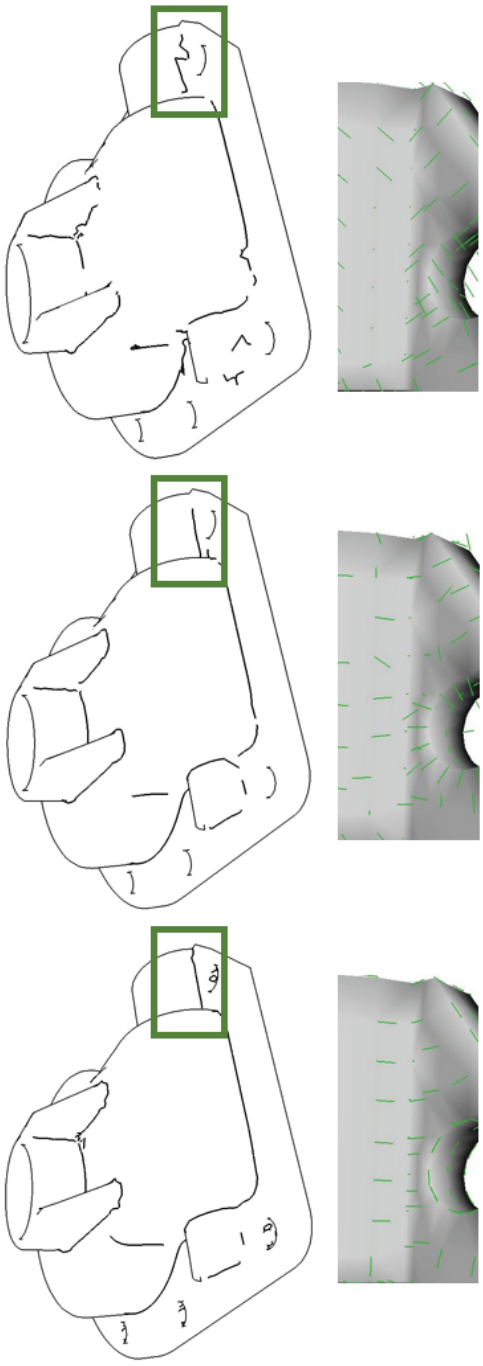

Fig. B.2 Zoom in of a part of the casting model rendered with valley-like feature lines and corresponding critical directions (from left to right): the occlusion and the direction of maximum occlusion (left), the surface area descriptor and the principal direction of curvature (middle), the volume descriptor and the principal direction of curvature (right).