An Active Contour Approach to Extract Feature Regions from Triangular Meshes

Kyungha Min[1] and Moon-Ryul Jung[2]
[1] Division of Digital Media, Sangmyung University,
Seoul, Korea
[e-mail: minkh@smu.ac.kr]
[2] Graduate School of Media, Sogang University,
Seoul, Korea
[e-mail: moon@sogang.ac.kr]
*Corresponding author: Kyungha Min

Received November 23, 2010; revised December 17, 2010; accepted January 11, 2011; published March 31, 2011

Abstract

We present a novel active contour-based two-pass approach to extract smooth feature regions from a triangular mesh. In the first pass, an active contour formulated in level-set surfaces is devised to extract feature regions with rough boundaries. In the second pass, the rough boundary curve is smoothed by minimizing internal energy, which is derived from its curvature. The separation of the extraction and smoothing process enables us to extract feature regions with smooth boundaries from a triangular mesh without user's initial model. Furthermore, smooth feature curves can also be obtained by skeletonizing the smooth feature regions. We tested our algorithm on facial models and proved its excellence.

Keywords: Feature extraction, active contour, level-set, triangular meshes, non-photorealistic rendering

[1]This research is supported by the Ministry of Culture, Sports and Tourism (MCST) and Korea Culture Content Agency (KOCCA) in the Culture Technology (CT) Research and Development Program 2010. This research is also supported by Basic Science Research Program through the Korea Research Foundation (KRF) funded by the Ministry of Education, Science and Technology at 2010 (2010-0006807). [2]This work was supported by the Sogang University Research Grant of 2010.

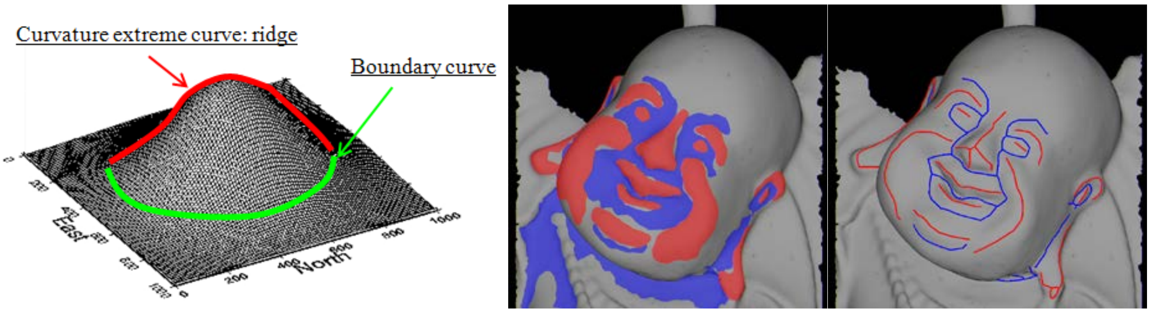
DOI: 10.3837/tiis.2011.03.007

1. Introduction

Features on triangular meshes provide important abstraction of shape, which can be used in many mesh processing tasks, including non-photorealistic rendering (NPR). Recently, there has been increasing research on developing new methods of extracting features from triangular meshes, which can be classified into techniques based on skeletonization [1][2][3][4][5][6], active contours [7][8][9][10][11], and fitting [12][13][14], together with miscellaneous schemes [15][16][17]. Most of these techniques produce curvature-extreme curves such as ridges or ravines. However, we strongly assert that regions of homogeneous shape are at least as important in describing the form of an object; and we introduce a technique that searches for the boundaries of such regions (see **Fig. 1-(a)**).

The extraction of features from objects was originally studied in computer vision and image processing. The active contour model [18] is a well-known feature extraction technique, but its results depend heavily on the initial contours created by users. Level-set surfaces [19] were subsequently combined with the active contour model [20][21][22], making this user input unnecessary.

We have developed a two-pass scheme, based on active contours, that can extract both smooth feature regions and smooth feature curves from triangular meshes (see **Fig. 1-(b)**). Like other recent authors, we dispense with an initial model by formulating the active contour model in terms of level-set surfaces defined on a triangular mesh. However, a drawback of previous methods which use level-set surfaces to initialize active contours is that the boundaries of the resulting regions are far from smooth [11][20][21]. We avoid this problem by separating the processes of extraction and smoothing. In the first pass, we extract raw feature regions from a triangular mesh using an active contour formulated from level-set surfaces. In the second pass, we smooth these raw feature regions. We can also go on to skeletonize the smooth feature regions to produce smooth feature curves; these can perform a similar role to curvature-extreme curves.



(a) A curvature-extreme curve compared with a boundary curve of a homogeneous region.

(b) Smooth feature regions and smooth feature curves extracted from the Buddha face model by our algorithm

Fig. 1. An overview of our result: Feature regions surrounded by smooth boundary curves as well as smooth curvature extreme curves are extracted.

The ability of our scheme to extract smooth feature regions as well as smooth curvature-extreme curves without the need for user-created initial models represents a significant advance over most existing schemes operating on triangular meshes, which only extract curvature-extreme curves.

The rest of this paper is organized as follows. In Section 2 we review related research, and in Section 3 we introduce our algorithm. In Section 4 we describe the first pass, in which we use active contours to extract raw feature regions, and in Section 5 we describe the second pass, in which active contours are used to smooth the raw feature regions. In Section 6 we develop a scheme for extracting feature curves from feature regions. In Section 7, we present an implementation of our algorithm and compare our results with those from related techniques. Finally we draw conclusions and give suggestions for future work in Section 8.

2. Related Work

Most methods for extracting features from triangular meshes are based on one of three techniques: skeletonization, active contours and fitting.

2.1 Skeletonization-based schemes

Rossl et al. [4] proposed a scheme to extract feature lines from triangulated surfaces using morphological operators such as opening and closing. They approximate curvature values at each vertex and threshold them to build a binary feature vector, and then use their morphological operators to remove noises and artifacts from the feature vectors at the vertices. Feature lines are then obtained by applying a skeletonization scheme to the feature vectors. Hubeli and Gross [2] developed a multiresolution feature extraction algorithm, which consists of two stages. In an initial classification stage, a feature weight is computed for each edge in the mesh using operators such as best-fit polynomials fitting, and finding angles between best-fit polynomials. In the subsequent detection stage, important edges are selected and used to construct a patch, which is skeletonized to form a feature curve. Gumhold et al. [1] extracted feature lines from point clouds using another two-stage approach. In the first stage, they assign penalty weights to the points and then prune the neighbor graph while minimizing the contribution of these weights using a minimum spanning graph algorithm. In the second stage, the feature lines and junctions are recovered from the resulting sub-graph. Watanabe and Belyaev [6] proposed a stable feature detection scheme for triangular meshes. They use a new algorithm to approximate the mean and Gaussian curvature of a mesh, which are then combined in a nonlinear way. Pauly et al. [3] put forward a multi-scale feature extraction scheme for point-sampled surfaces. These authors select feature candidates through principal component analysis, and then build initial feature curves by using a hysteresis threshold to compute a minimum-cost spanning tree. The initial feature curves are subsequently smoothed using active contours. Stylianou and Farin [5] extract crest lines from triangulated surfaces. They estimate curvatures at the vertices of the surface and identify the vertices with locally maximum curvatures. Then they apply a region-growing algorithm to these vertices to build crest regions, which are skeletonized to form crest lines.

2.2 Active contour-based schemes

Milroy et al. [10] introduced the first snake-based feature segmentation algorithm for 3D triangular meshes. Andrew [7] subsequently proposed a scheme in which a geodesic curve is constructed on a triangular mesh by connecting two user-specified vertices. The geometric snake, proposed by Lee and Lee [9], detects feature edges on a triangular mesh by projecting the region surrounding a snake into a 2D plane, in which a snake `slithers' from an initial configuration to the feature curve by energy minimization. Jung and Kim [8] also proposed a snake-based feature extraction scheme for triangular meshes. The common drawback of these schemes is that they require users to construct an initial snake model.

2.3 Fitting-based schemes

Ohtake et al. [13] proposed an algorithm for detecting ridges and valley lines that combines multi-level implicit surface fitting with finite-difference approximations. Yoshizawa et al. [14] suggested local fitting of polynomials to triangular meshes and then detecting crest lines. From estimates of curvatures and curvature derivatives derived from these polynomials. Kim and Kim [12] used a moving-least-squares approximation to estimate local differential information near a vertex by means of an approximation surface. In their framework, ridge and valley vertices are detected as zero-crossings, and then connected along the direction of principal curvature.

2.4 Other schemes

Sahner et al. [16] proposed a scheme in which extreme feature lines are extracted from surface meshes using discrete Morse theory. By prioritizing features from the surface meshes, their scheme can control the level of details in the feature lines. But, this scheme gives the user little control over the types of features that are found. Kim et al. [15] exploit the color as well as the geometry of a model. Their scheme finds both feature curves and feature regions, but of course it cannot be used when no colir is available. Zhihong et al. [17] suggested a scheme that extracts crest lines from triangular meshes using contextual information gathered from the neighborhood vertices.

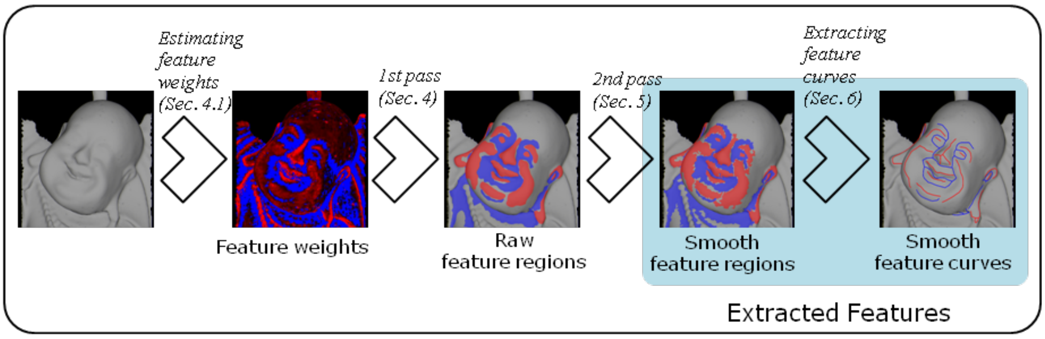


Fig. 2. An overview of our algorithm

3. Overview of the algorithm

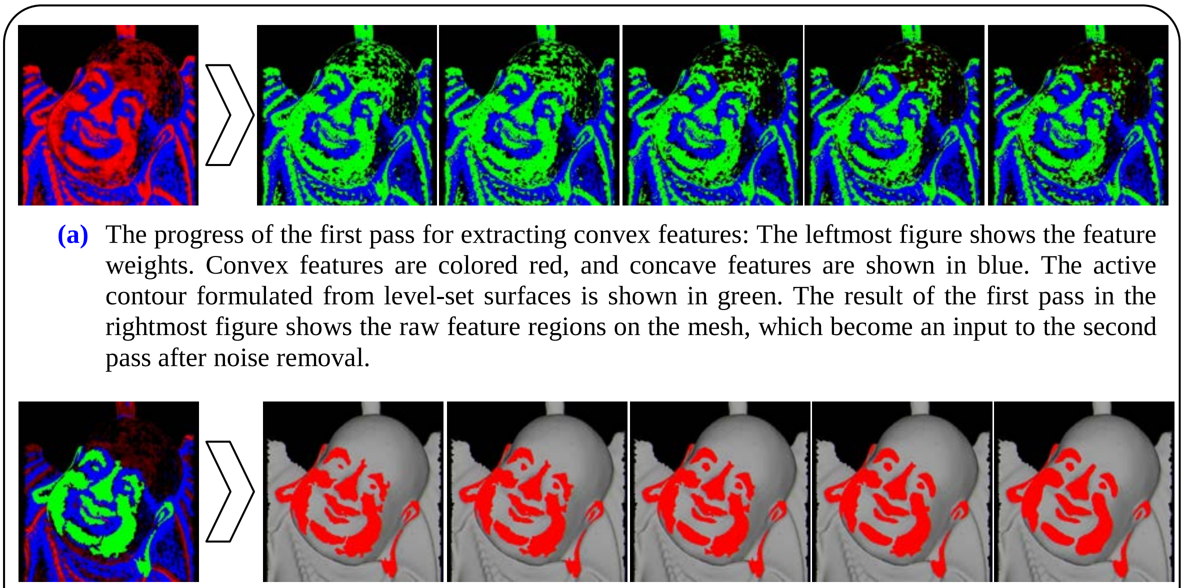
The input to our method summerized in **Fig. 2** is a 3D object represented as a triangular mesh. This is processed as follows (see also **Fig. 3**).

Step 1. Feature weights are estimated for each vertex on the mesh.

Step 2. In the first pass, we extract raw feature regions using an active contour model formulated in terms of level-set surfaces [21] which operates on a triangular mesh [11]. Since smoothing is performed in the second pass, we include no smoothing terms at this stage [11]. However, we add a threshold term to provide an element of user control. Noise, in the form of very small regions and holes in regions, is eliminated, and we call the results of the first pass raw feature regions.

Step 3. The boundaries of the raw feature regions are expressed as piecewise-linear curves. In the second pass, an energy minimizing active contour model is applied to these curves to create smooth feature regions. This smoothing is applied to each raw feature regions individually.

Step 4. The smooth feature regions are skeletonized and the resulting curves are smoothed. These feature curves represent either ridges or ravines. Curves obtained in this way are smoother than those derived directly from raw feature regions.



(a) The progress of the first pass for extracting convex features: The leftmost figure shows the feature weights. Convex features are colored red, and concave features are shown in blue. The active contour formulated from level-set surfaces is shown in green. The result of the first pass in the rightmost figure shows the raw feature regions on the mesh, which become an input to the second pass after noise removal.

(b) The progress of the second pass for extracting features. The raw feature regions after noise removal, shown in the leftmost figure become an input to the second pass.

Fig. 3. Stages in the two-pass approach to extracting features. The process illustrated is working on convex features, which as colored red.

4. First pass: extracting raw feature regions

In the first pass, raw feature regions are extracted by an active contour model that minimizes its external energy. The energy functional of the active contour is based on classifying the vertices on the mesh as inside or outside the active contour [11][21]. It is formulated from level-set surfaces and solved numerically on the triangular mesh.

4.1 Estimating feature weights

We now describe how we specify the kind of features which will be extracted. We regard vertices at which the mesh has a high curvature as candidate for feature region. The likelihood that each vertex belongs to a feature region is expressed as a weight, which is determined from the estimate of the local curvature at the vertex. Several researchers have proposed schemes for approximating curvatures on triangular meshes [23][24][25][26]. We propose a simple method in which the local curvature at a vertex is estimated from the hinge angles between the edges incident to t hat vertex. The hinge angle of an edge is the angle between the normals to the two faces adjacent to that edge. The hinge angle is positive when the edge is convex, and negative when it is concave. We assume that the maximum and the minimum hinge angle at a vertex respectively approximate the maximum and the minimum principal curvatures.

The maximum hinge angle at a vertex is called its *convex* feature weight, and expresses the likelihood that the vertex belongs to convex regions. Similarly, the minimum hinge angle at a vertex is called the *concave* feature weight, and expresses the likelihood that the vertex belongs to concave regions.

We denote the feature weight of a vertex v as u (v) and the hinge angle of an incident edge e is denoted by d ($\underline{e}$). If the set of edges incident to v is $\{e^v{}_1, ..., e^v{}_n\}$, then the feature weights are defined as follows, depending on the kind of feature that we are trying detect:

$$u(\mathrm{v}) = \begin{cases} \max\limits_{1\le i\le n}\{d(e_i^{\mathrm{v}})\}, & \text{for extracting convex features} \\ \min\limits_{1\le i\le n}\{d(e_i^{\mathrm{v}})\}, & \text{for extracting concave features.} \end{cases} \quad \text{.......................}(1)$$

4.2 Energy functional for the active contour model

We formulate the extraction of feature curves from meshes as a minimum partition problem [21]. Given an image, the minimum partition problem is to find a boundary curve C such that the region inside C and the region outside C are both as homogeneous as possible. This problem can be reformulated as an energy minimization. We can contrive a similar formulation over a 3D mesh once we have assigned feature weights to each vertex. We consider a feature to be a homogeneous region with high feature values, and not the boundary curve. Nevertheless, a feature may still turn out to be a curve, if the feature weights of that curve's vertices are very different from those of their neighbors.

In the case of an image, the partitioning approach has the advantage over a snake that it can be used when the gradients of image intensity are difficult to estimate. Even when these gradients are available, it is difficult to ensure that the snake ends up at the appropriate `edge' in the image. Similar considerations recommend the partitioning approach for edges.

The first two terms in the energy expression are designed to classify the region into an inside and an outside with respect to the feature curve. These terms, F_1 (c_1, C) and F_2 (c_2, C), are defined as follows:

$$F_1(c_1, C) = \int_{v\in inside(C)} |u_0(v) - c_1|^2\, dv, \text{..}(2)$$

$$F_2(c_2, C) = \int_{v\in outside(C)} |u_0(v) - c_2|^2\, dv, \text{..}(3)$$

where C is the contour curve, c_1 is the average feature weight of the vertices *inside*(C), and c_2 that of the vertices *outside*(C). The vertices in *inside*(C) or *outside*(C) include those that lie on the curve C. A feature weight u_0 (v) is estimated at a vertex $v \in V$, where V is the set of vertices on the mesh. The curve C that minimizes F_1 (c_1, C) + F_2 (c_2, C) decomposes the domain M into a region inside and another outside C, each of which is as homogeneous as possible. Further details are described elsewhere [11][21].

To mark the vertices whose feature weights are greater than a given threshold T, we introduce another energy term F_3:

$$F_3(T,C)=\int_{v\in inside(C)}\left|T-u_0(v)\right|dv\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(4)$$

When the curve *C* that minimizes F_3 (*T*, *C*) is found, the domain inside *C* consists of vertices whose feature values u_0 (*v*) are greater than the threshold *T*. This region may not be as homogeneous as possible, because the energy term F_3 (*T*, *C*) is in conflict with the term F_1 (c_1, *C*) + F_2 (c_2, *C*). The curve *C* representing the best compromises the two terms need to be found.

The external energy *F* of our first-pass active contour can now be defined as the following weighted sum of the three terms, F_1, F_2, and F_3:

$$F(c_1,c_2,T,C)=\lambda_1 F_1(c_1,C)+\lambda_2 F_2(c_2,C)+\nu F_3(T,C)\ldots\ldots\ldots\ldots(5)$$

4.3 Level-set formulation of the energy functional

We use a level-set formulation which minimizes *F* to represent *C*. We assign level-set value *φ* to each vertex in *V* from the values of *C*, *inside*(*C*) and *outside*(*C*) [27] as follows:

$$\begin{cases}\phi(v)>0, \text{ for } v\in inside(C)=\omega,\\ \phi(v)=0, \text{ for } v\in C=\partial\omega,\\ \phi(v)<0, \text{ for } v\in outside(C)=\omega^c.\end{cases}\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(6)$$

C is the border between *ω* and ω^C, and is thus described as an interface of a level-set surface.

To represent the terms in the energy functional *F* using a level-set formulation, we introduce the Heaviside function *H* (*x*), which is 1 for non-negative *x* and 0 for negative *x*, and the delta function δ_0 (*x*), which is defined as *dH*(*x*)/*dx* [11][21][27].

Using these functions, F_1 and F_2 can be redefined level-set formulations:

$$\begin{aligned}&F_1^{LS}(c_1,\phi)=\int_V (u_0(v)-c_1)^2 H(\phi(v))dv,\\ &F_2^{LS}(c_2,\phi)=\int_V (u_0(v)-c_2)^2 (1-H(\phi(v)))dv, \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(7)\\ &\text{where } c_1=\frac{\int_V u_0(v)H(\phi(v))dv}{\int_V H(\phi(v))dv},\quad c_1=\frac{\int_V u_0(v)(1-H(\phi(v)))dv}{\int_V (1-H(\phi(v)))dv}.\end{aligned}$$

The threshold term is also redefined as follows:

$$F_3^{LS}(T,\phi)=\int_V \left|T-u_0(v)\right|H(\phi(v))dv\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(8)$$

The energy functional F^{LS}, which is a level-set formulation of *F*, can now be represented as follows:

$$\begin{aligned}F^{LS}(c_1,c_2,T,\phi)=&\lambda_1\int_V (u_0(v)-c_1)^2 H(\phi(v))dv\\ &+\lambda_2\int_V (u_0(v)-c_2)^2(1-H(\phi(v)))dv+\nu\int_V \left|T-u_0(v)\right|H(\phi(v))dv\ldots\ldots(9)\end{aligned}$$

4.4 Numerical minimization of the energy functional

By applying the calculus of variations [27], F^{LS} can be minimized by finding a value of φ at which $\partial F/\partial \varphi$, which is the gradient of F^{LS} with respect to φ, is zero. The requirement of $\partial F/\partial \varphi(\varphi) = 0$ leads to the following Euler-Lagrange equation:

$$EL(\phi) = \lambda_1 \delta_0(\phi(v))(u_0(v) - c_1)^2 + \lambda_2 \delta_0(\phi(v))(u_0(v) - c_2)^2 - \nu \delta_0(\phi(v))\left|T - u_0(v)\right| \text{...(10)}$$

This is a nonlinear equation in φ. Its solution can be transformed to the problem of solving the following differential equation:

$$\begin{aligned}\frac{\partial \phi}{\partial t} &= -EL(\phi) \\ &= -\lambda_1 \delta_0(\phi(v))(u_0(v) - c_1)^2 + \lambda_2 \delta_0(\phi(v))(u_0(v) - c_2)^2 - \nu \delta_0(\phi(v))\left|T - u_0(v)\right| \text{...(11)}\end{aligned}$$

Here t is an artificial time representing the amount of iteration involved in the optimization process. The of Equations (10) and (11) is dealt with in more detail elsewhere [11][27]. In some formulations [21][27] there is a boundary condition to be satisfied at the edge of the domain. But our domain V is a closed mesh which does not have any boundary, and so there are no boundary conditions. The numerical scheme for solving Equation (11) can be written as follows [19][28]:

$$\phi^{n+1}(v) = \phi^{n}(v) + \Delta t \delta_0(\phi(v))\left[-\lambda_1 (u_0(v) - c_1)^2 + \lambda_2 (u_0(v) - c_2)^2 - \nu \left|T - u_0(v)\right|\right] \text{....(12)}$$

As the value of the level-set function $\varphi(v)$ at vertex v changes in this scheme, it is not affected by the level-set function values at neighboring vertices. This means that the active contour used in the first pass does not consider internal energy, and so the output of the first pass is not smooth.

5. Second pass: smoothing feature regions

The raw feature regions extracted in the first pass form the input to the second pass, in which another active contour moves towards the raw feature region while staying as smooth as possible (see **Fig. 4-(a)**). At the start of the second pass, the user can make a selection from the set of raw feature regions. The level-set value $\varphi(v)$ at each vertex v is set to 1 if v lies inside a selected region, and otherwise to 0.

5.1 The active contour of a raw feature region

The second-pass is designed to find the active contour for each raw feature region, which is a polyline on the mesh that is as close as possible to the boundary of the raw feature region and as smooth as possible. Initially, the active contour of a raw feature region can lie outside or inside the boundary of that region, or it may cut its boundary. The active contour C for a raw feature region is a closed polyline consisting of a sequence of points p_i obtained as follows:

$$\mathrm{p}_i = t_i \, \mathrm{v}_i + (1 - t_i)\, \mathrm{w}_i, \text{...(13)}$$

where $e_i = (v_i, w_i)$ are the edges of the mesh (see **Fig. 4-(b)**). We call $e_i = (v_i, w_i)$ candidate edges because they may form part of the optimal active contour.

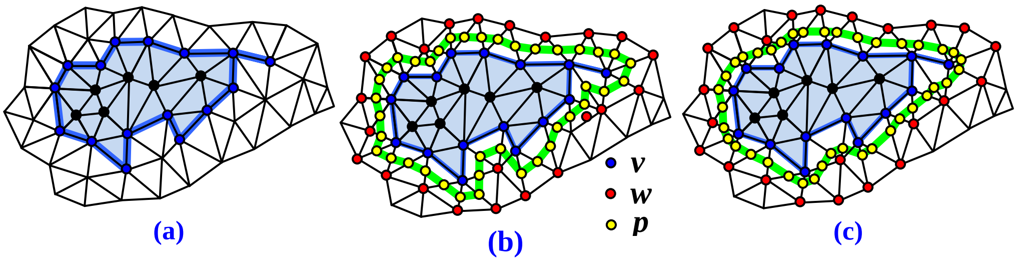


Fig. 4. Smoothing procedure in the second pass: **(a)** output of the first pass (the blue polygon is the boundary of a raw feature region), **(b)** the initial model for smoothing (green polygon), **(c)** the smoothed boundary (green polygon).

5.2 Building the energy functional for the active contour

The energy functional *J* for the second-pass active contour method is a weighted sum of the external energy *E* and the internal energy *I*. The external energy *E* represents the difference between the active contour *C* and the boundary of the raw feature region, and the internal energy *I* represents the smoothness of the active contour, which is approximated by the sum of the length and curvature of the active contour *C*:

$$J(C) = \alpha I(C) + (1-\alpha)E(C)\text{......................................}(14)$$

The level-set of the external energy *E* is:

$$E(\phi) = \int_{v \in inside(C)} (1-\phi(v))dv + \int_{v \in outside(C)} \phi(v)dv\text{.........................}(15)$$

The value $\varphi(v)$ of the level-set at a vertex *v* is 1 if *v* lies inside a raw feature region, and 0 otherwise. The vertices that contribute to the first term are those within the current active contour *C* but outside the raw feature region. The vertices that contribute to the second term are those within the raw feature region but outside the active contour. Thus *E* is minimized if the active contour encloses only the vertices belonging to the raw feature region.

The active contour of a raw feature region is computed as follows:

(a) Set the current boundary of the feature region to the boundary of the raw feature region. The active contour will be constructed relative to this boundary in the next step.

(b) Consider the set of edges $\{v_i, w_i\}$, where v_i are the vertices on the current boundary of the feature region and w_i are vertices connected to v_i. Find the active contour that passes through the edges $\{v_i, w_i\}$, which minimizes the sum of the external energy and the internal energy. The external energy of the active contour is computed with respect to the original raw feature region.

(c) Remove a vertex v_k from the boundary of the feature region, if this reduces the total energy of the optimal active contour computed in Step (b). The active contour should be constructed relative to the new boundary of the feature region. The vertices to be added or removed should be chosen in some reasonable order. Select v_k from the current boundary of the feature region to maximize the local internal energy of the active contour at vertex v_k is the maximum. Then remove v_k, if it reduces the total energy of the active contour.

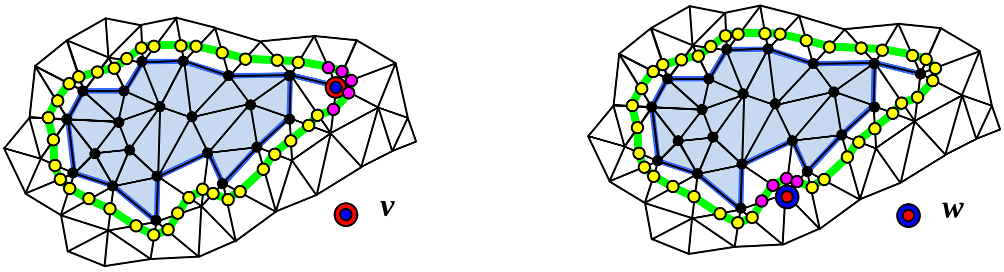
(d) Add a vertex w_k to the boundary of the feature region, if this reduces the total energy of the optimal active contour computed in Step (b). Find the vertex w_k that is connected by edges $\{v_k, w_k\}$ to the vertex v_k on the current boundary of the feature region, which minimizes the local internal energy of the active contour. Add this vertex w_k to the active contour, if it reduces its total energy.

(e) Repeat Steps (c) and (d) until the total energy of the active contour is no longer reduced.

Fig. 5 shows the process of adding or removing vertices to the candidate edges. Steps (c) and (d) involve the local internal energy I_l (C, v_i) of the active contour C near a vertex v_i on the candidate edges. I_l (C, v_i) is the sum of the arc lengths and curvatures of the points on the active contour connected to vertex v_i, expressed as follows:

$$I_l(C,v_i) \approx \sum_{i=a}^{b} \left(l_i(C) + \kappa_i(C) \right), \qquad (16)$$

where $(p_a, ..., p_b)$ are the points on the contour adjacent to v_i. We estimate I_l (C, w_i) for each vertex w_i (see **Fig. 5-(b)**).



(a) Selecting a vertex v to remove: the active contour points connected to v (shown in violet) are the points that participate in the estimation of the internal energy $I(v)$ around v.

(b) Selecting a vertex w to add: the active contour points connected to w (shown in violet) are the points that participate in the estimation of $I(w)$.

Fig. 5. Modifying the boundary of a raw feature region.

5.3 Minimizing the internal energy

Given the current boundary of the feature region, the external energy of the active contour relative to the raw feature region is fixed regardless of where the active contour lies on the candidate edges. So the optimal active contour has to be found by minimizing the internal energy. $I(C)$, the internal energy of the active contour C is the approximate length of the curve, $l(C)$, plut its approximate curvature κ (C):

$$I(C) = l(C) + \kappa(C) \qquad (17)$$

And $l(C)$ can be estimated as follows:

$$l(C)=\sum_{i=0}^{n-1} l_i^{\,2}(C),\ \text{where}\ l_i(C)=\left|p_i-p_{i+1}\right|$$
$$=\left|t_i(\mathrm{v}_i-\mathrm{w}_i)-t_{i+1}(\mathrm{v}_{i+1}-\mathrm{w}_{i+1})+\mathrm{w}_i-\mathrm{w}_{i+1}\right| \ldots\ldots.(18)$$

The edge $e_{i+1} = (v_{i+1}, w_{i+1})$ is an immediate neighbor of edge $e_i = (v_i, w_i)$ in the set of candidate edges defined above. The curvature κ (C) is approximated by the sum of the squares of the second-order differences at each point of the curve:

$$\kappa(C)=\sum_{i=0}^{n}\kappa_i^{\,2}(C),\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots.(19)$$

$$\text{where}\quad \kappa_i(C)=\left|p_{i+1}-2p_i+p_{i-1}\right|$$
$$=\left|t_{i+1}(v_{i+1}-w_{i+1})-2t_i(v_i-w_i)+t_{i-1}(v_{i-1}-w_{i-1})+w_{i+1}-2w_i+w_{i-1}\right|.$$

The internal energy I(C) is minimized by finding a set of parameters t_i that minimizes I. Since I is a quadratic form of t_i, we can find the value of t_i that minimizes I by applying the well-known method of steepest gradient descent. An initial guess at the boundary of a curve is made by setting t_i to 0.5. An example of an initial boundary curve of a feature region is shown in **Fig. 5-(b)**. The initial boundary curves on the Buddha face model are illustrated in the second figure of **Fig. 3-(b)**. **Fig. 4-(c)** shows the active contour that minimizes I while E remains unchanged. An equivalent active contour on the Buddha face model is shown in **Fig. 3-(b)**. **Fig. 3** illustrates the processes that comprise the two passes.

6. Extracting feature curves

We extract feature curves from the smooth feature regions using a skeletonization algorithm based on a peeling process. It repeatedly removes a vertex from a region until the remaining vertices form a set of curves which constitutes the skeleton of that region. The results of skeletonization strategies of this sort depend on the order of vertex removal. In t his algorithm, the order is determined by the skeleton weight of each vertex, which is the likelihood that a vertex belongs to the skeleton. The skeleton weight of a vertex v is the sum of its medial axis weight and its feature weight u_0 (v). The medial axis weight of a vertex is the shortest distance from that vertex to the boundary of the region. Thus the skeleton weight k of a vertex v can be expressed as follows:

$$k(\mathrm{v})=\omega\, u_0(\mathrm{v})+(1-\omega)\, u_1(\mathrm{v}),\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots.(20)$$

where u_1 (v) is the medial axis weight of a vertex v. ω, which is a weighting parameter that determines the relative contributions of u_0 and u_1, is determined through testing various values. After several experiments, we found out that 0.9 makes most reasonable results on most models. As stated above, vertices are removed in ascending order of k until the remaining vertices form a set of skeleton curves. These curves are then smoothed using the internal energy minimization algorithm described in Section 5.3. It is possible for a peeling-based skeletonization algorithm to produce a skeleton which is empty. To avoid the possible occurrence of this degenerate case, we determine some non-simple points among the vertices in the region from geometric considerations. These points are then protected from removal during the skeletonization process.

7. Implementation and results

Our algorithm was implemented on a PC with a Pentium IV 3.06 GHz CPU and a 2 GB main memory. We tested our algorithm on four models: the Buddha face, Venus, Male head, and Female head. The Buddha face is a part of the Stanford Buddha Model. The complexity of the models and the times required for each step of the algorithm are given in **Table 1**, and the parameters we chose for the algorithm are given in **Table 2**.

Fig. 8 shows several views of the feature regions and feature curves extracted from the three head models. Note that the algorithm was instructed to ignore the feature regions on the back of the heads of the Male and Female models since they are too noisy to get skeletonized.

Table 1. Comparison of the times required for each model

Model	No. of Vertices	No. of Features (Red/Blue)	Computation time (sec)			
			Estimating Feature weights	First pass	Second pass	Extracting Feature curves
Buddha head	35,378	14/6	0.851	2.105	3.795	3.140
Venus	67,173	96/47	1.732	3.914	5.215	4.538
Female	49,645	155/120	1.264	3.105	4.574	4.017
Male	44,986	214/60	1.195	2.887	4.492	3.792

Table 2. Table of parameters. The number in parenthesis in the first row represents the equation where the parameter is used for the first time.

Model	T (8)	λ_1 (9)	λ_2 (9)	v (9)	Δt (12)	α (14)	ω (20)
Buddha head	0.4	1.0	1.0	2.0	0.1	0.2	0.9
Venus	0.6	1.0	1.0	4.5	0.1	0.25	0.9
Female	0.4	1.0	1.0	2.0	0.1	0.15	0.9
Male	0.4	1.0	1.0	2.0	0.1	0.15	0.9

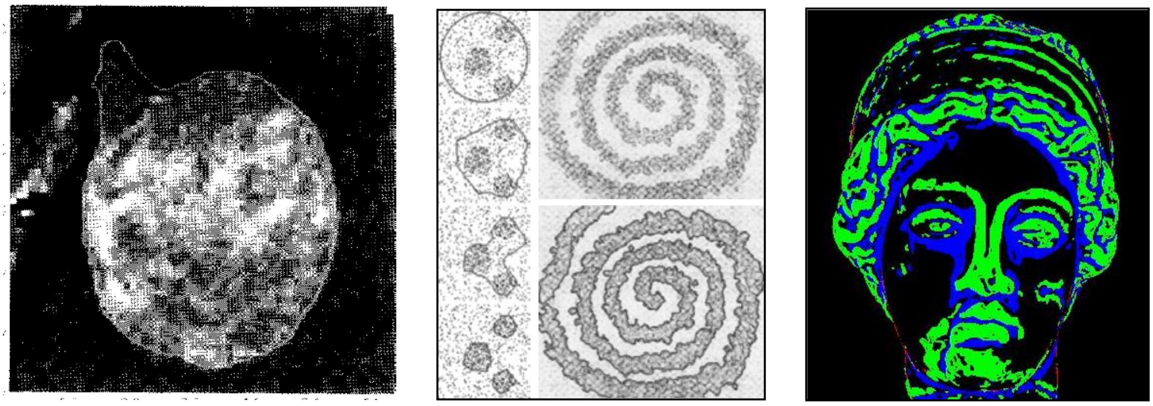
7.1 Comparison and analysis

We begin by comparing our feature regions with those extracted by three famous techniques which use active contours based on level-set surfaces [11][20][21]. Note that two techniques [20,21] operate in image domain, while the third [11] works on a triangular mesh. None of the feature regions found by three schemes are smooth, even though the extraction techniques are exhaustive (see **Fig. 6-(a)** and **8**). We consider this problem to be a consequence of the basic property of a level-set function, which only indicates whether a pixel or a vertex is inside or outside a region. This makes it difficult to obtain the first-order and second-order differential terms which are essential to achieve smooth results with an active contour method. Our scheme overcomes this difficulty by the use of separate extraction and smoothing processes.

We also compare the feature curves produced by our algorithm with those from related work, using the well-known Venus model. **Fig. 7** and **8** demonstrate that our feature curves are smoother than those produced by previous techniques: we attribute this to our smoothing of the feature regions.

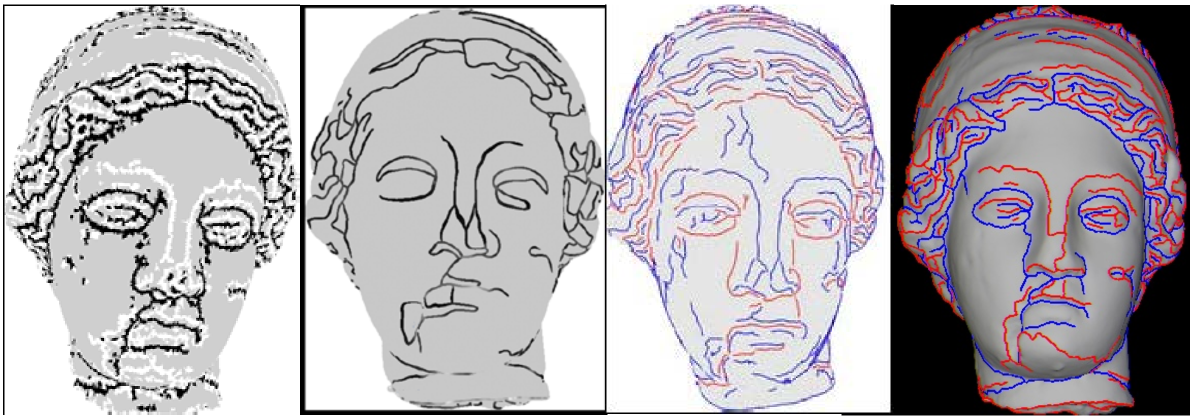
7.2 Discussion

The drawbacks of our scheme are that its results depend on the parameters shown in **Table 2**, which play a crucial role in feature extraction. We tried a diverse set of parameters and found best combinations of parameters, whose results are shown in **Fig. 8**. Another drawback is that our scheme is not suitable for extracting other types of features such as contours or suggestive contours [29]. Even though our technique can extract any features which can be described by applying weights to the vertices of a mesh, it is not an efficient way of extracting line features directly.



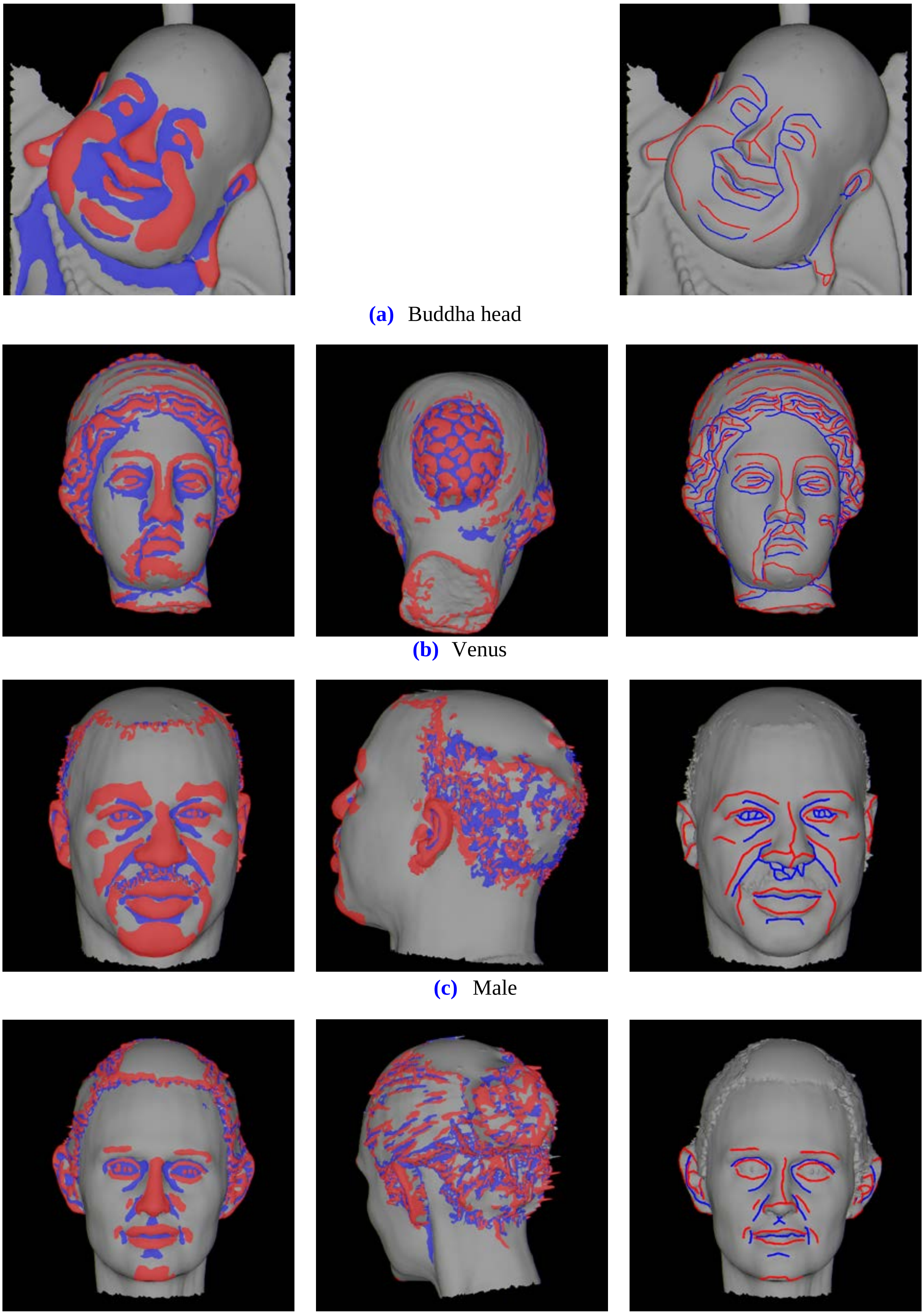
(a) [2] **(b)** [3] **(c)** [16]

Fig. 6. Results from other active contour models that use level-set surfaces



(a) [26] **(b)** [20] **(c)** [27] **(d)** [16]

Fig. 7. Feature curves from previous techniques.



(a) Buddha head

(b) Venus

(c) Male

(d) Female

Fig. 8. Features from Buddha head, Venus, Male and Female models. Blue regions are concave and red are convex; Blue curves are valleys and red are ridges.

8. Conclusion and future work

We have proposed an active contour approach to extracting feature regions from triangular meshes, using a two-pass approach in which raw feature regions extracted in a first pass are smoothed in a second pass. In addition, feature curves can subsequently be extracted from the smooth feature regions.

Furture research is required to extend the active contour technique to enable to extract features to match shape descriptions or other user-specified patterns and criteria. Another direction to investigate is the use of the features that can be extracted by our technique into rendering techniques and mesh-related operations such as morphing and editing.

References

[1] S. Gumhold, X. Wang and R. McLead, “Feature extraction from point clouds,” in *Proc. of 10th International Meshing Roundtable*, pp. 293-305, 2001. Article (CrossRefLink)

[2] A. Hubeli and M. Gross, “Multiresolutional feature extraction from unstructured meshes,” in *Proc. of IEEE Visualization*, pp. 287-294, 2001. Article (CrossRefLink)

[3] M. Pauly, R. Keiser and M. Gross, “Multi-scale feature extraction on point-sampled surfaces,” *Computer Graphics Forum*, vol. 22, no. 3, pp. 281-289, 2003. Article (CrossRefLink)

[4] C. Rossl, L. Kobbelt and H. P. Seidel, “Extraction of feature lines on triangulated surfaces using morphological operators,” in *Proc. of the AAAI spring symposium on Smart Graphics*, pp. 71-75, 2000. Article (CrossRefLink)

[5] G. Stylianou and G. Farin, “Crest lines for surface segmentation and flattening,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 10, no. 5, pp. 536-544, 2005. Article (CrossRefLink)

[6] K. Watanabe and A. Belyaev, “Detection of salient curvature features on polygonal surfaces,” *Computer Graphics Forum*, vol. 20, no. 3, pp. 385-392, 2001. Article (CrossRefLink)

[7] S. Andrew, “Interactive generation of feature curves on surfaces: A minimal path approach,” *Master’s Thesis*, Dept. of Computer Science, Univ. of Toronto, 2001.

[8] M. Jung and H. Kim, “Snaking across 3d meshes,” in *Proc. of Pacific Graphics*, pp.415-420, 2004. Article (CrossRefLink)

[9] Y. Lee and S. Lee, “Geometric snakes for triangular meshes,” *Computer Graphics Forum*, vol. 21, no. 3, pp. 229-238, 2002. Article (CrossRefLink)

[10] M. Milroy, C. Bradley and G. Vickers, “Segmentation of a wrap-around model using an active contour,” *Computer-Aided Design*, vol. 29, no. 4, pp. 299-320, 1997. Article (CrossRefLink)

[11] K. Min, D. Metaxas and M. Jung, “Active contours with level-set for extracting feature curves from triangular meshes,” in *Proc. of Computer Graphics International*, pp. 185-196, 2006. Article (CrossRefLink)

[12] S. Kim and C. Kim, “Finding ridges and valleys in a discrete surface using a modified mls approximation,” *Computer-Aided Design*, vol. 38, no. 2, pp. 173-180, 2006. Article (CrossRefLink)

[13] Y. Ohtake, A. Belyaev and H. P. Seidel, “Ridge-valley lines on meshes via implicit surface fitting,” *ACM Transactions on Graphics*, vol. 23, no. 3, pp. 609-612, 2004. Article (CrossRefLink)

[14] S. Yoshizawa, A. Belyaev and H. P. Seidel, “Fast and robust detection of crest lines on meshes,” in *Proc. of the ACM symposium on solid and physical modeling*, pp. 227-232, 2005. Article (CrossRefLink)

[15] H. Kim, H. Choi and K. Lee “Feature detection of triangular meshes based on tensor voting theory,”

Computer-Aided Design, vol. 41, no. 1, pp. 47-58, 2009. Article (CrossRefLink)

[16] J. Sahner, B.Weber, S. Prohaska and H. Lamecker, "Extraction of feature lines on surface meshes based on discrete morse theory," *Computer Graphics Forum*, vol. 27, no. 3, pp. 735-742, 2008. Article (CrossRefLink)

[17] M. Zhihong, C. Guo and Z. Mingxi, "Robust detection of perceptually salient features on 3D meshes," *The Visual Computer*, vol. 25, no. 3, pp. 289-295, 2009. Article (CrossRefLink)

[18] M. Kass, A. Witkin and D. Terzopoulos, "Snakes: Active contour models," *International Journal of Computer Vision*, vol. 1, no. 4, pp. 321-331, 1987. Article (CrossRefLink)

[19] S. Osher and J. Sethian, "Fronts propagating with curvature-dependent speed: algorithms based on hamilton-jacobi formulation," *Journal of Computational Physics*, vol. 79, no. 1, pp. 12-49, 1988. Article (CrossRefLink)

[20] V. Caselles, R. Kimmel and G. Sapiro, "Geodesic active contours," *International Journal of Computer Vision*, vol. 22, no. 1, pp. 61-79, 1997. Article (CrossRefLink)

[21] T. Chan and L. Vese, "Active contours without edges," *IEEE Transactions on Image Processng*, vol. 10, no. 2, pp. 266-277, 2001. Article (CrossRefLink)

[22] R. Malladi, J. Sethian and B.Vemuri, "Shape modeling with front propagation: A level set approach," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 2, pp. 158-175, 1995. Article (CrossRefLink)

[23] L. Kobbelt, "Discrete fairing," in *Proc. of 7th IMA Conference on the Mathematics of Surfaces*, pp. 101-131, 1997. Article (CrossRefLink)

[24] M. Meyer, M. Desbrun, P. Schroder and A. Barr, "Discrete differential-geometry operators for triangulated 2-manifolds," in *Proc. of International Workshop on Visualization and Mathematics*, pp. 35-58, 2002. Article (CrossRefLink)

[25] K. Min and M. Jung, "Segmentation of Triangular Meshes Using Multi-Scale Normal Variation," in *Proc. of International Symposium on Visual Computing*, pp. 831-840, 2006. Article (CrossRefLink)

[26] G. Taubin, "Estimating the tensor of curvature of a surface from a polyhedral approximation," in *Proc. of International Conference on Computer Vision*, pp. 902-907, 1995. Article (CrossRefLink)

[27] H. Zhao, T. Chan, B. Merriman and S. Osher, "A variational level set approach to multiphase motion," *Journal of Computational Physics*, vol. 127, no. 1, pp. 179-195, 1996. Article (CrossRefLink)

[28] L. Rudin, S. Osher and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D*, vol. 60, pp. 259-268, 1992. Article (CrossRefLink)

[29] D. DeCarlo, A. Finkelstein, S. Rusinkiewicz and A. Santella, "Suggestive contours for conveying shapes," *ACM Transactions on Graphics*, vol. 22, no. 3, pp. 848-855, 2003. Article (CrossRefLink)

Kyungha Min received his MS in Computer Science from KAIST in 1992. He received his BS and Ph.D in Computer Science and Engineering from POSTECH in 1994 and 2000, respectively. His main research interests are computer graphics and image processing.

Moon-Ryul Jung received Ph.D. in Computer Science from the University of Pennsylvania in 1992. His main research interests are artificial creatures, evolutionary algorithms and fluid simulation, with a particular interest in using them for artistic creation.