J Sci Comput
DOI 10.1007/s10915-016-0260-3

On Geodesic Curvature Flow with Level Set Formulation Over Triangulated Surfaces

Zheng Liu[1] · **Huayan Zhang**[2] · **Chunlin Wu**[3]

Received: 27 November 2015 / Revised: 14 May 2016 / Accepted: 28 July 2016

Abstract The geodesic curvature flow is an important concept in Riemannian geometry. The flow with level set formulation has many applications in image processing, computer vision, material sciences, etc. The existing discretizations on triangulated surfaces are based on either finite volume method or finite element method with piecewise linear function space, which are suitable for vertex-based two-phase problems. The contour (zero level set) in existing methods passes through triangles of the mesh. However, in some graphic applications, such as mesh segmentation (to divide a whole mesh into several sub-meshes without ambiguous triangular stripes), the cutting contour is needed to be along the edges of the mesh. Moreover, multi-phase segmentation by a single level set function is a difficult problem for a long time. In this paper, we try to tackle these two problems. We propose a new discretization which has simpler formulation and more sparse coefficient matrix. We prove the existence and uniqueness, regularization behavior and maximum–minimum principle of our discrete flow. Therein the maximum–minimum principal has not been presented before. Lots of experiments show that, the limit of the flow would be a piecewise constant solution with '*discontinuity set*' to be the closed geodesics of the surface. We therefore propose a constrained discrete geodesic curvature flow, which is also analyzed theoretically. The linear system of the constrained flow can be equivalently reformulated into a much smaller one (especially in the narrow band algorithm), which dramatically reduces the computation cost. Combined with a narrow band algorithm, the constrained flow with topologically correct

✉ Chunlin Wu
wucl@nankai.edu.cn

Zheng Liu
liu.zheng.jojo@gmail.com

Huayan Zhang
zhy101@mail.ustc.edu.cn

[1] National Engineering Research Center of Geographic Information System, China University of Geosciences, Wuhan, China

[2] School of Computer Science and Software, Tianjin Polytechnic University, Tianjin, China

[3] School of Mathematical Sciences, Nankai University, Tianjin, China

Published online: 23 August 2016

initializations (easy to be got by simple existing methods or manual inputs) yields a multi-phase segmentation method by a single level set function. We test our two flows in closed curve evolution and multi-region segmentation applications. The numerical experiments are given to demonstrate the effectiveness.

Keywords Geodesic curvature flow · Level set · Triangulated surfaces · Curve evolution · Multi-phase segmentation

1 Introduction

The geodesic curvature flow, also known as curve shortening flow, plays important roles in both theory and applications.

It is a crucial tool [20] of the closed geodesic theory, which is a fundamental part of Riemannian geometry. Since the curve is supposed to evolve under the geodesic curvature dependent velocity, it is expected to find closed geodesics on manifolds. Right now it is clear that, on Euclidean plane, any simple curve (either convex or nonconvex) converges to a round point in a finite time by the curve shortening flow, [15,17,19]. The behavior of the geodesic curvature flow on surfaces is more interesting [16,20,28,31]. Even the existence and uniqueness of solutions are proved under particular assumptions. Besides, the flow has limit curves with diverse shapes. Unlike the planar case, a closed initial curve on a manifold evolves into one of two shapes under the geodesic curvature flow. It may disappear or become closed geodesic(s). Moreover, geodesics on manifolds can be stable or unstable [31].

In applications, the flow is usually written in level set [32] formulations. It is applied extensively in image processing, computer vision, material sciences, etc. The flow gives front propagation with a curvature-dependent speed in multi-phase physical simulations and material science [32]. In image processing, morphological image scale-spaces [23] constructed by this flow provide some useful scale-space properties such as the so-called inclusion order preserving [13], which are not held by other image scale-spaces. It is also used to detect image edges [5].

Due to the interesting behavior of the flow on surfaces, several approaches have been proposed in recent years to compute and simulate it with applications on surfaces. In [10], the authors studied this flow by using standard methods for manifolds, i.e., cutting the interested manifold into a set of charts and solving geodesic curvature flows on these charts separately. Various possible cases of final curves were illustrated. Cheng et al. [8] presented numerical methods for curve evolution over implicit surfaces, where the flow equation is solved in a narrow band of 3D Cartesian grids near the surface. In [24,39], Kimmel et.al. considered the geodesic curvature flow on parametric manifolds for bending invariant scale-space concepts and edge detection of images painted on surfaces. Detailed numerical schemes were provided. Wu and Tai proposed in [42] a discrete geodesic curvature flow on triangulated surfaces by using FVM with piecewise linear function space. Several theoretical properties and applications of the discrete flow were also presented. Later on, the discrete flow in [42] was improved in [44] for interactive mesh segmentation. In [25], Lai et al. used FEM to discretize the flow on triangulated surfaces and showed successful two-region surface segmentation application for medical data.

In many computer graphic applications, such as mesh segmentation (to divide a whole mesh into several sub-meshes), the contour is needed to be along the edges of the mesh. However, the contour (zero level set) of the flows [25,42] discretized in piecewise linear

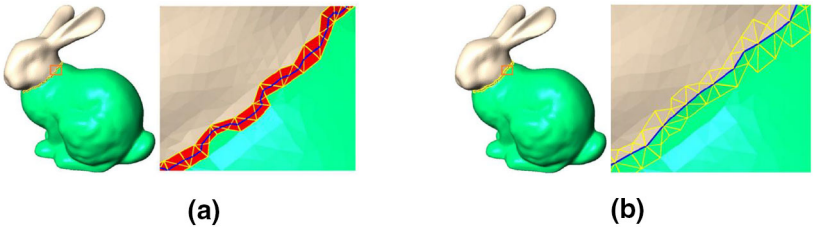

Fig. 1 Applying the geodesic curvature flow to partition the Bunny surface into 2 parts. **a** the result of discrete flow in piecewise linear function space, where the cutting contour (zero level set) is plotted in *blue*; **b** the result of discrete flow in piecewise constant function space, where the cutting contour is plotted in *blue*. As can be seen, the cutting contour in (**a**) passes through a strip of *triangles* plotted in *red*, while the cutting contour in (**b**) is along a set of *triangle edges*. To partition the surface mesh into 2 sub-meshes (while maintaining the same triangulation), a postprocessing step is needed to classify the triangles in the *red* strip in (**a**). However, this postprocessing step is avoided in (**b**) (Color figure online)

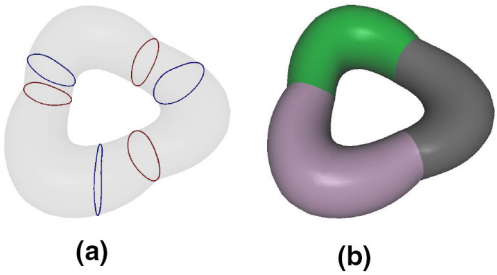

Fig. 2 Our constrained geodesic curvature flow and narrow band algorithm applied to segment the Torus surface. **a** the initial contours plotted in *blue* and the final contours in *red*; **b** the segmentation result. This surface cannot be segmented to three parts by conventional level set method, since the three contours cannot be represented by the zero level set of a function on the Torus (Color figure online)

function space usually passes through triangles of the mesh, which results in generating ambiguous triangular stripe region for surface segmentation application (see Fig. 1a). Moreover, multi-phase segmentation by a single level set function is a difficult problem for a long time. See Fig. 2 for an example. If to segment the Torus surface to three parts, it is impossible to represent the cutting contours by the zero level set of a function.

In this paper, we try to tackle above two problems. We first propose a new numerical method to discretize geodesic curvature flow on triangulated surfaces, which directly evolves curves on the triangle edges and thus avoids the postprocessing step to classify or partition the triangles in the undetermined strips; see Fig. 1b. With a different technique in existing methods, our method has several advantages. It has simpler formulation and more sparse coefficient matrix. We can prove not only the existence, uniqueness, and regularization behavior, but also maximum–minimum principal, among which the latest has not been presented in all previous methods. Lots of experiments show that, the limit of the proposed discrete flow would be a piecewise constant solution with '*discontinuity set*' to be the closed geodesics of the surface. We therefore propose a constrained geodesic curvature flow, which is also analyzed theoretically. To speedup the constrained flow, we present a new narrow band algorithm, showing multi-phase segmentation application with a single level set function on surfaces; see Fig. 2 and others in Sect. 5.3.

It has been for a long time a difficult problem for multi-phase segmentation using a single level set function. Conventional level set method [32] embedded in active contour

models [4–7] segment an image into two regions. Later several scientists proposed to use multiple level set functions for multi-phase segmentation or tracking [35,41,45]. In [3,40], a hierarchical splitting technique was presented, which iteratively splits regions obtained by traditional level set approach. Very recently, there are a few works trying to tackle this problem. [27] uses an augmented Lagrangian optimization to evolve the level set function under some special constraints, thus implementing multi-phase segmentation using a single level set function. The method avoids the regularization in Dirac function, but involves high order polynomial constraints yielding a non-convex optimization and the computational un-stability of the coefficient matrix. The method need good initializations to get correct segmentation results. In [12], a multi-region segmentation approach by a single non-negative level set function was proposed, which utilizes the Voronoi implicit interface method [36,37]. The key of this method is an accurate and fast computation of unsigned distance function through Eikonal equation at each evolution step, together with two evolving ε-level sets to encapsulate the segmentation contours. In contrast, we design in this paper a multi-phase segmentation method with a single level set function by constrained geodesic curvature flow and a narrow band algorithm. As long as the initialization has correct topology, our method can obtain good segmentation results. Our method avoids to solve the Eikonal equation which is not a trivial work [14,34] on triangulated surfaces. It does not need to solve a non-convex optimization problem involving high order polynomial constraints. Besides, our algorithm is easy to implement on triangulated meshes.

This paper is organized as follows. In Sect. 2, we review the conventional level set formulation of geodesic curvature flow on smooth 2-manifolds. In Sect. 3, we propose a new discrete geodesic curvature flow in piecewise constant function space with theoretical analysis, and compare it with the discrete flow in [42]. By the observation of a large number of experiments and the discussion at the end of Sect. 3, we propose a novel constrained geodesic curvature flow in Sect. 4 with theoretical analysis. The system of the constrained flow can be equivalently reformulated into a much smaller one, which dramatically reduces the computation cost. A new narrow band algorithm will also be presented to combine the constrained geodesic curvature flow for multi-phase segmentation. In Sect. 5, the applications of two discrete flows in closed curve evolution and multi-phase segmentation on triangulated surfaces are discussed. Section 6 concludes the paper.

2 Geodesic Curvature Flow Equation Over Smooth 2-Manifolds

In this section, we review geodesic curvature flow equation with a level set formulation [32] over smooth 2-manifolds.

Assume that $\mathcal{M} \subset \mathbb{R}^3$ is a 2-dimensional smooth manifold and ∇, div are intrinsic gradient and divergence operators on $\mathcal{M}$. Suppose that $\mathcal{C} \subset \mathcal{M}$ is a curve defined on $\mathcal{M}$. Assume that $w : \mathcal{M} \to \mathbb{R}^+$ is a positive scalar function serving as a weight, which depends on applications. Then, the weighted length of $\mathcal{C}$ is

$$E(\mathcal{C}) = \int_{\mathcal{C}} w dl. \tag{1}$$

By the gradient descent method, we use the Euler–Lagrange equation to minimize (1) (see [5,39] for details). Then, we have

$$\mathcal{C}_t = (\kappa w - \langle \nabla w, \mathcal{N} \rangle)\mathcal{N}, \tag{2}$$

where t is an artificial time parameter, κ is the geodesic curvature and $\mathcal{N}$ is the unit normal of curve $\mathcal{C}$. This equation evolves the initial curve $\mathcal{C}_0$ towards a local minima of the weighted length of $\mathcal{C}$.

Suppose that $\phi : \mathcal{M} \to \mathbb{R}$ is a flow equation and $\mathcal{C}$ can be seen as the zero level set of ϕ. According to the work of Osher and Sethian [32], the curve evolution Eq. (2) can be reformulated into the level set form as follows:

$$\begin{cases} \phi_t = |\nabla\phi| \mathrm{div}\left(w \frac{\nabla\phi}{\sqrt{|\nabla\phi|^2+\beta}} \right) \\ \frac{\partial\phi}{\partial n}\big|_{\partial\mathcal{M}} = 0 \\ \phi(t=0) = \phi_0, \end{cases} \tag{3}$$

where β is a small positive number introduced to avoid division by zero (we use $\beta = 0.00001$ in all the examples), and ϕ_0 is a given initial value of the flow function. The level set method, proposed by Osher and Sethian, converts the curve shortening problem into the problem of finding the steady solution ($\phi_t = 0$) of (3). We mention that, in Euclidean space, this derivation of the evolution equation can be found in Sethian's book [38].

As mentioned, several numerical methods of this flow on surfaces have been proposed during the last few years [8,25,39,42]. In next section, we propose a new discretization on triangulated surfaces with a simpler formulation and more theoretical discussions than previous ones.

3 Discretization and Analysis of Geodesic Curvature Flow Over Triangulated Surfaces

3.1 Notation

Assume that M is a compact triangulated surface of arbitrary topology with no degenerate triangles in $\mathbb{R}^3$. The set of vertices, edges and triangles of M are denoted as $\{v_i : i = 0, 1, \ldots, \mathrm{V}-1\}$, $\{e_i : i = 0, 1, \ldots, \mathrm{E}-1\}$ and $\{\tau_i : i = 0, 1, \ldots, \mathrm{T}-1\}$, respectively. Here V, E and T are the numbers of vertices, edges and triangles of the triangulated surface, respectively. If v is an endpoint of an edge e, then we denote it as $v \prec e$. Similarly, that e is an edge of a triangle τ is denoted as $e \prec \tau$; that v is a vertex of a triangle τ is denoted as $v \prec \tau$. Let $D_1(i)$ be the 1-ring of the triangle τ_i, which are the triangles sharing some common edges with τ_i.

We further introduce the relative orientation of an edge e to a triangle τ, which is denoted by $\mathrm{sgn}(e, \tau)$ as follows. Assume first that all the triangles are with anticlockwise orientation and all edges are with fixed orientations which are randomly chosen. For an edge $e \prec \tau$, if the orientation of e is consistent with the orientation of τ, then $\mathrm{sgn}(e, \tau) = 1$; otherwise $\mathrm{sgn}(e, \tau) = -1$.

3.2 Piecewise Constant Function Space and Operators

In this subsection, we introduce piecewise constant function space, which is used to describe scalar piecewise constant data field. We define the space $V_M = \mathbb{R}^{\mathrm{T}}$, which is isomorphic to the piecewise constant function space over M. For example, $u = (u_0, u_1, \ldots, u_{\mathrm{T}-1}) \in V_M$. It means that the value of u restricted on the triangle τ is u_τ, which is written as $u|_\tau$ sometimes.

For any $u^1, u^2, u \in V_M$, we define the inner product and norm as follows:

$$(u^1, u^2)_{V_M} = \sum_{\tau} u^1|_\tau u^2|_\tau s_\tau, \quad \|u\|_{V_M} = \sqrt{(u,u)_{V_M}}, \tag{4}$$

where s_τ is the area of triangle τ.

For any $u \in V_M$, we define the jump of u over an edge e as

$$[u]_e = \begin{cases} \sum\limits_{\tau, e\prec\tau} u|_\tau \mathrm{sgn}(e,\tau), & e \nsubseteq \partial M \\ 0, & e \subseteq \partial M \end{cases}. \tag{5}$$

Due to the piecewise constant function space, it is natural to define the gradient operator by

$$\nabla : u \to \nabla u, \quad \nabla u|_e = [u]_e, \quad \forall e, \quad \text{for} \quad u \in V_M.$$

It can be regarded as the signed amplitude of the usual vector definition of the gradient. However, in real computation this simplified definition is enough. In the vector definition, one need choose a tangent space at an edge. This tangent space (at an edge) is ambiguous. A discussion on this can be found in [43].

We then denote the range of ∇ by Q_M, i.e., $Q_M = \mathrm{Range}(\nabla)$. The Q_M space is equipped with the following inner product and norm:

$$(p^1, p^2)_{Q_M} = \sum_{e} p^1|_e p^2|_e l_e, \quad \|p\|_{Q_M} = \sqrt{(p,p)_{Q_M}}, \tag{6}$$

for $p^1, p^2, p \in Q_M$, where l_e is the length of the edge e.

It is straightforward to derive the adjoint operator of $-\nabla$, say, the divergence operator $\mathrm{div} : Q_M \to V_M$, by using the above inner products in V_M and Q_M. For $p \in Q_M$, $\mathrm{div} p$ is given by

$$(\mathrm{div} p)|_\tau = -\frac{1}{s_\tau} \sum_{e\prec\tau} p|_e \mathrm{sgn}(e,\tau) l_e, \forall \tau. \tag{7}$$

3.3 Discrete Geodesic Curvature Flow

We approximate the geodesic curvature flow Eq. (3) at each face of M. For one face τ_i, it reads

$$\phi_t|_{\tau_i} = |(\nabla\phi)|_{\tau_i}| \left(\mathrm{div} \left(w \frac{\nabla\phi}{\sqrt{|\nabla\phi|^2 + \beta}} \right) \right)|_{\tau_i}. \tag{8}$$

After approximating $|(\nabla\phi)|_{\tau_i}|$ outside the divergence operator by the average of the gradient on each edge of face τ_i, we have

$$\phi_t|_{\tau_i} = \frac{\sum\limits_{e\prec\tau_i} |[\phi]_e| l_e}{\sum\limits_{e\prec\tau_i} l_e} \left(\mathrm{div} \left(w \frac{\nabla\phi}{\sqrt{|\nabla\phi|^2 + \beta}} \right) \right)|_{\tau_i}, \tag{9}$$

where $[\phi]_e = \nabla\phi|_e$ (see gradient operator (5)). The divergence operator (7) gives

$$\left(\mathrm{div} \left(w \frac{\nabla\phi}{\sqrt{|\nabla\phi|^2 + \beta}} \right) \right)|_{\tau_i} = -\frac{1}{s_{\tau_i}} \sum_{e\prec\tau_i} \frac{w_e l_e}{\sqrt{|[\phi]_e|^2 + \beta}} \left([\phi]_e \mathrm{sgn}(e,\tau_i) \right),$$

where w_e is a weight function defined on each edge. According to (5), we verify that $[\phi]_e \mathrm{sgn}(e,\tau_i) = (\phi_i \mathrm{sgn}(e,\tau_i) + \phi_j \mathrm{sgn}(e,\tau_j))\mathrm{sgn}(e,\tau_i) = \phi_i - \phi_j$, where $e = \tau_i \cap \tau_j$ and $j \in D_1(i)$.

Therefore, the semi-implicit discretization of (9) can be written as

$$\frac{\phi_i^{n+1}-\phi_i^n}{\Delta t}=-\frac{1}{s_{\tau_i}}\frac{\sum\limits_{e\prec\tau_i}|[\phi^n]_e|l_e}{\sum\limits_{e\prec\tau_i}l_e}\sum_{\substack{e=\tau_i\cap\tau_j,\\ j\in D_1(i)}}\frac{w_e l_e}{\sqrt{|[\phi^n]_e|^2+\beta}}\left(\phi_i^{n+1}-\phi_j^{n+1}\right). \tag{10}$$

Denoting $\Phi^{(n)}=(\phi_0^n,\phi_1^n,\ldots,\phi_{\mathrm{T}-1}^n)'$, then Eq. (10) is formulated into the following matrix form:

$$\left(S+\Delta t G(\Phi^{(n)})H(\Phi^{(n)})\right)\Phi^{(n+1)}=S\Phi^{(n)},$$

where $S=\mathrm{diag}(s_{\tau_0},s_{\tau_1},\ldots,s_{\tau_{\mathrm{T}-1}})$,

$$G(\Phi^{(n)})=\mathrm{diag}\left(\frac{\sum\limits_{e\prec\tau_0}|[\phi^n]_e|l_e}{\sum\limits_{e\prec\tau_0}l_e},\frac{\sum\limits_{e\prec\tau_1}|[\phi^n]_e|l_e}{\sum\limits_{e\prec\tau_1}l_e},\ldots,\frac{\sum\limits_{e\prec\tau_{\mathrm{T}-1}}|[\phi^n]_e|l_e}{\sum\limits_{e\prec\tau_{\mathrm{T}-1}}l_e}\right)$$

are two diagonal matrices and $H(\Phi^{(n)})=h_{ij}$ with

$$h_{ij}=\begin{cases}\sum\limits_{e\prec\tau_i}\dfrac{w_e l_e}{\sqrt{|[\phi^n]_e|^2+\beta}}, & j=i\\ -\dfrac{w_e l_e}{\sqrt{|[\phi^n]_e|^2+\beta}}, & j\in D_1(i),\quad \text{where}\quad e=\tau_i\cap\tau_j\\ 0, & \text{others.}\end{cases} \tag{11}$$

Let Φ_0 be an initial flow function. Given a fixed time step Δt, we can calculate a sequence $\{\Phi^{(n)},n=0,1,2,\ldots.\}$ as follows:

$$\begin{cases}(S+\Delta t G(\Phi^{(n)})H(\Phi^{(n)}))\Phi^{(n+1)}=S\Phi^{(n)}\\ \Phi^{(0)}=\Phi_0.\end{cases} \tag{12}$$

For the convenience of description, we call the sequence $\{\Phi^{(n)},n=0,1,2,\ldots\}$ computed by (12) as the *discrete geodesic curvature flow* of the initial flow function Φ_0. Compared to previous discretizations, our discrete flow has simpler formulation and more sparse coefficient matrices at each iteration.

Remark We should point out that the concept of the *discrete geodesic curvature flow* can also be defined for time sequences with variable time steps Δt. For convenience we use fixed time step Δt in this paper.

3.4 Analysis of the Discrete Geodesic Curvature Flow

In this subsection, we present several fundamental theoretical aspects of *discrete geodesic curvature flow*. They include existence and uniqueness, regularization behavior and maximum–minimum principle. The assumption of no degenerate triangles gives the boundedness of the discrete curvatures [30] and the nonzero triangle areas. These facts, together with the boundedness of the coefficients defined in (11), help to establish these theoretical results. Starting from the following three lemmas, one can verify the existence and uniqueness, as well as the regularization behavior, by the techniques similar to [42]. Meanwhile, the maximum–minimum principal will be presented. To the best of our knowledge, the maximum–minimum principal is not be presented in any existing methods.

Lemma 1 *Assume that D is a diagonal matrix with nonnegative elements and A is symmetric positive semidefinite. Then, the principal minors of the matrix DA are all nonnegative.*

Proof Let $D = \mathrm{diag}(d_1, d_2, \ldots, d_{\mathrm{T}})$. The assertion follows immediately from

$$(DA)\begin{pmatrix} i_1 & i_2 & \ldots & i_r \\ i_1 & i_2 & \ldots & i_r \end{pmatrix} = d_{i_1} \ldots d_{i_r} A \begin{pmatrix} i_1 & i_2 & \ldots & i_r \\ i_1 & i_2 & \ldots & i_r \end{pmatrix}, \tag{13}$$

for any $1 \le i_1 < \ldots < i_r \le \mathrm{T}$. □

Lemma 2 *Assume A is a $\mathrm{T} \times \mathrm{T}$ matrix and λ is a constant. Then,*

$$det(\lambda I + A) = \lambda^{\mathrm{T}} + \sum_{1 \le k \le \mathrm{T}} \lambda^{\mathrm{T}-k} \sum_{i_1 < i_2 < \ldots < i_k} A \begin{pmatrix} i_1 & i_2 & \ldots & i_k \\ i_1 & i_2 & \ldots & i_k \end{pmatrix}, \tag{14}$$

where I is an identity matrix and

$$A \begin{pmatrix} i_1 & i_2 & \ldots & i_k \\ i_1 & i_2 & \ldots & i_k \end{pmatrix}$$

is a principal minor of A.

Proof This can be proved in a way similar to that given in ([11], pp.180–182). □

Lemma 3 *The matrix $H(\Phi^{(n)})$ with elements defined via (11) is symmetric and positive semidefinite with* $\mathrm{rank}(H) = \mathrm{T} - 1$.

Proof Since $h_{ij} = h_{ji}$, the symmetry of $H(\Phi^{(n)})$ is obvious. On the other hand, for any vector v, we have

$$\begin{aligned} v' H(\Phi^{(n)}) v &= \sum_{ij} h_{ij} v_i v_j = \sum_i h_{ii} v_i^2 + \sum_i \sum_{j \in D_1(i)} h_{ij} v_i v_j \\ &= \sum_i \sum_{e \prec \tau_i} \frac{w_e l_e}{\sqrt{|[\phi^n]_e|^2 + \beta}} v_i^2 - \sum_i \sum_{\substack{e = \tau_i \cap \tau_j, \\ j \in D_1(i)}} \frac{w_e l_e}{\sqrt{|[\phi^n]_e|^2 + \beta}} v_i v_j \\ &= \sum_{e = \tau_i \cap \tau_j} \frac{w_e l_e}{\sqrt{|[\phi^n]_e|^2 + \beta}} (v_i^2 + v_j^2) - 2 \sum_{e = \tau_i \cap \tau_j} \frac{w_e l_e}{\sqrt{|[\phi^n]_e|^2 + \beta}} v_i v_j \\ &= \sum_{e = \tau_i \cap \tau_j} \frac{w_e l_e}{\sqrt{|[\phi^n]_e|^2 + \beta}} (v_i - v_j)^2 \ge 0. \end{aligned}$$

Therefore, the matrix $H(\Phi^{(n)})$ is positive semidefinite. Moreover, the positive definiteness of w_e and l_e indicates that $\mathrm{rank}(H) = \mathrm{T} - 1$. □

Theorem 1 (existence and uniqueness) *For any initial value Φ_0, with a fixed time step Δt and $t^0 = 0$, there exists a unique discrete geodesic curvature flow $\{\Phi^{(n)}, n = 0, 1, 2, \ldots\}$.*

Proof In fact, we only need to prove the coefficient matrix of the system (12) is invertible. We compute the determinant of $(S + \Delta t G(\Phi^{(n)}) H(\Phi^{(n)}))$. Let

$$\lambda_i = \frac{\sum_{e \prec \tau_i} |[\phi]_e^n| l_e}{\sum_{e \prec \tau_i} l_e}, \quad i = 0, 1, \ldots, \mathrm{T} - 1,$$

we have

$$\begin{aligned}
&\det(S + \Delta t G(\Phi^{(n)})H(\Phi^{(n)})) \\
&\quad = \det(S + \Delta t \ \mathrm{diag}(\lambda_0, \lambda_1, \ldots, \lambda_{\mathrm{T}-1})H(\Phi^{(n)}) \\
&\quad = \det(S) \cdot \det(I + \Delta t \ \mathrm{diag}(\frac{\lambda_0}{s_0}, \frac{\lambda_1}{s_1}, \ldots, \frac{\lambda_{\mathrm{T}-1}}{s_{\mathrm{T}-1}})H(\Phi^{(n)})) \\
&\quad := \det(S) \cdot \det(I + \Lambda H) \\
&\quad = \det(S) \cdot \left(1 + \sum_{1 \le k \le \mathrm{T}} \sum_{i_1 < i_2 < \ldots < i_k} \Lambda H \begin{pmatrix} i_1 \ i_2 \ \ldots \ i_k \\ i_1 \ i_2 \ \ldots \ i_k \end{pmatrix}\right) \\
&\quad > 0,
\end{aligned}$$

by Lemmas 1, 2 and 3. This proves the invertibility of the coefficient matrix of the system (12). □

Theorem 2 (regularization behavior) *For the flow function $\Phi(t^{(n)})$, we assume that $L_1 = \{i | \phi_j^n = \phi_i^n, j \in D_1(i)\}$ and $L_2 = \{0, 1, \ldots, \mathrm{T}-1\} \backslash L_1$ are two index sets. Then, $(I + \Delta t S^{-1} G(\Phi^{(n)})H(\Phi^{(n)}))^{-1}$ has eigenvalues $0 < \mu_0, \mu_1, \ldots, \mu_{\mathrm{T}-1} \le 1$ with corresponding eigenvectors $\{b_i = b_{i,0}, b_{i,1}, \ldots, b_{i,\mathrm{T}-1}\}$, where $i = 0, 1, \ldots, \mathrm{T}-1$, which is complete. Moreover,*

1. *If L_1 is empty, then the largest eigenvalue $\mu_{max} = 1$ with a unique eigenvector $(1, 1, \ldots, 1)$;*

2. *If L_1 is nonempty, then for all $i \in L_1$, $\mu_i = 1$ is the eigenvalue with $(1, 1, \ldots, 1)$ as one of the corresponding eigenvectors; for $i \in L_2$, $0 < \mu_i < 1$ and $b_{ij} = 0$, $j \in L_1$ in its corresponding eigenvector b_i.*

Proof Let

$$D = S^{-1} G(\Phi^{(n)}),$$

which is a diagonal matrix with nonnegative elements.

1. Since L_1 is empty, it is obvious that G is invertible. We then have

$$\begin{aligned}
S^{-1} G(\Phi^{(n)})H(\Phi^{(n)}) = DH &= \sqrt{D}\sqrt{D}H \\
&\sim (\sqrt{D})^{-1}\sqrt{D}\sqrt{D}H\sqrt{D} \\
&= \sqrt{D}H\sqrt{D},
\end{aligned}$$

implying that DH is similar to a symmetric matrix $\sqrt{D}H\sqrt{D}$. Therefore, $I + \Delta t S^{-1} G(\Phi^{(n)})H(\Phi^{(n)})$ is similar to a symmetric matrix, and its inverse. This imply that all the eigenvalues of $I + \Delta t S^{-1} G(\Phi^{(n)})H(\Phi^{(n)})$ are real and the set of eigenvectors is complete. On the other hand, as $H = (\sqrt{D})^{-1}\sqrt{D}H\sqrt{D}(\sqrt{D})^{-1}$, H and $\sqrt{D}H\sqrt{D}$ are then congruent. This shows that the eigenvalues of $\sqrt{D}H\sqrt{D}$ are all greater than or equal to zero. Therefore, the eigenvalues of DH are also greater than or equal to zero.

Now, we assume λ is an eigenvalue of DH with corresponding eigenvector b and have

$$DHb = \lambda b \Rightarrow Hb = \lambda D^{-1} b \Rightarrow b'Hb = \lambda b' D^{-1} b \Rightarrow \lambda = \frac{b'Hb}{b'D^{-1}b},$$

then using Lemma 3, we have $\lambda \ge 0$. For eigenvalue of DH is λ, it is obvious that eigenvalue of $(I + \Delta t DH(\Phi^{(n)}))^{-1}$ is

$$\mu = (1 + \Delta t \lambda)^{-1}. \tag{15}$$

Moreover, by Lemma 3, we know that the minimal eigenvalue of the matrix H is 0 with a unique eigenvector $(1, 1, \ldots, 1)$, therefore, the assertion follows.

2. If L_1 is nonempty, the matrix G is not invertible. For convenience, we use a series of permutations to relocate the zero diagonal elements of D together to the later part of the diagonal line. Denoting the element number of L_1 as $|L_1| = \mathrm{T} - r$, there exists an orthogonal matrix P such that

$$D = P^{-1} \begin{pmatrix} D_{rr} & 0 \\ 0 & 0 \end{pmatrix} P,$$

where D_{rr} is an $r \times r (r < \mathrm{T})$diagonal submatrix with positive diagonal elements. Hence,

$$\begin{aligned} & S^{-1} G(\Phi^{(n)}) H(\Phi^{(n)}) = DH(\Phi^{(n)}) \\ & = P^{-1} \begin{pmatrix} D_{rr} & 0 \\ 0 & 0 \end{pmatrix} PH(\Phi^{(n)}) \sim \begin{pmatrix} D_{rr} & 0 \\ 0 & 0 \end{pmatrix} PH(\Phi^{(n)}) P' \\ & := \begin{pmatrix} D_{rr} & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} B_{rr} & B_{rR} \\ B_{Rr} & B_{RR} \end{pmatrix} = \begin{pmatrix} D_{rr} B_{rr} & D_{rr} B_{rR} \\ 0 & 0 \end{pmatrix} \end{aligned}$$

where $R = \mathrm{T} - r$ and B_{rr} is a symmetric positive definite matrix according to Lemma 3.

We then analyze the matrix $\begin{pmatrix} D_{rr} B_{rr} & D_{rr} B_{rR} \\ 0 & 0 \end{pmatrix}$. By a similar argument as above, one can show that the eigenvalues of $D_{rr} B_{rr}$ are positive. Therefore, we can denote all the eigenvalues by $\lambda_0, \lambda_1, \ldots, \lambda_{r-1}, \lambda_r = \lambda_{r+1} = \ldots = \lambda_{\mathrm{T}-1} = 0$ with corresponding eigenvectors $e_0, e_1, \ldots, e_{\mathrm{T}-1}$.

In the next, we reveal the structures of e_i and show that for each λ_i, its algebraic multiplicity equals to its geometric multiplicity. Consider the following system of equations:

$$\begin{pmatrix} D_{rr} B_{rr} & D_{rr} B_{rR} \\ 0 & 0 \end{pmatrix} \begin{pmatrix} e_{i,0} \\ \vdots \\ e_{i,r-1} \\ e_{i,r} \\ \vdots \\ e_{i,\mathrm{T}-1} \end{pmatrix} = \lambda_i \begin{pmatrix} e_{i,0} \\ \vdots \\ e_{i,r-1} \\ e_{i,r} \\ \vdots \\ e_{i,\mathrm{T}-1} \end{pmatrix}, \quad i = 0, 1, \ldots, \mathrm{T} - 1.$$

For $i = 0, \ldots, r-1$, $e_{i,r} = e_{i,r+1} = \ldots = e_{i,\mathrm{T}-1} = 0$ and the eigenvector e_i can be determined by the sub linear system of $D_{rr} B_{rr}$. As $D_{rr} B_{rr}$ is similar to a real symmetric matrix, it is then similar to a diagonal matrix. Therefore, the algebraic multiplicity and geometric multiplicity of each λ_i are the same. For $i = r, r+1, \ldots, \mathrm{T}-1$, $\lambda_i = 0$, its geometric multiplicity is

$$\mathrm{T} - \mathrm{rank} \begin{pmatrix} D_{rr} B_{rr} & D_{rr} B_{rR} \\ 0 & 0 \end{pmatrix} = \mathrm{T} - r,$$

which is exactly the algebraic multiplicity. Therefore, the set of eigenvectors $\{e_i, i = 0, 1, \ldots, \mathrm{T}-1\}$ is complete.

In all, the eigenvalues of DH satisfy

$$\begin{cases} \lambda_i = 0, \ i \in L_1 \\ \lambda_i > 0, \ i \in L_2. \end{cases}$$

Now, we come back to $(I + \Delta t DH(\Phi^n))^{-1}$. Its eigenvalues $\{\mu_i, i = 0, 1, \ldots, \mathrm{T}-1\}$ satisfy

$$\begin{cases} \mu_i = 1, & i \in L_1 \\ 0 < \mu_i < 1, & i \in L_2, \end{cases}$$

and the corresponding eigenvectors $\{b_i, i = 0, 1, \ldots, \mathrm{T}-1\}$ satisfy

$$b_i = \begin{cases} (1, 1 \ldots, 1), & i \in L_1 \\ (b_{i,0}, \ldots, b_{i,k}, \ldots, b_{i,\mathrm{T}-1}) \quad \text{with} \quad \{b_{i,k} = 0, k \in L_1\}, & i \in L_2. \end{cases}$$

Therefore, the assertion follows. □

Corollary 1 (stability) *For any initial value $\Phi^{(0)}$, with a fixed time step Δt and $t^0 = 0$, we have that*

$$\|\Phi^{(n+1)}\|_2 \leq \|\Phi^{(n)}\|_2.$$

Proof The assertion follows immediately from Theorem 2. □

The regularization behavior can be interpreted as follows. Suppose that current flow function is $\Phi^{(n)}$. There are two case: L_1 is empty and L_1 is nonempty. If L_1 is empty, the eigenvalue $\mu_{max} = 1$ has a unique constant vector $(1, 1, \ldots, 1)$, which means that the zero frequency component along the eigenvector $(1, 1, \ldots, 1)$ is the only one retained and other components with higher frequency will shrink. If L_1 is nonempty, we can decompose the $\Phi^{(n)}$ as

$$\Phi^{(n)} = \sum_{i \in L_1} a_i b_i + \sum_{i \in L_2} a_i b_i, \tag{16}$$

and

$$\Phi^{(n+1)} = \sum_{i \in L_1} a_i b_i + \sum_{i \in L_2} \mu_i a_i b_i, \tag{17}$$

where we use the result that $\mu_i = 1$ for $i \in L_1$. It is implied that in (17) components $a_i b_i$ for $i \in L_1$ keep unchanged while other $a_i b_i$ for $i \in L_2$ shrink. According to Theorem 2, in the decomposition of (17), the first summation $\sum_{i \in L_1} a_i b_i$ are fully preserved, the components of the second summation $\sum_{i \in L_2} \mu_i a_i b_i$ satisfy $b_{i,j} = 0, j \in L_1$. It is implied that only the first summation $\sum_{i \in L_1} a_i b_i$ contributes to $\phi_j^n, j \in L_1$, while the second summation not contribute to it. From the definition of L_1, we know that these unchanged $\phi_j^n, j \in L_1$ should be piecewise constant. On the other hand, the shrinking components $a_i b_i, i \in L_2$ have the supports in transition domains between piecewise constant region of $\Phi^{(n)}$ should shrink as high-frequency signals. In all, the regularization effect of the discrete flow can be concluded as that, *constant or piecewise constant components preserve while high-frequency components supported in transition domains shrink.*

Theorem 3 (maximum-minimum principle) *Let $\Phi^{(0)}$ be an initial value and*

$$m(\Phi^{(0)}) := \min_{j=0,1,\ldots,\mathrm{T}-1} \phi_j^0, \qquad M(\Phi^{(0)}) := \max_{j=0,1,\ldots,\mathrm{T}-1} \phi_j^0,$$

then, $m(\Phi^{(0)}) \leq \phi_i^n \leq M(\Phi^{(0)})$ for $i = 0, 1, \ldots, \mathrm{T}-1, n = 1, 2, \ldots$.

Proof We have

$$(S + \Delta t G(\Phi^{(n)}) H(\Phi^{(n)})) \Phi^{(n+1)} = S \Phi^{(n)}$$

so

$$\begin{aligned} \Phi^{(n+1)} &= (S + \Delta t G(\Phi^{(n)}) H(\Phi^{(n)}))^{-1} S \Phi^{(n)} \\ &= (I + \Delta t S^{-1} G(\Phi^{(n)}) H(\Phi^{(n)}))^{-1} \Phi^{(n)} \\ &:= A \Phi^{(n)}, \end{aligned} \tag{18}$$

where $A = (a_{ij})_{\mathrm{T}\times\mathrm{T}}$. Since $G(\Phi^{(n)})H(\Phi^{(n)})$ is positive semidefinite (by Lemma 3), from Theorem 2.1 on page 3 in [33] we can see that $I + \Delta t S^{-1} G(\Phi^{(n)})H(\Phi^{(n)})$ is an M-Matrix. Therefore, the elements of A are all nonnegative. On the other hand, by Lemma 3, we can easily verify that the sum of each row of $I + \Delta t S^{-1} G(\Phi^{(n)})H(\Phi^{(n)})$ is unit and hence of A. Therefore,

$$\phi_i^{n+1} = \sum_j a_{ij}\phi_j^n \le \max_j \phi_j^n \sum_j a_{ij} = \max_j \phi_j^n$$

and

$$\phi_i^{n+1} = \sum_j a_{ij}\phi_j^n \ge \min_j \phi_j^n \sum_j a_{ij} = \min_j \phi_j^n$$

hold for all i and n. The assertion follows this immediately. □

We point out that, by similar techniques as above, the discretization method in [42] can be shown having the maximuml–minimum principal for meshes with no obtuse triangles. In contrast, our discretization in piecewise constant function space can remove this restriction to obtain the principle.

3.5 Differences Between our Flow and the Flow in [42]

The flow proposed by Wu and Tai [42] is discretized in piecewise linear function space, while ours is discretized in piecewise constant function space. Basically, the discretization in piecewise linear function space is suitable to deal with vertex-based problems. In contrast, the discretization in piecewise constant function space is suitable to deal with face-based problems. The definitions of basis functions and differential operators of these two discretizations are totally different.

For clarity, we give a brief review of the discretization in [42]. We denote $N_1(i)$ as the 1-neighborhood of vertex v_i, which is the indices of vertices connecting to v_i; see Fig. 3b. Let $M_1(i)$ be the 1-disk of the vertex v_i. $M_1(i)$ is the indices of triangles containing v_i; see Fig. 3c. The dual cell of vertex v_i is showed in Fig. 4a. We denote the area of the dual cell of vertex v_i as c_i.

We denote ψ as the level set function of the geodesic curvature flow in piecewise linear function space. Denoting $\Psi^{(n)} = (\psi_0^n, \psi_1^n, \ldots, \psi_{\mathrm{V}-1}^n)'$. Then the discrete flow in [42] is as follows:

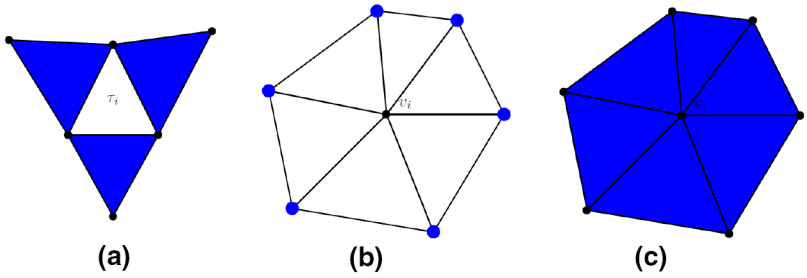

Fig. 3 The illustration of $D_1(i)$, $N_1(i)$ and $M_1(i)$. The elements contained in $D_1(i)$, $N_1(i)$ and $M_1(i)$ are plotted in *blue* respectively. **a** $D_1(i)$ is the 1-ring of the *triangle* τ_i, which refers to 3 *triangles*; **b** $N_1(i)$ is the 1-neighborhood of vertex v_i, which refers to 6 vertices; **c** $M_1(i)$ is the 1-disk of vertex v_i; which refers to 6 *triangles* (Color figure online)

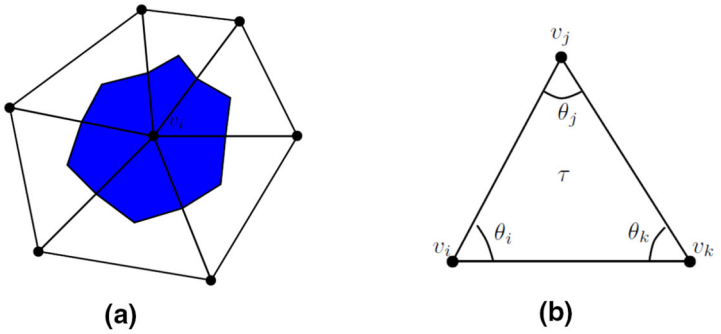

Fig. 4 **a** Dual cell of vertex v_i which plotted in *blue*; **b** Computation of $t_{ij,\tau}, t_{ik,\tau}, t_{jk,\tau}$ of *triangle* τ for (21) (Color figure online)

$$\begin{cases} (C + \Delta t W(\Psi^{(n)}) L(\Psi^{(n)})) \Psi^{(n+1)} = C\Psi^{(n)} \\ \Psi^{(0)} = \Psi_0, \end{cases} \tag{19}$$

where $C = \mathrm{diag}(c_0, c_1, \ldots, c_{\mathrm{V}-1})$,

$$W(\Psi^{(n)}) = \mathrm{diag}\left(\frac{\sum\limits_{\tau \prec M_1(0)} |[\psi^n]_\tau| s_\tau}{\sum\limits_{\tau \prec M_1(0)} s_\tau}, \frac{\sum\limits_{\tau \prec M_1(1)} |[\psi^n]_\tau| s_\tau}{\sum\limits_{\tau \prec M_1(1)} s_\tau}, \ldots, \frac{\sum\limits_{\tau \prec M_1(\mathrm{V}-1)} |[\psi^n]_\tau| s_\tau}{\sum\limits_{\tau \prec M_1(\mathrm{V}-1)} s_\tau} \right)$$

are two diagonal matrices. The calculation of $[\psi]_\tau$ is referred to the article [42]. The matrix $L(\Psi^{(n)}) = l_{ij}$ in (19) is

$$l_{ij} = \begin{cases} \sum\limits_{\tau \prec M_1(i)} \dfrac{w_\tau}{\sqrt{|[\psi^n]_\tau|^2 + \beta}} t_{ii,\tau}, & j = i, \\ \sum\limits_{\tau, [v_i, v_j] \prec \tau} \dfrac{w_\tau}{\sqrt{|[\psi^n]_\tau|^2 + \beta}} t_{ij,\tau}, & j \in N_1(i) \\ 0, & \text{others}, \end{cases} \tag{20}$$

where

$$\begin{cases} t_{ii,\tau} = -\frac{1}{2}\cot\theta_k - \frac{1}{2}\cot\theta_j, \\ t_{ij,\tau} = \frac{1}{2}\cot\theta_k, \\ t_{ik,\tau} = \frac{1}{2}\cot\theta_j, \end{cases} \tag{21}$$

as shown in [42]; see also Fig. 4b.

For clearly, we list differences between our discretization and that in [42]:

(1) Our discretization can make the evolving curve exactly distribute on edges of the mesh, while that in [42] cannot. Our discretization is suitable for surface segmentation application, which prevents generating ambiguous triangular strips (see Fig. 1).
(2) Our discretization has better theoretical property. For any meshes, the maximum-minimum principal is hold for ours. For poor quality mesh with obtuse triangles, the maximum-minimum principal is not hold in [42].
(3) Our discretization has simpler formulation. From (19) to (21), we can see that the discreatization in [42] needs to calculate the trigonometric function and areas of dual cells. In contrast, our discretization is simple in calculation; see our discretization in Sect. 3.3.
(4) Our discretization has more sparse coefficient matrix. For example, as we can see in Fig. 3, the number of nonzero elements in the ith row of the coefficient matrix (11) is 3

(see Fig. 3a), while that of the coefficient matrix (20) is 6 (see Fig. 3b, c). However, the dimension of the system (12) of our discretization is larger than the system (19) in [42], because the number of triangles of the mesh is usually larger than that of vertices.

3.6 A Discussion on the Limit Behavior

The limit behavior of geodesic curvature flow depends on the geometry of the manifold and the initial flow function. We observe from lots of experiments that, the flow function tends to be piecewise constant after enough number of iterations. Right now we cannot directly prove the convergence of the flow function. However, if it converges, i.e., $\Phi^{(n)} \to \Phi^{(\star)}$ as $n \to \infty$, we get, from (12),

$$(G(\Phi^{(\star)})H(\Phi^{(\star)}))\Phi^{(\star)} = 0, \tag{22}$$

where $G(\Phi^{(\star)})$ is a diagonal matrix and $H(\Phi^{(\star)})$ is a symmetric and positive semidefinite matrix described in (11). The system (22) indicates either $g_{ii} = 0$ or $\sum_j h_{ij}\phi_j^\star = 0, \forall i$. If $g_{ii} = 0$, we have

$$g_{ii} = \frac{\sum\limits_{e \prec \tau_i} |[\phi^\star]_e| l_e}{\sum\limits_{e \prec \tau_i} l_e} = 0,$$

which means $\phi^\star|_{\tau_j} = \phi^\star|_{\tau_i} = \phi_j^\star = \phi_i^\star, \forall \tau_j \in D_1(i)$. If $\sum_j h_{ij}\phi_j^\star = 0$, by using the divergence operator (7), we have

$$\begin{aligned}\sum_j h_{ij}\phi_j^\star &= \sum_{\substack{e=\tau_i \bigcap \tau_k, \\ k \in D_1(i)}} \frac{w_e l_e}{\sqrt{|[\phi^\star]_e|^2 + \beta}}(\phi_i^\star - \phi_k^\star) \\ &= -s_{\tau_i}\left(\mathrm{div}(w\frac{\nabla\phi^\star}{\sqrt{|\nabla\phi^\star|^2+\beta}})\right)|_{\tau_i} = 0,\end{aligned}$$

which gives $\left(\mathrm{div}(w\frac{\nabla\phi^\star}{\sqrt{|\nabla\phi^\star|^2+\beta}})\right)|_{\tau_i} = 0$. Note that $\mathrm{div}(w\frac{\nabla\phi^\star}{\sqrt{|\nabla\phi^\star|^2+\beta}})$ is an approximation of the geodesic curvature of the level set, when $w = 1$. A large number of experiments show that, *the limit of geodesic curvature flow would be a piecewise constant solution with discontinuity set to be the closed geodesics of the surface*. This can also be observed from the Eq. (3) in the continuous setting.

As can be seen, in piecewise constant function space, geodesics cannot be directly got from the zero level set of the flow function. Thus, we assume geodesics distribute at the position where the sign of the flow function changes. This idea can be extended to solve multi-phase segmentation problem by a single level set function. The limit behavior observed from experiments provides a possible way to partition the surface to components separated by closed geodesics. We will propose a constrained version in the next section, together with a novel narrow band algorithm to accelerate evolving of the constrained flow function. The narrow band algorithm is designed to approximate the limit behavior with piecewise constant function initialization where function value is a constant in each phase.

4 Constrained Geodesic Curvature Flow with a Narrow Band Algorithm

We now present a constrained geodesic curvature flow with a narrow band algorithm. Although the limit behavior has not been rigorously proved, this constrained flow helps in practice to implement multi-region surface segmentation with a single level set function. It will be described in the discrete setting.

4.1 Constrained Discrete Geodesic Curvature Flow

Assume that some particular triangles are specified and required to keep the flow function values unchanged at these triangles during the curve evolution. According to the limit of the flow function, the final flow function would be a piecewise constant flow function whose discontinuity set contains some closed geodesics. It is natural that the unchanged flow function values can be used as the labels of the subregions of the surface. For this purpose, we divide all the triangle indices into two sets K_1 and K_2. K_1 is the set of indices encapsulating the curve evolution, while K_2 is the set of indices where the flow function values are constrained to be fixed during the evolution. When $\tau_i \in K_1$, ϕ_i^{n+1} is updated by the conventional level set method. On the contrary, for $\tau_i \in K_2$, we set $\phi_i^{n+1} = \phi_i^n$.

Therefore, we come to following matrix form:

$$\begin{cases} (S + \Delta t G_c(\Phi^{(n)}) H(\Phi^{(n)})) \Phi^{(n+1)} = S\Phi^{(n)} \\ \Phi^{(0)} = \Phi_0, \end{cases} \tag{23}$$

where $S = \mathrm{diag}(s_{\tau_0}, s_{\tau_1}, \ldots, s_{\tau_{\mathrm{T}-1}})$, $H(\Phi^{(n)})$ is the matrix introduced in (11), and $G_c(\Phi^{(n)}) = \mathrm{diag}(g_0, g_1, \ldots, g_{\mathrm{T}-1})$ is also a diagonal matrix with

$$g_i = \begin{cases} \dfrac{\sum\limits_{e \prec \tau_i} |[\phi^n]_e| l_e}{\sum\limits_{e \prec \tau_i} l_e}, & \tau_i \in K_1 \\ 0, & \tau_i \in K_2. \end{cases} \tag{24}$$

Let Φ_0 be an initial flow function. With a fixed time step Δt and $t^0 = 0$, the sequence $\{\Phi^{(n)}, n = 0, 1, 2, \ldots\}$ determined by (23) is called as *constrained discrete geodesic curvature flow* of the initial flow function Φ_0.

Remark In real computation, (23) can be reformulated into a linear system $Ax = b$ with much smaller dimension, where x is a vector whose entries refer to triangle indices in K_1. The reason is that, there is no need to calculate the equations of (23) whose indices contained in K_2. Thus, we can remove these equations from (23), which dramatically reduces the computation cost.

4.2 Analysis of the Constrained Discrete Geodesic Curvature Flow

In this subsection, we present some fundamental theoretical aspects of *constrained discrete geodesic curvature flow*, which include the existence and uniqueness, regularization behavior and maximum-minimum principal.

Theorem 4 (existence and uniqueness) *For any initial value Φ_0, with a fixed time step Δt and $t^0 = 0$, there exists a unique constrained discrete geodesic curvature flow $\{\Phi^{(n)}, n = 0, 1, 2, \ldots\}$ determined by* (23).

Proof We only need to prove the coefficient matrix of the system (23) is invertible. We compute the determinant of $(S + \Delta t G_c(\Phi^{(n)})H(\Phi^{(n)}))$ and have

$$\begin{aligned}
&\det(S + \Delta t G_c(\Phi^{(n)})H(\Phi^{(n)})) \\
&\quad = \det(S + \Delta t \ \mathrm{diag}(g_0, g_1, \ldots, g_{\mathrm{T}-1})H(\Phi^{(n)}) \\
&\quad = \det(S) \cdot \det\left(I + \Delta t \ \mathrm{diag}\left(\frac{g_0}{s_0}, \frac{g_1}{s_1}, \ldots, \frac{g_{\mathrm{T}-1}}{s_{\mathrm{T}-1}}\right) H(\Phi^{(n)})\right) \\
&\quad := \det(S) \cdot \det(I + \Lambda H) \\
&\quad = \det(S) \cdot \left(1 + \sum_{1 \le k \le \mathrm{T}} \sum_{i_1 < i_2 < \ldots < i_k} \Lambda H \begin{pmatrix} i_1 & i_2 & \cdots & i_k \\ i_1 & i_2 & \ldots & i_k \end{pmatrix}\right) \\
&\quad > 0,
\end{aligned}$$

by Lemmas 1, 2 and 3. This proves the invertibility of the coefficient matrix of the system (23). □

Theorem 5 (regularization behavior) *For the constrained flow function $\Phi^{(n)}$, we assume that $L_1 = \{i | \phi_j^n = \phi_i^n, \ j \in D_1(i)\}$, $L_2 = K_2$ (the index set for constrained triangles) and $L_3 = \{0, 1, \ldots, \mathrm{T}-1\} \backslash (L_1 \cup L_2)$ are three index sets. Then, $(I + \Delta t S^{-1} G_c(\Phi^{(n)})H(\Phi^{(n)}))^{-1}$ has eigenvalues $0 < \mu_0, \mu_1, \ldots, \mu_{\mathrm{T}-1} \le 1$ with corresponding eigenvectors $\{v_i = (v_{i,0}, v_{i,1}, \ldots, v_{i,T-1})', i = 0, 1, \cdots, T-1\}$, which is complete. Moreover, for all $i \in L_1 \cup L_2$, $\mu_i = 1$ is the eigenvalue with $(1, 1, \ldots, 1)$ as one of the corresponding eigenvectors; for $i \in L_3, 0 < \mu_i < 1$ and $v_{ij} = 0, j \in L_1 \cup L_2$ in its corresponding eigenvector v_i.*

Proof Let

$$D = S^{-1} G_c(\Phi^{(n)}),$$

which is a diagonal matrix with nonnegative elements.

Since L_2 is nonempty, then G_c is not invertible. For convenience, we use a series of permutations to relocate the zero diagonal elements of G_c together to the later part of the diagonal line. Denoting the element number of $L_1 \cup L_2$ as $|L_1 \cup L_2| = \mathrm{T} - m$, there exists an orthogonal matrix P such that

$$D = P^{-1} \begin{pmatrix} D_{mm} & 0 \\ 0 & 0 \end{pmatrix} P,$$

where D_{mm} is an $m \times m (m < \mathrm{T})$ diagonal submatrix with positive diagonal elements. Hence,

$$\begin{aligned}
&S^{-1} G_c(\Phi^{(n)})H(\Phi^{(n)}) = DH(\Phi^{(n)}) \\
&\quad = P^{-1} \begin{pmatrix} D_{mm} & 0 \\ 0 & 0 \end{pmatrix} PH(\Phi^{(n)}) \sim \begin{pmatrix} D_{mm} & 0 \\ 0 & 0 \end{pmatrix} PH(\Phi^{(n)})P' \\
&\quad := \begin{pmatrix} D_{mm} & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} J_{mm} & J_{mq} \\ J_{qm} & J_{qq} \end{pmatrix} = \begin{pmatrix} D_{mm} J_{mm} & D_{mm} J_{mq} \\ 0 & 0 \end{pmatrix}
\end{aligned}$$

where $q = \mathrm{T} - m$ and J_{mm} is a symmetric positive definite matrix according to Lemma 3.

We then analyze the matrix $\begin{pmatrix} D_{mm} J_{mm} & D_{mm} J_{mq} \\ 0 & 0 \end{pmatrix}$, one can show that the eigenvalues of $D_{mm} J_{mm}$ are positive. Therefore, we can denote all the eigenvalues of DH by

$\lambda_0, \lambda_1, \ldots, \lambda_{m-1}, \lambda_m = \lambda_{m+1} = \ldots = \lambda_{\mathrm{T}-1} = 0$ with corresponding eigenvectors $e_0, e_1, \ldots, e_{\mathrm{T}-1}$.

With a similar analysis of Theorem 2, we can see the eigenvalues of DH satisfy

$$\begin{cases} \lambda_i = 0, & i \in L_1 \cup L_2, \\ \lambda_i > 0, & i \in L_3. \end{cases}$$

Now, we come back to $(I + \Delta t DH(\Phi^n))^{-1}$. Its eigenvalues $\{\mu_i, i = 0, 1, \ldots, \mathrm{T}-1\}$ satisfy

$$\begin{cases} \mu_i = 1, & i \in L_1 \cup L_2 \\ 0 < \mu_i < 1, & i \in L_3, \end{cases}$$

and the corresponding eigenvectors $\{v_i, i = 0, 1, \ldots, \mathrm{T}-1\}$ satisfy

$$v_i = \begin{cases} (1, 1 \ldots, 1), & i \in L_1 \cup L_2, \\ (v_{i,0}, \ldots, v_{i,k}, \ldots, v_{i,\mathrm{T}-1}) \text{with} \{v_{i,k} = 0, k \in L_1 \cup L_2\}, & i \in L_3. \end{cases}$$

Therefore, the assertion follows. □

Theorem 6 (maximum-minimum principle) *Let $\Phi^{(0)}$ be an initial value and*

$$m(\Phi^{(0)}) := \min_{j=0,1,\ldots,\mathrm{T}-1} \phi_j^0, \quad M(\Phi^{(0)}) := \max_{j=0,1,\ldots,\mathrm{T}-1} \phi_j^0,$$

then, $m(\Phi^{(0)}) \le \phi_i^n \le M(\Phi^{(0)})$ for $i = 0, 1, \cdots, \mathrm{T}-1, n = 1, 2, \ldots$.

Proof This can be proved in a way similar to Theorem 3. □

4.3 A Narrow Band Algorithm Applied to Multi-phase Segmentation

Narrow band implementation is a basic technique in level set method [1,2,9,45], whose main purpose is to reduce the computational cost. Here we propose a new narrow band algorithm which, when combined with the constrained geodesic curvature flow, is quite effective for multi-phase segmentation. Compared to conventional narrow band algorithms, our algorithm has two differences. First, our algorithm uses narrow bands not only to reduce the computational cost, but also to track the contour (Note that in multi-phase segmentation, contour tracking is a difficult problem). Second, the initialization and re-initialization of our algorithm use piecewise constant function (inspired by the limit of the flow function), instead of the signed distance function (which is not easy to get on irregular grids).

4.3.1 Overview of Narrow Band Algorithm

In this subsection, we present an overview of the narrow band algorithm.

Initialization We begin with inputting N closed curves as the initial contours. Given N contours a region-growing like method is used to separate the entire surface M into S regions, i.e., $M = \cup_{i=1}^{\mathrm{S}} M_i$ with the regions $\{M_i : i = 1, 2, \ldots, \mathrm{S}\}$. After that, we use a piecewise constant level set function to initialize these regions as following

$$\phi = i \text{ in } M_i, \ i = 1, 2, \cdots, \mathrm{S}. \tag{25}$$

This initialization method is different from the conventional method which uses Eikonal equation to get a signed distance function.

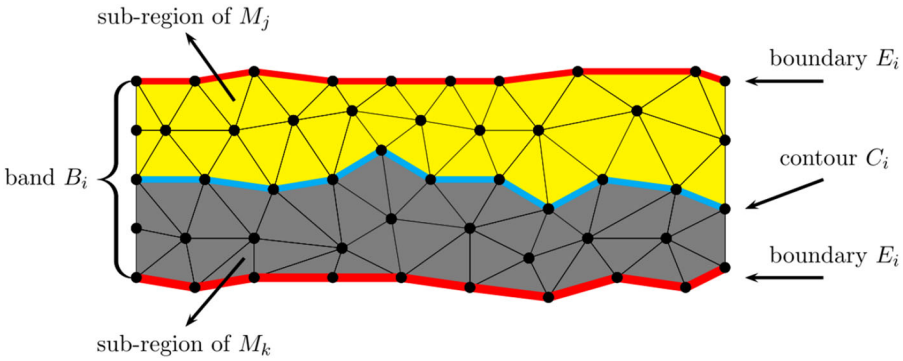

Fig. 5 The illustration of the narrow band data structure. The width of the band is 2. The contour C_i encapsulated in the narrow band B_i is plotted in *blue*. The boundary E_i of the narrow band is plotted in *red*. The two regions contained in the band, which are separated by the contour, are plotted in *yellow* and *gray*, respectively. Assume these two regions to be two sub-regions of M_k and M_j, respectively. The set of the region labels of the band is $R_i = (j, k)$ (Color figure online)

Narrow Band Construction For given contours, we can build corresponding narrow bands. We denote the contours as $\{C_i : i = 0, 1, \ldots, \mathrm{N}\}$, where one contour $C_i = (e_1, e_2, \ldots, e_n)$ is a set of edges. The narrow bands, encapsulating these contours, are denoted as $\{B_i : i = 0, 1, \ldots, \mathrm{N}\}$. We use diagram to illustrate one narrow band B_i in Fig. 5 where the bandwidth is 2.

We can see that the narrow band B_i contains the following information:

1. The set $R_i = (j, k)$ of region labels;
2. The set T_i of triangles contained in the band B_i;
3. The boundary E_i of B_i.

Hence, the narrow band data structure is an array $B_i = \{R_i, T_i, E_i\}$. The construction of this data structure includes three steps, which is detailed in Algorithm 1.

Algorithm 1 Build narrow band data structure

Input: One contour C_i; Narrow band width k.
Output: Narrow band data structure $B_i = \{R_i, T_i, E_i\}$.

1. 1.1 Find region label set $R_i = (j, k)$ satisfying $M_j \bigcap M_k = C_i$ (M_j, M_k are two regions separated by contour C_i);
1.2 Add R_i to B_i;
2. 2.1 Get vertex set V_{C_i} of contour C_i;
2.2 **For all** $v \in V_{C_i}$ **do**
Find k-ring neighbour triangle set $N_k(v)$ of vertex v (see Fig. 6 for $k = 2$);
Add $N_k(v)$ to the set T_i;
2.3 Delete duplicated triangles in T_i;
2.4 Add T_i to B_i;
3. 3.1 Find edge set E_i as the band boundary;
3.2 Add E_i to B_i.

Update of Contour and Segmentation Regions Unlike the conventional level set method where the contour is the zero level set, we regard the contour as the 'discontinuity set' of the

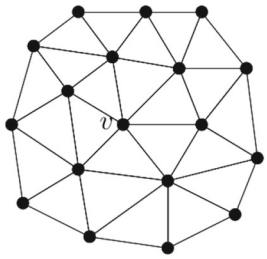

Fig. 6 The illustration of a two-ring neighbour face set of vertex v

level set function. Hence, we devise an algorithm (see Algorithm 2) to find a closed curve consisting of a set of edges with as large $|\nabla\phi|$ as possible. This closed curve is regarded as a contour. After updating the contour, we update the segmentation regions according to the position of the new contour. This algorithm is performed at every specified time steps (i.e., every 10 steps), for two reasons. The first reason is to reduce the computation cost. The second reason is that, more iterations make the level set function more 'stable' to track a valid new contour.

The algorithm consists of two steps. In the first step, we start from an edge with the largest $|\nabla\phi|$ to find a closed-curve whose edges have as large $|\nabla\phi|$ as possible. In the second step, we verify the validity of the new contour updated in step one by the length change. The length of the new contour should be less than the old one. Also, the decreasing speed of the contour length should be not too large. If the decreasing speed is too large, it manifests that the variation of the level set function is strong and the function is not 'stable' enough. We then take this case as an invalid update. If the new contour is a valid one we update the segmentation regions according to the position of the contour, otherwise we keep the contour unchanged. The details can be found in Algorithm 2.

Remark In most cases, the first step of Algorithm 2 can find a closed-curve by choosing an appropriate time step. If the length of the current curve is more than one and a half times of the old one, it is failed to find a closed-curve as the new contour. However, such case is rare. If it happens, we just need to keep the old contour unchanged. Although we cannot provide a theoretical guarantee, the second step of Algorithm 2, the verification of the validity of the new contour by the length change rate, can prevent producing the false contours in all of our examples.

Intersection Detection, Re-initialization and Reconstruction As we track the evolving contour in the narrow band, we must ensure the contour to stay within the band. Similar to [1,2], we do not rebuild a new band around the contour at each time step, because it is a time-consuming process. Instead, we use a band for as many iterations as possible until the contour intersects with the boundary of the band. In addition, when there is an intersection, we need to re-initialize the level set function and update the narrow band.

The intersection detection method is simple. We just need to verify whether there are duplicated vertices between the contour C_i and the boundary E_i of the band B_i. If there is an intersection, we use the region label set $R_i = (j, k)$ of the band B_i to re-initialize the level set function within the band B_i according to the position of C_i. Here, our re-initialization method boosts the constrained flow function to be a valid approximation of the limit behavior solution. After the re-initialization we directly apply Algorithm 1 to rebuild a new band from the contour C_i.

Algorithm 2 Update contour and segmentation regions

Input: One narrow band B_i; Evolved level set function ϕ.
Output: One contour C_i; Updated segmentation regions M_j and M_k.

$\tilde{C}_i = C_i, C_i = \{\varnothing\}$;

1. Starting from an edge with the largest $|\nabla\phi|$, find a closed-curve whose edges have as large $|\nabla\phi|$ as possible; and assign the curve to contour C_i

1.1 Find edge e_0 with the largest $|\nabla\phi|$ in band region B_i;
Get two end points $v_{start} \prec e_0$, $v_{end} \prec e_0$ of the edge e_0;
$C_i \leftarrow e_0$;

1.2 **Repeat**
Find next edge e_i has as large $|\nabla\phi|$ as possible which satisfies: $v_{start} \prec e_i$ and $e_i \notin C_i$;
$C_i \leftarrow e_i$;
Reset $v_{start} = v$, where $v \prec e_i$ and $v \neq v_{start}$;
Until$(v_{end} = v_{start})$

2. Check if C_i is a valid new contour; and update segmentation regions if necessary

2.1 Compute curve lengths $l_{C_i}, l_{\tilde{C}_i}$ of $C_i, \tilde{C}_i$, respectively;

2.2 **If** $l_{C_i} < l_{\tilde{C}_i}$ and $l_{C_i} > 0.75 l_{\tilde{C}_i}$ (C_i is valid)
Update segmentation regions M_j and M_k according to the position of C_i;
Else (C_i is invalid)
Use old contour ($C_i = \tilde{C}_i$);

Stopping Criterion We use the following two stopping criteria:

$$\sum_{i=1}^{\mathrm{N}} |l_i^n - l_i^{n-1}| = 0, \tag{26}$$

where l_i is the length of the contour C_i and N is the number of the contours; and

$$\frac{\sum_{i=0}^{\mathrm{T}-1} s_{\tau_i} |\phi_i^n - \phi_i^{n-1}|^2}{\sum_{i=0}^{\mathrm{T}-1} s_{\tau_i}} < \delta, \tag{27}$$

where δ is an accuracy threshold. In this paper, we set $\delta = 10^{-4}$. The stop criterion (26) verifies whether all lengths of the contours keep unchanged. The stop criterion (27) measures the change of the flow function.

4.3.2 The Narrow Band Algorithm

In summary, our narrow band algorithm is given in Algorithm 3, where the details of each sub-algorithm can be found in Sect. 4.3.1.

Remark For the consideration of computational efficiency, our narrow band algorithm is designed for curve evolution with fixed topology. We mention that, in multi-phase surface segmentation, this is enough in most cases. The reason is that many existing methods (e.g., random walks [26]) and manual inputs can provide initial segmentations with correct topology. Most contours of these initializations are even nearby the correct segmentation positions; see the examples in Sect. 5.2. Even with an initialization far away from the correct cutting positions, our method can evolve the contours to correct positions, as long as the topology of initialization is correct.

Algorithm 3 Narrow band algorithm

Input: N initial contours.
Output: N final contours $\{C_i\}_{i=1}^{\mathrm{N}}$; S segmentation regions $\{M_i\}_{i=1}^{\mathrm{S}}$.

1. **Initialization**:
   1.1 Use N initial contours $\{C_i\}_{i=1}^{\mathrm{N}}$ to divide surface M into S segmentation regions $\{M_i\}_{i=1}^{\mathrm{S}}$ satisfying $M = \bigcup_{i=1}^{\mathrm{S}} M_i$;
   1.2 Initialize the level set function ϕ by formulation (25);
   1.3 Build N initial narrow bands $\{B_i\}_{i=1}^{\mathrm{N}}$ from contours $\{C_i\}_{i=1}^{\mathrm{N}}$ by Algorithm 1;
2. **Repeat**
   2.1 Evolve ϕ according to system (23) with a specified time steps;
   2.2 Update contours $\{C_i\}_{i=1}^{\mathrm{N}}$ and segmentation regions $\{M_i\}_{i=1}^{\mathrm{S}}$ in narrow bands $\{B_i\}_{i=1}^{\mathrm{N}}$ by Algorithm 2;
   2.3 **For** i=1 to N **do**
   Detect intersection between C_i and E_i (boundary of B_i);
   If $C_i \bigcap E_i \neq \{\varnothing\}$
   Re-initialize ϕ within the narrow band B_i according to the updated contour C_i and region labels $R_i = (j, k)$;
   Rebuild the narrow band B_i from the updated contour C_i by Algorithm 1;

Until (stop criterions (26), (27) satisfied)

5 Applications and Numerical Examples

We have implemented our discrete and constrained discrete geodesic curvature flows using C++. Two applications will be presented in this section with some numerical examples. We use two quantities as in [42] to verify the robustness of our method to mesh quality. These quantities are defined as following:

$$D^{global} = \frac{\min_\tau \text{ area of } \tau}{\max_\tau \text{ area of } \tau},$$
$$D^{local} = \min_\tau \frac{\min_{e \prec \tau} \text{ length of } e}{\max_{e \prec \tau} \text{ length of } e}.$$

The mesh information of the surfaces used in this paper are listed in Table 1. From Table 1 we can see that, several meshes are very irregular, e.g., the Dumbbell surface. Our method can effectively handle all the surfaces and produce correct results.

5.1 Closed-Curve Evolution Under Geodesic Curvature Flow

In this subsection, we simulate closed-curve evolution under geodesic curvature-dependent velocity. In this case the weight function is set to be $w(\cdot) = 1$. As mentioned in Sect. 1, any closed-curve on a manifold will evolve to disappear or to closed geodesic(s). Some examples are provided, all of which were calculated by the discrete geodesic curvature flow (12). For ease of viewing, we plot the whole closed-curve on the surface. Its front part is plotted in blue while the occluded part is plotted in black.

The example in Fig. 7 demonstrates the maximum-minimum principle (Theorem 3) and the stability (Corollary 1) of our discrete geodesic curvature flow. The initial flow function was specified to be a general function with non zero gradients on most edges. In particular, we first constructed a PCA plane from the initial curve, then set the values of the initial flow function to be the signed Euclidean distance of the faces to the plane. These values were

Table 1 Mesh information of surfaces used in this paper

Surface	#Vertices	#Triangles	D^{global}	D^{local}
Bimba	30,002	60,000	0.107037	0.336681
Bunny	34,835	69,666	4.99319e−006	0.0632449
Dumbbell	18,878	37,752	2.63934e−007	0.000506865
Fertility	19,994	40,000	0.00831098	0.188908
Torus	3200	6400	0.256282	0.471074
Teddy	13,826	27,648	0.0803282	0.402108
Cup	15,006	30,008	0.000435722	0.022333
Horse	19,851	39,698	0.000429362	0.0372104
Child	26,798	53,592	0.102707	0.484837
Lamp	26,408	52,812	0.00929557	0.021453
Buste	25,467	50,930	0.0785729	0.314679
Momento	26,277	52,550	0.00564995	0.0945791
Gargoyle	25,002	50,000	0.000194802	0.0814815
Elk	24,013	48,026	0.00446882	0.143718

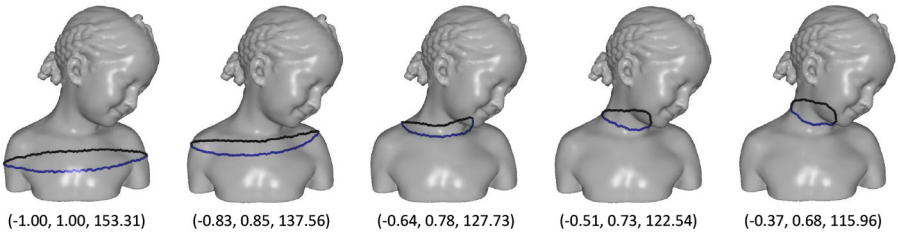

Fig. 7 Closed-curve evolution on the Bimba surface. The curve evolves from the body of the Bimba surface to the neck. The front part of the *curve* is plotted in *blue* while the occluded part is in *black*. From *left* to *right*, the timestamps are 0, 20, 40, 60, 110, respectively. The *numbers in the brackets* denote the minimum, maximum, and the ℓ_2 norm of the current flow function (Color figure online)

normalized within $[-1, 1]$. With a fixed time step $\Delta t = 0.5$, the curve evolves gradually from left to right in Fig. 7. The numbers in the brackets demonstrate the maximum-minimum principle and the stability of our discrete flow.

In Fig. 8, we compare curve evolution procedures by different initial flow functions, which can be interpreted by the regularization behavior (Theorem 2). We started from the same curve on the Bunny surface, but the initial flow functions were different. In the top row of Fig. 8, the initial flow function was set to $\phi = -1$ at one side of the curve and $\phi = 1$ at the other side; while in the bottom row of Fig. 8, we used an initialization by the method in Fig. 7. With a same time step $\Delta t = 0.5$, we recorded the evolution at several timestamps. As can be seen, both initial flow functions give similar evolution procedures. The input curve firstly shrinks and breaks into two parts. Finally one part on the back disappears while the other evolves to a closed geodesic around the ear. However, the evolution speeds are quite different. The evolution speed in the bottom row is more quickly than the speed in the top row, especially at the beginning. The reason of this phenomenon is the regularization behavior of the flow. The piecewise constant components of the flow function keep fixed while the high frequency components supported in transition domains shrink quickly. The flow function in the top

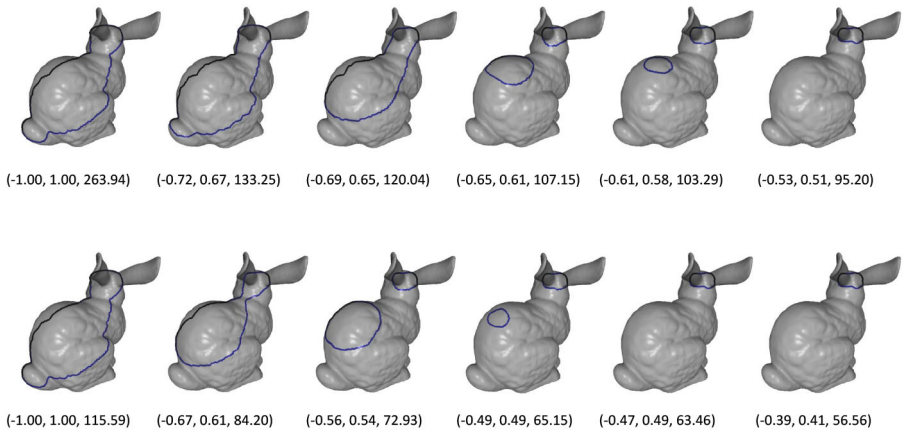

Fig. 8 Closed-curve evolution on the Bunny surface with different initial flow functions. In the first row, the initial flow function is set to $\phi = -1$ at one side of the initial curve and $\phi = 1$ at the other side. In the second row, the initial flow function is a general function constructed by the method in Fig. 7. The curve evolves in the second row more quickly than that in the first row, especially at the beginning. From *left* to *right*, the timestamps are 0, 30, 50, 70, 90, 120, respectively. The *numbers in the brackets* denote the minimum, maximum, and the ℓ_2 norm of the current flow function

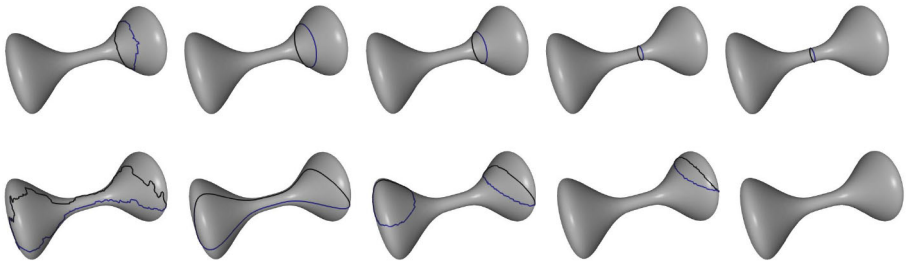

Fig. 9 Closed-curve evolution on the Dumbbell surface. In the *top* row the curve evolves progressively to a stable geodesic. In the *bottom* row the curve evolves firstly to an unstable geodesic which splits the Dumbbell surface to two halves, then shrinks and breaks into two curves, and finally the two curves both disappear

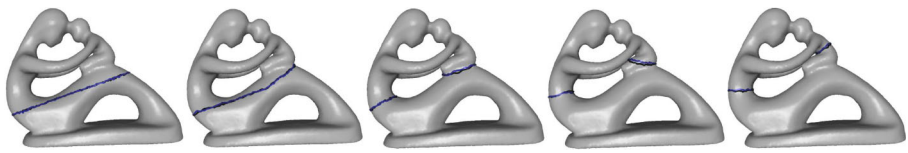

Fig. 10 Closed-curve evolution on the Fertility surface. The curve firstly shrinks, then it breaks into two curves. Finally the two curves evolve to be two stable geodesics, respectively

row has much larger constant components than that in the bottom. The maximum-minimum principle and the stability can also be observed in Fig. 8.

Figure 9 shows examples about stable and unstable closed geodesic(s) [31] on the Dumbbell surface. In the first row of Fig. 9, the curve evolves to a stable geodesic. In contrast, in the bottom row of Fig. 9 the curve evolves firstly to an unstable geodesic, and finally disappears. At last we show an example about curve evolution on a high-genus surface in Fig. 10.

Table 2 The accuracy test of the geodesic curvature flow for a circle moving on a unit sphere

The time step is 0.01 and the number of the iterations is 50

Mesh size (#triangles)	ℓ_2 Error	Order
2500	0.014076	–
10,000	0.012343	0.188803
40,000	0.011353	0.120303
160,000	0.009377	0.275844

5.2 Accuracy Test

In this subsection, we test the accuracy of our discretization. It is very hard to get analytical solution for geodesic curvature flow on general surfaces. We thus consider a circle moving from the initial position of $z = 0.5$ on the unit sphere by geodesic curvature flow. At $t = 0$ the circle is the zero level set of $\phi(x, y, z; 0) = 0.5 - z$, $(x, y, z) \in S^2$. By symmetry and basic calculations, we reformulate the geodesic curvature flow (3) as follows

$$\begin{cases} \frac{\partial \phi(x,y,z;t)}{\partial t} + z\frac{\partial \phi(x,y,z;t)}{\partial z} = 0, \\ \phi(x, y, z; 0) \qquad\qquad\quad = 0.5 - z, \end{cases} \tag{28}$$

where $(x, y, z) \in S^2$.

The analytic solution of the above equation reads

$$\phi(x, y, z; t) = 0.5 - ze^{-t}.$$

In the test, we sampled the unit sphere at several resolutions and compute the numerical solution and analytical solution. The ℓ_2 errors between the numerical and analytical solutions, as well as the accuracy orders, are recorded in Table 2. As expected, the accuracy order is low. However, the flow can still be applied in graphic applications, since a mesh is usually given and the resolution is fixed.

5.3 Multi-phase Segmentation by a Single Level Set Function

In this subsection, we discuss multi-phase surface segmentation, by using the constrained geodesic curvature flow coupled with the narrow band algorithm. Our constrained discrete flow is constructed in piecewise constant function space. It directly partitions a surface mesh into sub-meshes, avoiding the postprocessing step required by existing discrete flows in piecewise linear function space; see Fig. 1 for an example. More importantly, our constrained flow with narrow band algorithm provides a multi-phase segmentation by a single level set function. The conventional level set framework, where the zero level set is used for partitioning, is well suited for two-phase segmentation. While working for many image segmentation and simple surface segmentation applications, conventional level set method has some difficulty to partition high-genus surfaces. See Fig. 2 for an example. If to segment the Torus surface to three parts, it is impossible to represent the cutting contours by the zero level set of a function. Therefore, we propose to use our constrained flow and narrow band algorithm to solve this problem. Our method can generate correct segmentation result (see Fig. 2b).

5.3.1 The Weight Function $w(\cdot)$

The weight function $w(\cdot)$ plays an important role in driving the contour evolution. In active contour models [5,18,22,29] for planar image segmentation, $w(\cdot)$ depends on the intensity

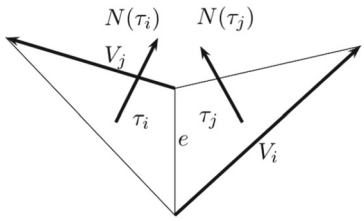

Fig. 11 The method to verify whether an edge e is a *concave edge*

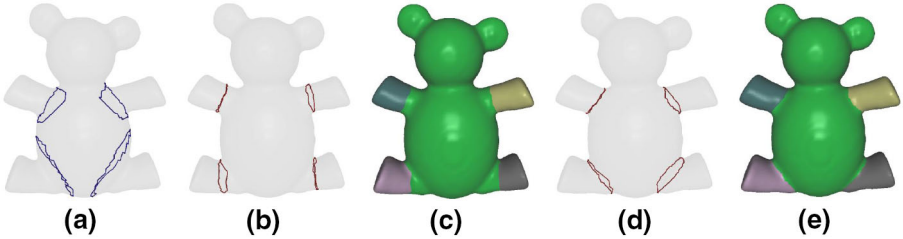

Fig. 12 The segmentation results with different weight functions on the Teddy surface. **a** The *blue* initial contours; **b** the *red* final contours calculated with the weight function $w = 1$; **c** the segmentation result by (**b**); **d** the *red* final contours calculated with weight function $w(\cdot)$ by (29); **e** the segmentation result by (**d**) which is with more human perception (Color figure online)

gradient of the image. On a triangulated surface, if $w(\cdot)$ is simply chosen as a constant function, e.g., $w = 1$, the final cutting contours would be locally shortest under the Euclidean metric. However, according to the minima rule [21], human perception tends to divide a surface into parts along minimum negative curvatures. A usual way is to require the weight function to be monotonically decreasing with respect to the absolute normal difference [26, 44]. In this paper, we set $w(\cdot)$ as follows:

$$w|_e = \frac{1}{1 + \lambda_{ij}\|N(\tau_i) - N(\tau_j)\|^2}, \quad \forall e, \tag{29}$$

where $N(\tau_i)$, $N(\tau_j)$ respectively denote the normal vectors of triangles τ_i and τ_j. The triangles τ_i, τ_j share the common edge e and λ_{ij} is a scaling factor. The scaling factor λ_{ij} is chosen by the following criterion. If the common edge e is a concave edge, $\lambda_{ij} = 5$, otherwise, $\lambda_{ij} = 1$. The method to verify whether the edge e is a concave edge is illustrated in Fig. 11. If $(N(\tau_i) \cdot V_i) > 0$ and $(N(\tau_j) \cdot V_j) > 0$, the edge e is a concave edge; otherwise, e is a convex edge.

The effect of the weight function $w(\cdot)$ is shown in Fig. 12. The flow with $w = 1$ generates final contours with local minimal lengths under Euclidean metric; see Fig. 12b. In contrast, with $w(\cdot)$ calculated by (29), we obtain the final contours as in Fig. 12d, which fit more human shape perception. In this paper, we set the weight function $w(\cdot)$ by (29) for all segmentation examples.

5.3.2 Initialization for Multi-phase Segmentation

Like conventional level set methods, our constrained flow and narrow band algorithm need an initialization. As the flow is a gradient descent method of the non-convex curve energy, it is to require the initial contours nearby the desired cutting positions. Fortunately, this kind of initial contours can be generated by simple manual inputs or many existing automatic

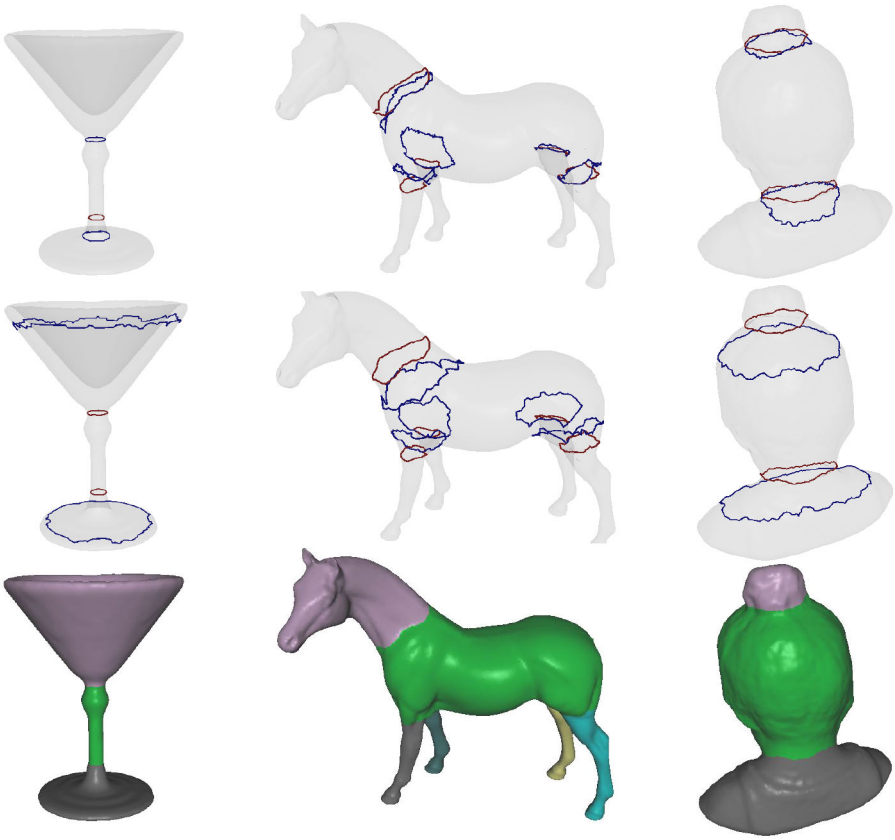

Fig. 13 The segmentation results with initial contours produced by random walks algorithm [26] and manual inputs. In the first and second row, the initial contours are plotted in *blue* and the final contour are plotted in *red*. In the first row, the initial contours are produced by rand walks algorithm [26]. In the second row, the initial contours are produced by manual inputs. The segmentation results are on the bottom row (Color figure online)

algorithms such as the random walks algorithm [26]. They can provide topologically correct initializations. For example, the initial contours in Fig. 12a are produced by rough manual inputs. Our constrained flow and narrow band algorithm evolve these contours and produce final cutting contours in Fig. 12b, d. Figure 13 shows some examples, where the initial contours were generated by random walks algorithm [26] and manual inputs respectively. As we can see in the first row of Fig. 13, our constrained flow and narrow band algorithm can be regarded as an effective refining method of existing methods (e.g., random walks [26]). As shown in the first row of Fig. 13, our method can efficiently drive the contours to the desired cutting positions, although the initial contours generated by manual inputs are far from the desired cutting positions.

5.3.3 Results of Multi-phase Segmentation

In addition to Figs. 2 and 13, we provide some experimental results to show that our method can produce desired multi-phase segmentations on both simple and complex surfaces. We

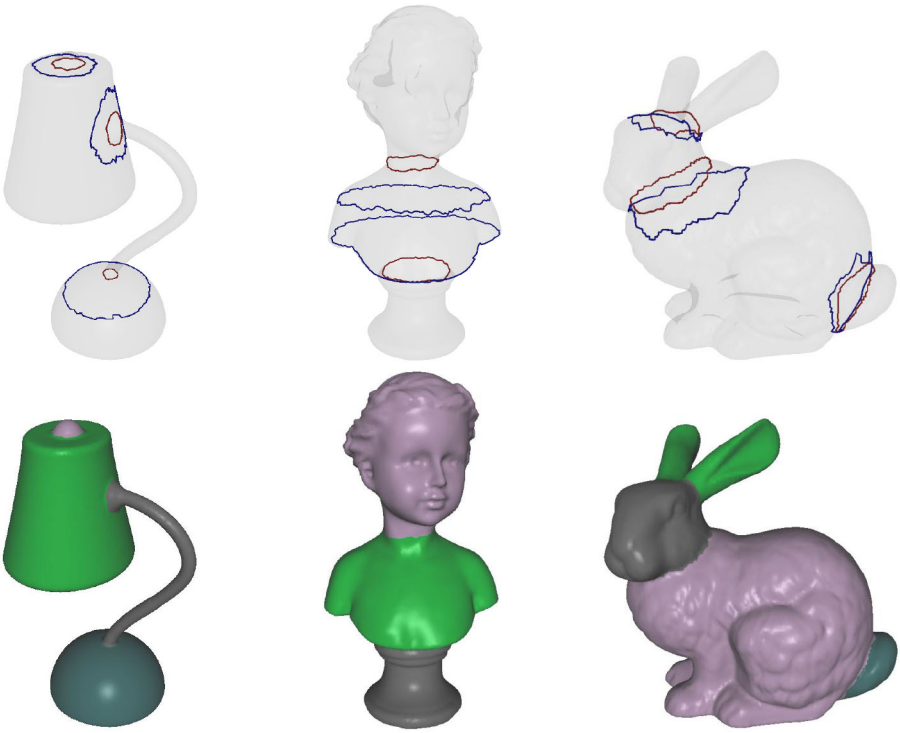

Fig. 14 Segmentation results on more surfaces. The *top* row shows the initial contours plotted in *blue* and the final contours in *red*. The segmentation results are on the *bottom* row (Color figure online)

have tested our method on over thirty different surfaces and illustrated a part of the results in Figs. 14 and 15. It can be seen that, in all the results, the final contours are smooth and along geometric edges, due to their minimal weighted curve lengths. Such smooth and geometry-aware properties match the human perception well. Our constrained flow and narrow band algorithm implement multi-phase surface segmentation with a single level set function.

5.4 Remarks on the Implementation

We have implemented our method using C++ on a notebook with Intel dual core 2.10/2.10 GHZ processor and 4GB RAM. The implementation of our method consists of two parts. The first part is to solve the single-step evolution iteratively, by using the linear system (12) and (23), which is the most time-consuming part of our method. We use Intel **MKL**® package to solve the two nonsymmetric systems (12) and (23) by computing a Cholesky decomposition. The matrices $G(\Phi^{(n)})$, $H(\Phi^{(n)})$ in (12) and matrices $G_c(\Phi^{(n)})$, $H(\Phi^{(n)})$ in (23) are updated dynamically in each iteration, according to the absolute gradient of the flow function Φ. The second part is the narrow band algorithm described in Sect. 4.3, which includes several sub-algorithms. For example, the data structures of narrow bands are builded by Algorithm 1; the tracking of the new contours in narrow bands is Algorithm 2, etc. We mention that the efficiency of our method depends on the geometry of the surface, as well as the initializations of the curve and the flow function. If the initial curve is nearby a geodesic, the flow 'converges' quickly; otherwise, it need more iterations.

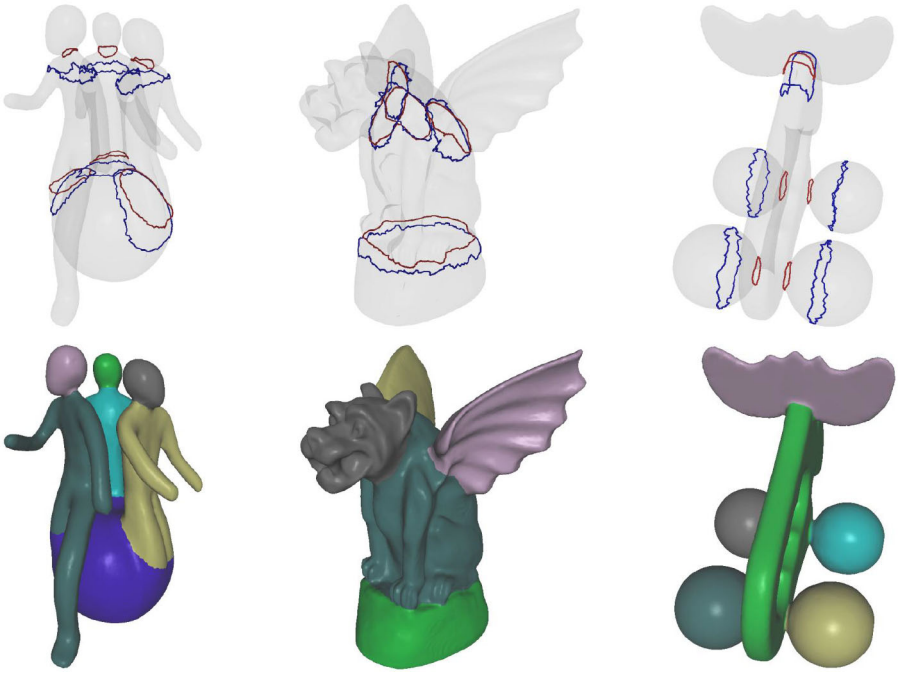

Fig. 15 Segmentation results on more complex surfaces. The *top* row shows the initial contours plotted in *blue* and the final contours in *red*. The segmentation results are on the *bottom* row (Color figure online)

Table 3 CPU costs for closed-curve evolution examples of our flow

Surface	CPU costs (s)	
	Gradually changing function initialization	Piecewise constant function initialization
Bimba (Fig. 7)	221.921	343.182
Bunny (Fig. 8)	212.915	292.189
Dumbbell (first row of Fig. 9)	93.221	143.216
Fertility (Fig. 10)	101.752	147.658

For each example, we use the same time step $\Delta t = 0.5$ and record two CPU costs produced by two different initial flow functions

The CPU costs of our method for all closed-curve evolution examples in Sect. 5.1 are recorded in Table 3. For each example, we record two computational times of our flow produced by two different initial flow functions. One initial flow function is a gradually changing function constructed by the method in Fig. 7, while the other initial flow function is a piecewise constant function ($\phi = -1$ at one side of the initial closed-curve and $\phi = 1$ at the other side). We set the time step $\Delta t = 0.5$ for all closed-curve evolution examples. Currently, we only design the narrow band algorithm for multi-phase segmentation problem. The initialization of the narrow band algorithm should be with correct topology, which is easily to be got by simple existing method (e.g., random walks [26]) or manual input.

Table 4 CPU costs for multi-phase segmentation examples by narrow band algorithm

Surface	CPU costs (s)
Torus (Fig. 2)	4.235
Teddy (Fig. 12b)	13.689
Teddy (Fig. 12d)	7.4176
Cup (first row of Fig. 13)	1.412
Cup (second row of Fig. 13)	8.702
Horse (first row of Fig. 13)	14.192
Horse (second row of Fig. 13)	18.608
Child (first row of Fig. 13)	2.448
Child (second row of Fig. 13)	13.088
Lamp (Fig. 14)	7.394
Buste (Fig. 14)	26.078
Bunny (Fig. 14)	18.536
Momento (Fig. 15)	8.840
Gargoyle (Fig. 15)	16.616
Elk (Fig. 15)	15.877

The CPU costs of narrow band algorithm for multi-phase segmentation examples in Sect. 5.3 are listed in Table 4. For all multi-phase segmentation examples, the time step and the bandwidth of the narrow band algorithm are set to be 0.5 and 2 respectively. As can be seen, by the narrow band algorithm, the computation costs are dramatically decreased, compared to those listed in Table 3.

6 Conclusion and Future Work

In this paper, we proposed a new numerical method to calculate geodesic curvature flow on triangulated surfaces. The new discretization can directly evolve curve on edges of the mesh and thus avoids the postprocessing step to classify or partition the triangles in undetermined strips. Compared to previous methods, our discretization has several advantages. It has simpler formulation and more sparse coefficient matrix. Not only the existence, uniqueness, and regularization behavior, but also the maximum-minimum principal, have been proved. Therein the maximum-minimum principal has not been presented in all previous approaches. Lots of experiments show that, the limit of the discrete flow is a piecewise constant solution with '*discontinuity set*' to be the closed geodesics of the surface. We therefore proposed a discrete constrained geodesic curvature flow with detailed theoretical analysis. The system of the constrained flow can be equivalently reformulated into a much smaller one. Thus, the computational cost is dramatically reduced. This constrained flow, combined with a new narrow band algorithm, yields multi-phase surface segmentation application with a single level set function. This combination can be considered as an cooperation between Lagrangian framework and Eulerian framework. Our two flows were applied to closed-curve evolution and multi-region surface segmentation, respectively. The numerical experiments demonstrated the effectiveness.

Several problems are left open. First, a rigorous proof of the convergence of the flow function is missing. Second, to design a narrow band algorithm for the case of topology

change is a future work. Third, similar discretizations can be applied to other curvature flows such as Willmore flow.

Acknowledgments This work was supported by the NSF of China (No. 11301289), Fundamental Research Funds for Central Universities [China University of Geosciences (Wuhan)] and Fundamental Research Funds for Central Universities (Nankai University). Chunlin Wu is the corresponding author.

References

1. Adalsteinsson, D., Sethian, J.A.: A fast level set method for propagating interfaces. J. Comput. Phys. **118**(2), 269–277 (1995)
2. Adalsteinsson, D., Sethian, J.A.: The fast construction of extension velocities in level set methods. J. Comput. Phys. **148**(1), 2–22 (1999)
3. Brox, T., Weickert, J.: Level set segmentation with multiple regions. IEEE Trans. Image Process. **15**(10), 3213–3218 (2006)
4. Caselles, V., Catté, F., Coll, F., Dibos, F.: A geometric model for active contours in image processing. Numer. Math. **66**(1), 1–31 (1993)
5. Caselles, V., Kimmel, R., Sapiro, G.: Geodesic active contours. In: Proceedings of the Fifth International Conference on Computer Vision, 1995, pp. 694–699 (1995)
6. Caselles, V., Kimmel, R., Sapiro, G., Sbert, C.: Minimal surfaces: a geometric three dimensional segmentation approach. Numer. Math. **77**(4), 423–451 (1997)
7. Chan, T., Vese, L.: Active contours without edges. IEEE Trans. Image Process. **10**(2), 266–277 (2001)
8. Cheng, L.T., Burchard, P., Merriman, B., Osher, S.: Motion of curves constrained on surfaces using a level-set approach. J. Comput. Phys. **175**(2), 604–644 (2002)
9. Chopp, D.L.: Computing minimal surfaces via level set curvature flow. J. Comput. Phys. **106**(1), 77–91 (1993)
10. Chopp, D.L., Sethian, J.A.: Flow under curvature: singularity formation, minimal surfaces, and geodesics. Exp. Math. **2**(4), 235–255 (1993)
11. Davis, H.T., Thomson, K.T.: Linear Algebra and Linear Operators in Engineering: With Applications in Mathematica. Academic Press, New York (2000)
12. Dubrovina, A., Rosman, G., Kimmel, R.: Multi-region active contours with a single level set function. IEEE Trans. Pattern Anal. Mach. Intell. **37**(8), 1585–1601 (2015)
13. Evans, L.C., Spruck, J.: Motion of level sets by mean curvature: I. J. Differ. Geom. **33**(3), 635–681 (1991)
14. Fu, Z., Jeong, W., Pan, Y., Kirby, R., Whitaker, R.: A fast iterative method for solving the eikonal equation on triangulated surfaces. SIAM J. Sci. Comput. **33**(5), 2468–2488 (2011)
15. Gage, M.: Curve shortening makes convex curves circular. Invent. Math. **76**(2), 357–364 (1984)
16. Gage, M.: Curve shortening on surfaces. Ann. Sci. École Norm. Sup. **23**(2), 229–256 (1990)
17. Gage, M., Hamilton, R.S.: The heat equation shrinking convex plane curves. J. Differ. Geom. **23**(1), 69–96 (1986)
18. Goldenberg, R., Kimmel, R., Rivlin, E., Rudzsky, M.: Fast geodesic active contours. IEEE Trans. Image Process. **10**(10), 1467–1475 (2001)
19. Grayson, M.A.: The heat equation shrinks embedded plane curves to round points. J. Differ. Geom. **26**(2), 285–314 (1987)
20. Grayson, M.A.: Shortening embedded curves. Ann. Math. **129**(1), 71–111 (1989)
21. Hoffman, D.D., Richards, W.A.: Parts of recognition. Cognition **18**, 65–96 (1984)
22. Kass, M., Witkin, A., Terzopoulos, D.: Snakes: active contour models. Int. J. Comput. Vis. **1**(4), 321–331 (1988)
23. Kimia, B.B., Siddiqi, K.: Geometric heat equation and nonlinear diffusion of shapes and images. In: Proceedings of the 1994 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 1994 (CVPR'94.), pp. 113–120 (1994)
24. Kimmel, R.: Intrinsic scale space for images on surfaces: the geodesic curvature flow. In: Romeny, B.H., Florack, L., Koenderink, J., Viergever, M. (eds.) Scale-Space Theory in Computer Vision, pp. 212–223. Springer, Berlin (1997)
25. Lai, R., Shi, Y., Sicotte, N., Toga, A.: Automated corpus callosum extraction via laplace-beltrami nodal parcellation and intrinsic geodesic curvature flows on surfaces. In: Proceedings of the 2011 International Conference on Computer Vision, pp. 2034–2040 (2011)
26. Lai, Y., Hu, S., Martin, R., Rosin, P.: Fast mesh segmentation using random walks. In: Proceedings of the 2008 ACM Symposium on Solid and Physical Modeling, pp. 183–191 (2008)

27. Lie, J., Lysaker, M., Tai, X.C.: A variant of the level set method and applications to image segmentation. Math. Comput. **75**(255), 1155–1174 (2006)
28. Ma, L., Chen, D.: Curve shortening flow in a riemannian manifold. 2003. arXiv:math/0312463
29. Malladi, R., Sethian, J.A., Vemuri, B.C.: Shape modeling with front propagation: a level set approach. IEEE Trans. Pattern Anal. Mach. Intell. **17**(2), 158–175 (1995)
30. Meyer, M., Desbrun, M., Schröder, P., Barr, A.: Discrete differential-geometry operators for triangulated 2-manifolds. In: Hege, H.-C., Polthier, K. (eds.) Visualization and Mathematics III: Mathematics and Visualization, pp. 35–57. Springer, Berlin, Heidelberg (2003)
31. Mikula, K., Sevcovic, D.: Evolution of curves on a surface driven by the geodesic curvature and external force. Appl. Anal. **85**(4), 345–362 (2006)
32. Osher, S., Sethian, J.A.: Fronts propagating with curvature dependent speed: algorithms based on Hamilton–Jacobi formulations. J. Comput. Phys. **79**(1), 12–49 (1988)
33. Poole, G., Boullion, T.: A survey on M-matrices. SIAM Rev. **16**(16), 419–427 (1974)
34. Qian, J., Zhang, Y., Zhao, H.: Fast sweeping methods for eikonal equations on triangular meshes. SIAM J. Numer. Anal. **45**(1), 83–107 (2007)
35. Samson, C., Blanc-Féraud, L., Aubert, G., Zerubia, J.: A level set model for image classification. Int. J. Comput. Vis. **40**(3), 187–197 (2000)
36. Saye, R.I., Sethian, J.A.: The Voronoi implicit interface method for computing multiphase physics. Proc. Natl. Acad. Sci. **108**(49), 19498–19503 (2011)
37. Saye, R.I., Sethian, J.A.: Analysis and applications of the Voronoi implicit interface method. J. Comput. Phys. **231**(18), 6051–6085 (2012)
38. Sethian, J.A.: Level Set Methods and Fast Marching Methods: Evolving Interfaces in Geometry, Fluid Mechanics, Computer Vision and Materials Sciences. Cambridge University Press, Cambridge (1999)
39. Spira, G., Kimmel, R.: Geometric curve flows on parametric manifolds. J. Comput. Phys. **223**(1), 235–249 (2007)
40. Tsai, A., Yezzi, A., Willsky, A.: Curve evolution implementation of the Mumford–Shah functional for image segmentation, denoising, interpolation, and magnification. IEEE Trans. Image Process. **10**(8), 1169–1186 (2001)
41. Vese, L., Chan, T.: A multiphase level set framework for image segmentation using the Mumford and Shah model. Int. J. Comput. Vis. **50**(3), 271–293 (2002)
42. Wu, C., Tai, X.C.: A level set formulation of geodesic curvature flow on simplicial surfaces. IEEE Trans. Vis. Comput. Graph. **16**(4), 647–662 (2010)
43. Zhang, H., Wu, C., Zhang, J., Deng, J.: Variational mesh denoising using total variation and piecewise constant function space. IEEE Trans. Vis. Comput. Graph. **21**(7), 873–886 (2015)
44. Zhang, J., Wu, C., Cai, J., Zheng, J., Tai, X.C.: Mesh snapping: robust interactive mesh cutting using fast geodesic curvature flow. Comput. Graph. Forum **29**(2), 517–526 (2010)
45. Zhao, H., Chan, T., Merriman, B., Osher, S.: A variational level set approach to multiphase motion. J. Comput. Phys. **127**(1), 179–195 (1996)