Journal of Computational Physics 228 (2009) 6333–6348

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

Adaptive variational curve smoothing based on level set method

Yu Wang [a], Songhe Song [b,1], Zhijun Tan [c], Desheng Wang [a,*,2]

[a] Division of Mathematical Sciences, School of Physical and Mathematical Sciences, Nanyang Technological University, Singapore 637371, Singapore
[b] Department of Mathematics and System Science, School of Science, National University of Defense and Technology, Changsha, Hunan 410073, China
[c] Singapore-MIT Alliance, 4 Engineering Drive 3, National University of Singapore, Singapore 117576, Singapore

ARTICLE INFO

Article history:
Received 9 January 2009
Received in revised form 12 April 2009
Accepted 14 May 2009
Available online 20 May 2009

Keywords:
Curve smoothing
Level set method
Adaptive meshing
Finite volume method
Unstructured meshes

ABSTRACT

This paper presents an adaptive method for variational curve smoothing based on level set implementation. A suitable cost functional is minimized via solving the derived Euler–Lagrangian equation, of which the discretization is conducted on unstructured triangular meshes by employing a simple and effective finite volume scheme. Through adaptive refinement of the mesh, the geometry features of the given curve can be well resolved in a cost-effective way. Various numerical experiments demonstrate the effectiveness and efficiency of the proposed approach.

1. Introduction

Curve smoothing is an interesting problem from both practical and theoretical views. It is a necessary pre-processing and post-processing step in addressing various computer vision problems. As an example, following segmentation object boundaries have many zigzags and spurious components, to do further analysis the curve must be smoothed out, which is illustrated in Fig. 1.

Among the various existing curve and surface smoothing methods [1–4], the variational approach is gaining popularity due to its robustness and effectiveness in handling noises. Variational curve smoothing is based on minimizing certain functionals, from which the Euler–Lagrange partial differential equations are derived via calculus of variation and solved by gradient descent flow methods or the recently developed graph-cut methods [3]. The functional proposed in [3] is employed here, which is the combination of a fidelity term measuring the "distance" of a curve from the data and a smoothness term measuring the curve's L_1-norm or length. To measure the distance between the curve and the given data, one option is

$$E[C] = \int_C \phi_{data}(C(s))ds,$$

where $\phi_{data}(x, y)$ is a "distance field", induced by the original noisy curve and C is the curve for which we are looking. This functional is related to the geodesic active contour models in image segmentation [5]. An alternative approach is to measure the distance between curves by the area of "symmetric differences" based on indicator functions, which is also suitable for

* Corresponding author.
E-mail addresses: wang0312@ntu.edu.sg (Y. Wang), shsong31@gmail.com (S. Song), smatz@nus.edu.sg (Z. Tan), desheng@ntu.edu.sg (D. Wang).

[1] The work is supported in part by the NFS of China (10571178).
[2] The work is supported in part by Singapore MOE ARC 29/07 T207B2202, MOE RG 59/08 M52110092 and NRF 2007IDM-IDM 002-010.

0021-9991/$ - see front matter © 2009 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2009.05.021

(a) Image (b) Segmentation (c) Contours

Fig. 1. Noisy boundary from segmentation.

discrete optimization methods such as graph cuts method used in [3]. However, a very fine mesh is required to represent a curve with adequate accuracy, which renders a huge computational burden for PDE-based methods. Instead, in the present paper an implicit level set function is used to represent a curve, which has been widely accepted since [6] and has the advantage of being continuous around the curve and topology-friendly. Thus, the distance between two curves can be measured via the integral of the difference of the signed distance functions:

$$\int_{\Omega} |\phi - \varphi|,$$

where ϕ and φ are the signed distance functions of the two curves. For the smoothness term, the arc-length is a natural choice, which is the total variation of the Heaviside function in level set terminology. The complete form of the energy functional will be given in Section 2.

To minimize the proposed functional, one choice is to apply the graph-cuts based method as in [3], where uniform grids are used to construct the graph. Another option is to solve the corresponding Euler–Lagrange PDE via time-marching methods, mainly on uniform grids. In both cases, the use of uniform grids causes an unaffordable computational cost, in order to capture the geometry irregularity of both the given noisy curve and the final solution curve. Thus applying adaptive triangular meshing is a natural method and is of much necessity. Besides, since multi-scale property is inherent in curve smoothing, using adaptive triangular meshes is also expected to improve the solution. For these purposes, the development of an adaptive level set method motivates this work.

Although efforts have been put to solve the level set equation on quadtree (octree in 3D) meshes [7] and also on triangular meshes in [8], there is no well-developed method on non-uniform triangular meshes for geometric smoothing. On the other hand, finite volume methods (FVM's) [9–11] have been widely used in computational science and engineering due to the nice properties in observing the conservation laws and working well for unstructured meshes. In this work, an easy-to-implement, yet very effective finite volume scheme is developed for the underlying time-dependent PDE. The development of the scheme is based on the assumption that the solution is linear on each triangle so that the gradient is piecewise constant in each triangle. Subsequently, the average nonlinear diffusion term is evaluated via summation of the line integrals around the faces of the control volume. The details of the scheme will be presented in Section 3.

Based on the developed FV level set method, an adaptive procedure for curve smoothing is proposed. The mesh around the newly constructed curve is adaptively refined for iterative solutions of the Euler–Lagrange PDE in FVM until convergence. The mesh refinement bases on sizing modification, normalized edge length computation, local mesh edge splitting and contraction. Also, edge swapping and smoothing are performed for the optimization of the mesh, which enhances the robustness of the FVM. Various numerical examples demonstrate the effectiveness of the proposed adaptive procedure very well.

The remaining part of the paper is organized as follows. Section 2 introduces the functional for curve smoothing problems and briefly reviews the level set implementation on uniform grids. Section 3 discusses the finite volume scheme on

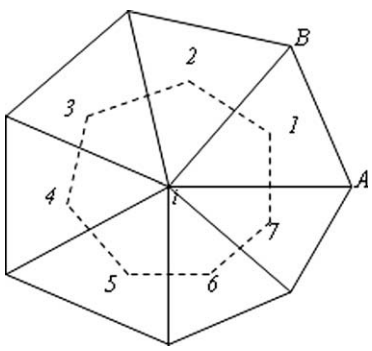
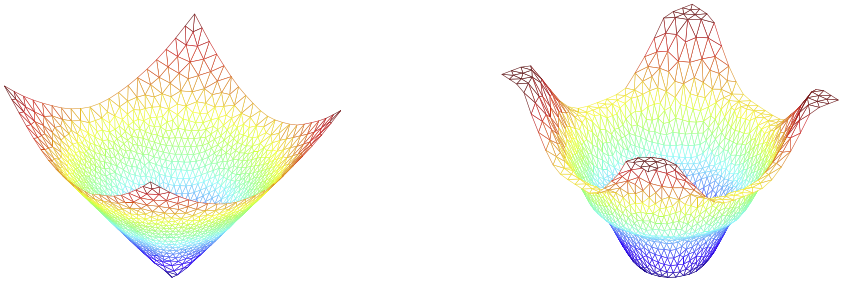
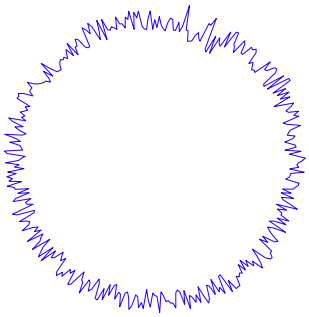


Fig. 2. A example of control volume.

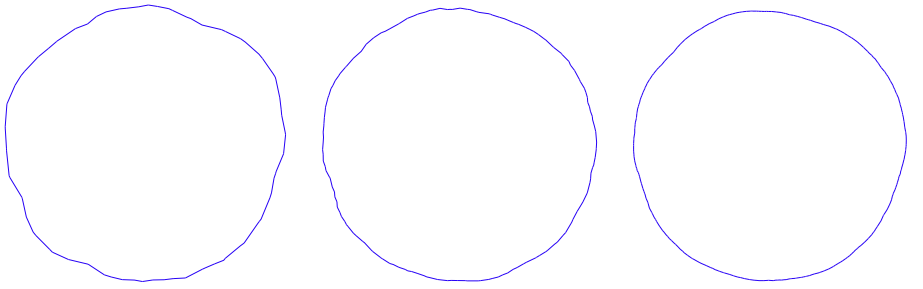


(a) Signed distance function (b) Degenerated signed distance function

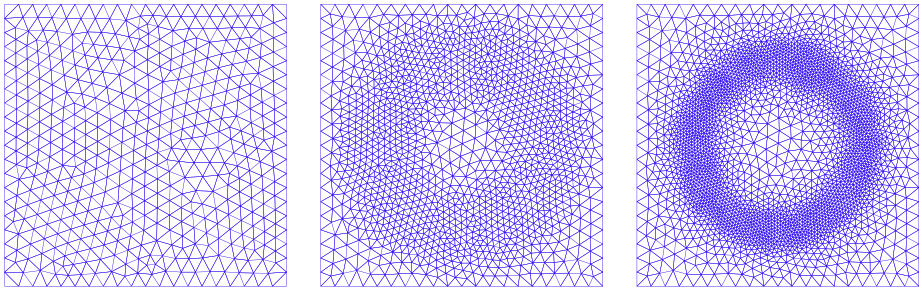
Fig. 3. An example of the evolution of the distance function.



(a) Original noisy circle



(b) Smoothed curve on mesh1 (c) Smoothed curve on mesh2 (d) Smoothed curve on mesh3



(e) mesh1 (f) mesh2 (g) mesh3

Fig. 4. Noisy circle smoothed on adaptive meshes.

unstructured triangular meshes, based on which in Section 4, the adaptive mesh refinement strategy is presented. In Section 5, numerical experiments are included and finally concluding remarks are given in Section 6.

2. Variational curve smoothing

A curve smoothing model usually consists of two basic components, one of which is the similarity measure, and the other is the regularization or smoothness term [12]. And the solution curve is defined as the minimizer of a certain functional, which can be given in an abstract manner as follows:

$$C = argmin(similarity(C, C_0) + regularity(C)),$$

where C_0 is the noisy curve and C is the smooth curve.

If a curve is represented by its signed distance function, one choice for the similarity term can be the integral of the difference of the two distance functions. And a natural choice for the regularity term is the arc-length of the curve. Therefore, the curve smoothing model can be formulated as

$$E_1(\phi, \lambda) = \int |\nabla H(\phi)| + \lambda \int |\phi - \varphi|, \tag{1}$$

where ϕ, φ is the signed distance function of the smooth curve and noisy curve respectively, λ is a given constant parameter and $H(z)$ is the Heaviside function given by

$$H(z) = \begin{cases} 1, & z \geqslant 0, \\ 0, & z < 0. \end{cases}$$

Minimizing functional E_1, the corresponding Euler–Lagrangian equation is obtained as

$$-\delta(\phi) div\left(\frac{\nabla\phi}{|\nabla\phi|}\right) + \lambda\frac{\phi - \varphi}{|\phi - \varphi|} = 0.$$

By applying the gradient descent flow method, the time-marching PDE goes as follows:

$$\frac{\partial\phi}{\partial t} = div\left(\frac{\nabla\phi}{|\nabla\phi|}\right) - \lambda\frac{\phi - \varphi}{|\phi - \varphi|}. \tag{2}$$

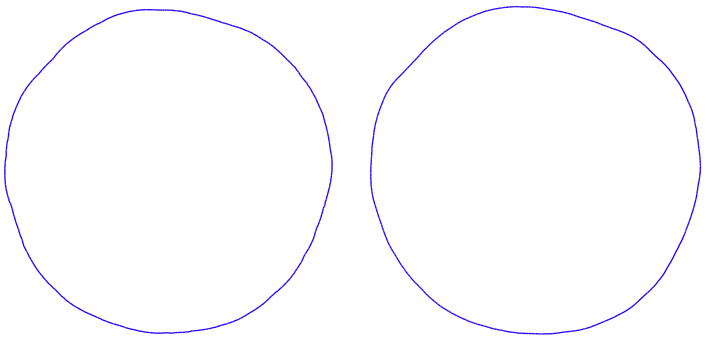
Note that $\delta(\phi)$ is replaced by 1, which was employed in many other works [13] with success.

On uniform quad meshes the evolution equation (2) can be discretized in the following finite difference scheme:

$$\frac{\phi^{n+1} - \phi^n}{dt} = D_x^- F_x^{+,n} + D_y^- F_y^{+,n} - \lambda\frac{\phi^n - \varphi^n}{\sqrt{(\phi^n - \varphi^n)^2 + \varepsilon}}, \tag{3}$$

Table 1
Energy and CPU time on adaptive meshes.

	Node	CPU time (s)	Eng.
Adap. mesh1	503	2.38	4.267
Adap. mesh2	1335	4.70	4.201
Adap. mesh3	3026	7.97	4.155
Adap. mesh4	5293	28.45	4.142



(a) Smoothed curve on adaptive mesh (b) Smoothed curve on uniform mesh

Fig. 5. Smoothed curves on uniform and adaptive mesh.

where

$$D_x^- = \frac{\phi_{i,j} - \phi_{i,j-1}}{dx}, \quad D_x^+ = \frac{\phi_{i,j+1} - \phi_{i,j}}{dx}, \quad D_y^- = \frac{\phi_{i,j} - \phi_{i-1,j}}{dy}, \quad D_y^+ = \frac{\phi_{i+1,j} - \phi_{i,j}}{dy},$$

and

$$F_x^{\pm,n} = \frac{D_x^{\pm}\phi^n}{\sqrt{\left(D_x^{\pm}\phi^n\right)^2 + \left(D_y^{\pm}\phi^n\right)^2 + \varepsilon}}, \quad F_y^{\pm,n} = \frac{D_y^{\pm}\phi^n}{\sqrt{\left(D_x^{\pm}\phi^n\right)^2 + \left(D_y^{\pm}\phi^n\right)^2 + \varepsilon}}.$$

Here ε is a small positive number, which is taken as 10^{-6} in this work.

In order to solve the complicated geometry structures of the given curve, a very fine uniform mesh is required and therefore it causes very high computational costs. Then, the use of unstructured non-uniform triangular meshes is a natural choice. This results in the necessity to develop the level set method on triangular meshes, which will be presented next.

3. Level set method on triangular meshes

In [7] adaptive level set methods were developed on quadtree grids and in [8] Barth and Sethian developed level set methods on triangular meshes, and monotone finite element type numerical schemes were proposed for general Hamilton–acobi equations. In addition, the Petrov–Galerkin approximation was utilized to achieve higher order accuracy. In [9] Hu and Shu designed a higher order WENO finite volume schemes for 2-D conservation laws on triangular meshes, which can be applied to level set equations. As all these schemes stem from conservation laws, the numerical fluxes used in these schemes may not work for the nonlinear diffusion problem in Eq. (2) in a cost-effective manner. In curve smoothing, high order accuracy may not result in a significant difference in the final solution, while in computational fluid dynamics the order of accuracy matters a lot [9]. Therefore, a simpler numerical scheme is expected to work for our curve smoothing problem. For the purpose, a linear finite volume scheme is developed, which is easy-to-implement and yet very effective.

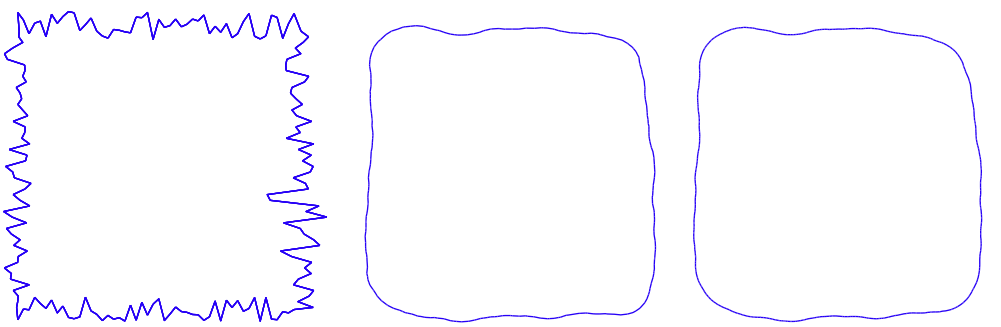
By Gauss theorem, Eq. (2) can be rewritten as

$$\frac{\partial \bar{\phi}}{\partial t} = \frac{1}{|T_i|}\sum_{j\in N(i)}\int_{e_{ij}} \frac{\nabla\phi}{|\nabla\phi|}\cdot\vec{n}\,ds - \frac{\lambda}{|T_i|}\int_{T_i}\frac{\phi-\varphi}{|\phi-\varphi|}d\Omega, \tag{4}$$

where T_i is the control volume which is the polygon formed by connecting the emanating triangle centroids of vertex i shown in Fig. 2, and $\bar{\phi}$ is the average of ϕ on control volume T_i whose volume is represented by $|T_i|$. $N(i)$ is the index set of the adjacent vertices and e_{ij} denotes the edge connecting vertices i and j. $\vec{n}$ denotes the unit outward normal vector.

Table 2
Energy and CPU time on uniform meshes.

Mesh size	Node	CPU time (s)	Eng.
0.05	1919	2.33	4.267
0.025	7462	7.64	4.17
0.0125	29566	76.45	4.15



(a) Noisy rectangle (b) Smoothed curve on $h =$ 0.025 uniform mesh (c) Smoothed curve on $h =$ 0.0125 uniform mesh

Fig. 6. Noisy rectangle smoothed on uniform meshes.

To compute $\nabla\phi$, linear reconstruction via least-square methods is used, which is based on the assumption that $\nabla\phi$ is piecewise constant in each triangle. The reconstruction is of low order and works effectively, although higher order schemes such as WENO [9] could give better numerical accuracy. The latter requires significantly higher computational cost, but hardly makes observable difference on resulting curves. Furthermore, the numerical flux is a subtle issue for higher order reconstruction methods. Thus, for the sake of efficiency, linear reconstruction is preferred here. Without loss of generality, Δ_{iAB} is taken as an example for an illustration of the scheme and $\partial\Delta_{iAB}$ is its boundary. As $\nabla\phi$ is constant in Δ_{iAB}, by Gauss theorem it follows that

$$\begin{aligned}(\nabla\phi)_{\Delta_{iAB}} &\approx \frac{1}{|\Delta_{iAB}|}\int_{\Delta_{iAB}}(\nabla\phi)d\Omega = \frac{1}{|\Delta_{iAB}|}\oint_{\partial\Delta_{iAB}}\phi\cdot\vec{n}\,ds = \frac{1}{|\Delta_{iAB}|}\left(\int_{iA}\phi\cdot\vec{n}_{iA}\,ds + \int_{AB}\phi\cdot\vec{n}_{AB}\,ds + \int_{Bi}\phi\cdot\vec{n}_{Bi}\,ds\right)\\ &= \frac{1}{2|\Delta_{iAB}|}[|iA|(\phi_i+\phi_A)\vec{n}_{iA} + |AB|(\phi_A+\phi_B)\vec{n}_{AB} + |Bi|(\phi_B+\phi_i)\vec{n}_{Bi}]. \end{aligned} \tag{5}$$

Here $\vec{n}_{\{\cdot\}}$ are unit normal vectors and the subscript denotes the corresponding edge in the control volume. Thus, in each triangle a constant $\nabla\phi$ is reconstructed by using nodal values. And the first term of the right hand in (4) can be approximated as follows:

$$\frac{1}{|T_i|}\sum_{j\in N(i)}\int_{e_{ij}}\frac{\nabla\phi}{|\nabla\phi|}\cdot\vec{n}\,ds \approx \frac{1}{|T_i|}\sum_{j\in N(i)}\left[|e_{ij1}|\left(\frac{\nabla\phi}{|\nabla\phi|}\right)_{ij1} + |e_{ij2}|\left(\frac{\nabla\phi}{|\nabla\phi|}\right)_{ij2}\right]\cdot\vec{n}_{ij},$$

where $\left(\frac{\nabla\phi}{|\nabla\phi|}\right)_{ij1}$ and $\left(\frac{\nabla\phi}{|\nabla\phi|}\right)_{ij2}$ are evaluated at the two vertices of the jth face by the formula of

$$\left(\frac{\nabla\phi}{\sqrt{\phi_x^2+\phi_y^2+\epsilon}}\right),$$

and e_{ij1} and e_{ij2} are two segments of the jth face, which are split by a triangle edge. Although there might be jumps at the intersection points (e.g. the intersection of the segments iA and 17 in Fig. 2), the numerical stability is not degraded in our experiments. ϵ is set to be 10^{-6}, which is the same as that on a uniform quad grid. And our numerical experiment results show that the choice is appropriate in terms of numerical stability and solution accuracy. By Taylor expansion, $\phi(x,y)\approx\bar{\phi}_i+\phi_x(x-x_c)+\phi_y(y-y_c)$, which leads to

$$\frac{1}{|T_i|}\int_{T_i}\frac{\phi-\varphi}{|\phi-\varphi|}d\Omega \approx \frac{\phi_i-\varphi_i}{|\phi_i-\varphi_i|+\epsilon}.$$

For the sake of simplicity, the time discretization is conducted in an explicit Euler finite difference scheme

$$\bar{\phi}^{n+1} = \bar{\phi}^n + \left[\frac{1}{|T_i|}\sum_{j\in N(i)}\int_{e_{ij}}\frac{\nabla\phi^n}{|\nabla\phi^n|}\cdot\vec{n}\,ds - \frac{\lambda}{|T_i|}\int_{T_i}\frac{\phi^n-\varphi}{|\phi^n-\varphi|}d\Omega\right]\Delta t,$$

and Neumann boundary conditions are enforced by adding ghost cells. The initial condition ϕ^0 is taken to be the distance function generated by the given noisy curve φ.

One important aspect of the above numerical approach is that no numerical flux is defined across control volumes, due to the fact that the gradient is continuous between any two neighboring control volumes. This is because that the common face of two neighboring control volumes lies in the same triangle, whose nodal values are used for the reconstruction of gradients.

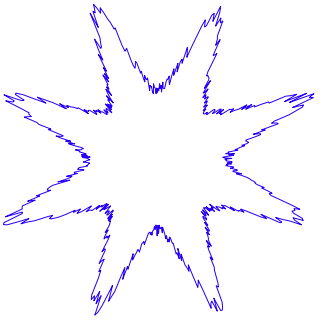
Table 3
Energy and CPU time on uniform meshes.

Mesh size	Node	CPU time (s)	Eng.
0.05	1919	3.00	5.923
0.025	7462	6.12	5.834
0.0125	29566	123.55	5.796

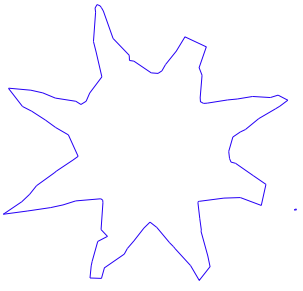
Table 4
Energy and CPU time on adaptive meshes.

	Node	CPU time (s)	Eng.
Adap. mesh1	1351	3.72	5.78
Adap. mesh2	2296	5.96	5.76
Adap. mesh3	4554	20.72	5.74

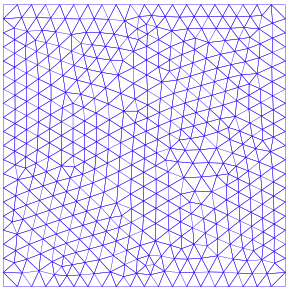
Therefore, the proposed method is quite different from cell-average based finite volume counterpart, which requires special numerical flux to control oscillations. This merit improves the stability of our method significantly, which is proved by various numerical examples. Also the method is easy-to-implement and readily to be extended for surface smoothing. And applying the above FV scheme to adaptive curve smoothing is quite straightforward if proper adaptive refining mechanisms can be developed, which will be presented in the following section.



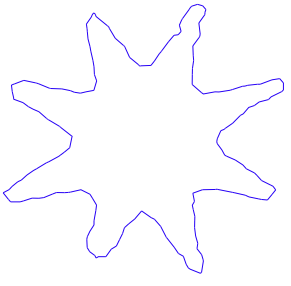
(a) Noisy eight-sided star curve



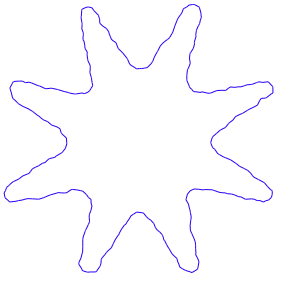
(b) The result on mesh1



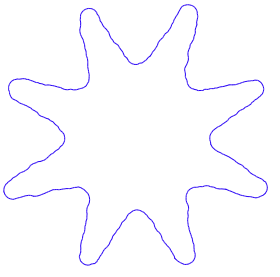
(c) mesh1



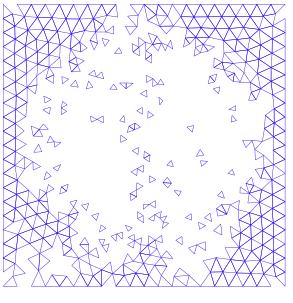
(d) The result on mesh2



(e) The result on mesh3



(f) The result on mesh4



4. Adaptive curve smoothing

An adaptive curve smoothing procedure on unstructured triangular meshes consists of loops of the form **SOLVE** → **SIZING MODIFY** → **REFINE**. The procedure **SOLVE** solves the time-dependent PDE (2) for the finite volume solution on the current mesh. With the computed level set function values on vertices, the smoothed curve is obtained as the zero level set. Then the procedure **SIZING MODIFY** modifies the sizing of the vertices falling within a narrow band enclosing the zero level set, i.e. the new curve. Finally, the procedure **REFINE** modifies the current triangular mesh by inserting points or contracting edges iteratively so that the refined mesh is consistent with the new mesh sizing distribution. The procedure **SOLVE** was discussed in Section 3, and mesh size modification and refinement will be discussed next, which will be followed by a flow chart of the adaptive curve smoothing algorithm.

4.1. Mesh size modification

Basically, the mesh size of each vertex in the current triangular mesh is to be modified based on the computed solution, i.e. the obtained new curve represented by the zero level set. Suppose that h_0 denotes the sizing function of the current mesh, and the signed distance function ϕ is also computed for each vertex, then the new mesh size is modified to $h_0/2$ for the vertices with $|\phi| < ch_0$, which means that the mesh sizing will be halved within a ch_0-width narrow band around the smoothed curve. And c is a constant parameter which may be tuned in each case. To ensure that the following new solution curve will lie within the refined area, the width of the band shall not be too narrow and c is set to be bigger than 2 in our numerical examples. On the other hand, a too large c will result in unnecessary refinement. A suggested choice is $2 \leqslant c < 4$. An optimal choice of c will be a future working direction and posteriori error estimate techniques may be pursued.

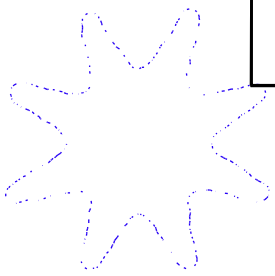
The mesh sizing modification will be followed by sizing gradation by applying the H-correction procedure proposed in [14], which will reduce the magnitude of ratio of neighboring edge lengths and smooth the sizing distribution as a result.

4.2. Mesh refinement through edge splitting/contraction

The refinement of a given triangular mesh is composed of several local mesh operations which include edge splitting/contraction, edge swapping and point smoothing [15–18]. Edge splitting and contraction are based on the computation of normalized edge lengths [17], which is utilized as a criterion for refining or coarsening a given edge. Usually, the value $C_s = \sqrt{2}$ is assigned to be the splitting parameter, and the value $C_c = 1/\sqrt{2}$ is set to be the coarsening parameter. For any edge whose normalized length is larger than C_s, a new point is introduced on the midpoint of the edge which is then halved and the two triangles adjacent to the edge is replaced by four new triangles. For any edge whose normalized length of an edge is less than C_c, the edge will be contracted in the way that either two end points are merged into their midpoint or one of them is removed while keeping the other. In edge splitting and contraction, local operations involving edge swapping and point

Table 5
Energy and CPU time on adaptive meshes.

	Node	CPU time (s)	Eng.
Adap. mesh1	503	0.985	7.026
Adap. mesh2	1236	2.03	6.917
Adap. mesh3	3386	4.16	6.897
Adap. mesh4	8271	14.16	6.599



smoothing are also performed to improve the mesh quality. The readers are referred to [19–25] for the details of the four elementary mesh modification operators.

The refinement procedure modifies, iteratively, an existing triangulation through edge splitting or contraction. The objective is to ensure that mesh elements are in a better conformity with the size specification. Edge lengths computed with respect to vertex sizing function are compared with the given splitting parameter C_s and contraction parameter C_c. After the edge splitting and contraction, the triangulation is finally optimized via a combination of edge swapping and point smoothing. The readers are referred to [17,18] for the details of a complete procedure of the mesh refinement and coarsening.

To summarize, the following is the flow chart of the algorithm for adaptive curve smoothing.

Algorithm 4.1. Adaptive curve smoothing

Step 1. Define a rectangular region Ω which shall cover the given noisy curve Γ_0. Generate an initial coarse triangular mesh M_0 of Ω and derive the mesh sizing function h_0 for the vertices, which can be evaluated as the average length of the edges connecting the vertices. Compute the signed distance function ϕ_0 of Γ_0 by fast sweeping[26]. Give an initial ϕ^0 and compute the initial energy E^0. Set the current triangular mesh $M = M_0$.

Step 2. Solve Eq. (2) on M by the finite volume scheme and get the solution ϕ^n. Compute the energy E^n. If $\frac{|E^n - E^{n-1}|}{E^{n-1}}$ is less than a given tolerance (which is set to be 10^{-4} for the examples in this paper), stop; otherwise goto Step 3.

Step 3. Modify the mesh size distribution h_n based on ϕ^n. Refine M based on the new mesh size function h_n and interpolate ϕ_n on the modified mesh M. Let $\phi_0 = \phi^n$, update h_n and goto Step 2.

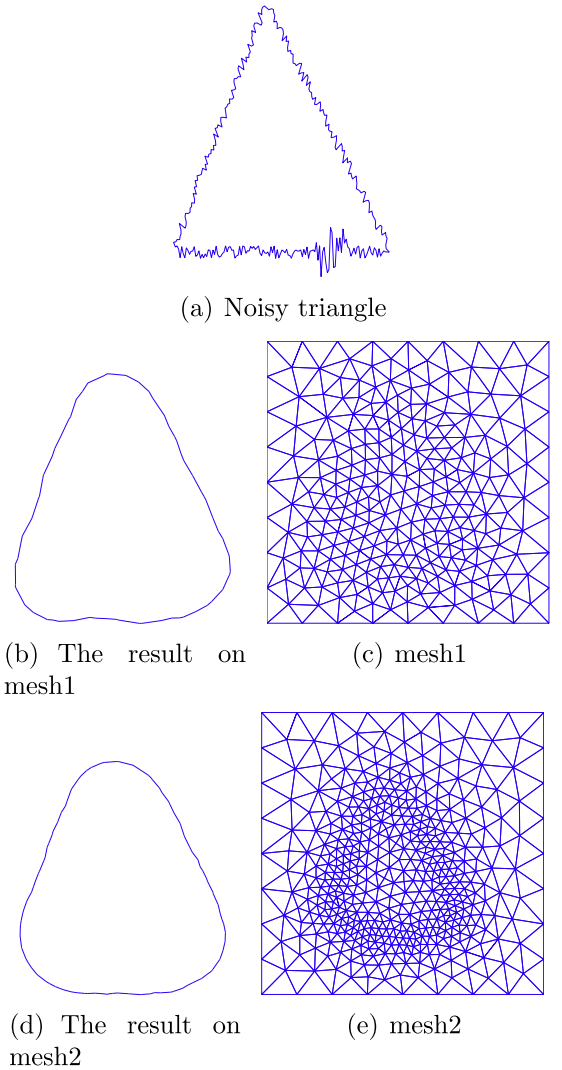


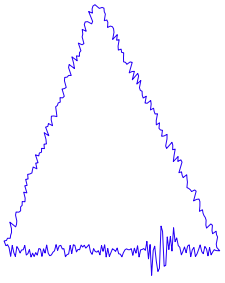
Fig. 11. Noisy triangle smoothed on different meshes.

5. Numerical experiments

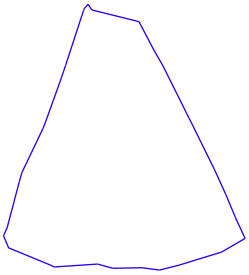
In this section, using the above proposed adaptive curve smoothing method several experiments are conducted. In each example, the curve is initially represented by its signed distance function, which is computed via the fast sweeping method introduced in [26]. After several time steps the ϕ in Eq. (2) may not be signed distance function strictly due to numerical round-off errors, which is demonstrated in Fig. 3. But this will not affect the quality of the resulting curve since actually all the level set contours are regularized in modified equation (2). In the following examples re-initialization is performed every 1000 iterations so as to correct energy computation. And our numerical results show that it is unnecessary to do more frequent re-initializations. The data statistics of each example involve the comparison of the numbers of mesh nodes, CPU time and energy on uniform and adaptive meshes. CPU times are only counted for FV iterations on each mesh and the overhead of the mesh refinement is excluded since it is negligible comparing to FV iterations, usually only taking 2–4%. In this context, by uniform mesh we mean that the mesh size associated to each vertex is almost constant. As for time steps, since the explicit time discretization is utilized, the CFL condition must be satisfied which means $dt \leqslant h_{min}^2$ where h_{min} is the smallest mesh size. In the following experiments, $dt = 1e-6$ is small enough even for adaptive meshes.

Experiment 1. A noisy circle

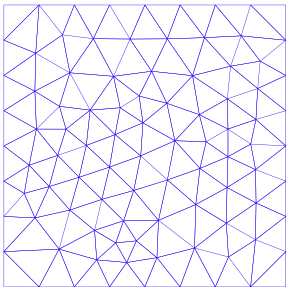
The input curve in the first experiment is a noisy circle shown in Fig. 4(a), which is generated by radially perturbing the points on a circle. The similarity parameter λ is set to be 0.1. The curves shown in Fig. 4(b)–(d) are the solutions obtained



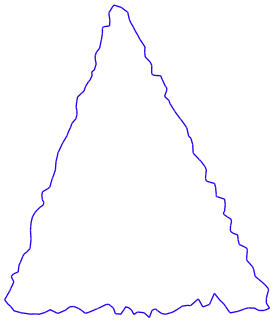
(a) original curve

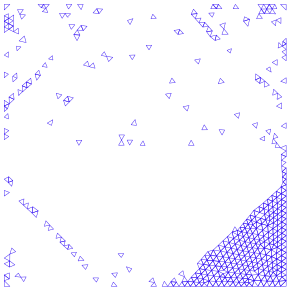


(b) Smoothed curve on mesh1 with $\lambda = 50$



(c) mesh1





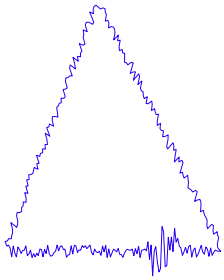
(d) Smoothed curve on mesh2 with $\lambda = 50$

on the adaptive meshes shown in Fig. 4(e)–(g), while the energy data is contained in Table 1, where 'Adap.' denotes 'adaptive'.

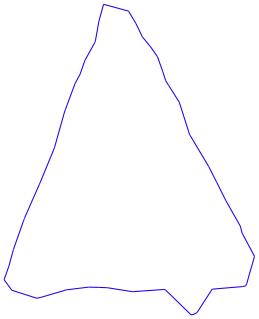
Both the solution curves and the data contained in Table 1 clearly show that the adaptation converges as the mesh is refined around the solution curve. The energy is decreasing from 4.267 to 4.155 and 4.142 which demonstrates the convergence. The fourth adaptive mesh *mesh4* is omitted here as the solutions on *mesh3* and *mesh4* have minute difference, which is clearly illustrated by the energy comparison in Table 1. The comparison between the solution curve on a uniform mesh with sizing $h = 0.0125$ and that on an adaptive mesh with the minimal mesh sizing $h_{min} = 0.0085$ is conducted and the two curves are shown in Fig. 5(a) and (b), respectively. The data statistics for the computations on three uniform meshes with sizing $h = 0.05, 0.025, 0.0125$ is contained in Table 2, which compares with the data in Table 1. It takes 76.45 s to obtain a solution with its energy being 4.15 on a uniform mesh, while it only needs around 15 s on an adaptive mesh for a solution with the energy being 4.155. This shows that the proposed adaptation procedure is very effective for curve smoothing.

Experiment 2. A noisy rectangle.

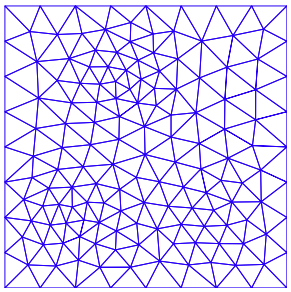
The second experiment is about a noisy rectangle. The computational results on uniform meshes are given in Fig. 6. It is obvious that the curve becomes smoother as the mesh goes finer. Fig. 7 shows the adaptive meshes and the corresponding solution curves. Here the similarity parameter λ is set to be 3.0. With far less mesh nodes, the adaptive meshes can give



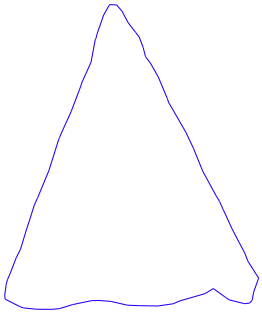
(a) Noisy triangle



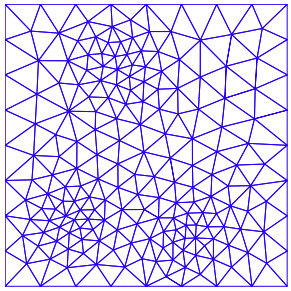
(b) Smoothed curve on mesh1 with $\lambda = 50$



(c) mesh1



(d) Smoothed curve on mesh2 with $\lambda = 50$



(e) mesh2

Fig. 13. Noisy triangle smoothed on curvature-based adaptive meshes.

results which are as accurate as those on uniform meshes. In Fig. 8, the curve smoothed on a uniform mesh with sizing $h = 0.0125$ and that on an adaptive mesh (*mesh3* in Fig. 7) are shown together for comparison.

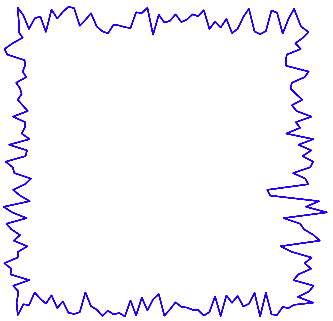
Table 3 contains the CPU time and energy data for the results obtained on uniform meshes with sizing $h = 0.05, 0.025, 0.0125$, respectively. It is clear that the energy is decreasing as long as mesh size becomes small, and also the CPU time becomes longer. Table 4 includes the CPU time and energy data for smoothed curves obtained on adaptive meshes shown in Fig. 7. From Table 4, it can be seen that in order to reach the same energy, uniform meshes need four times mesh nodes more than adaptive meshes do, which demonstrates the speed-up of the adaptation procedure significantly.

From the experiment it can also be observed that although smooth curves can be efficiently obtained by adaptive mesh refinement around the whole curve, the corners are smoothed out during the process. The phenomenon will be further investigated later.

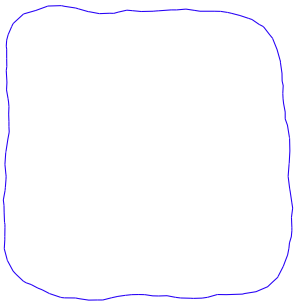
Experiment 3. A noisy star-shaped curve.

To further show the effectiveness of the proposed adaptive procedure, a complicated star-shaped noisy curve is smoothed in this experiment with $\lambda = 30$. The smoothed curves are shown in Fig. 9, and the energy and CPU time data are presented in Table 5.

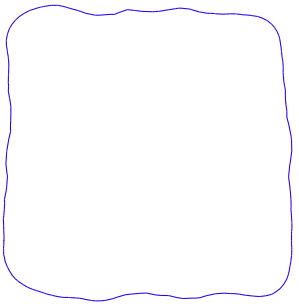
The solution obtained on a uniform mesh with 46271 nodes, is given in Fig. 10(b). The smoothed curve on an adaptive mesh included in Fig. 9(i) is shown in Fig. 10(a). The adaptive mesh has only 8271 mesh nodes, which shows the effectiveness of the adaptation procedure very well.



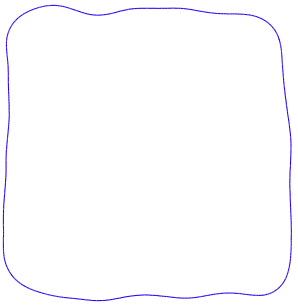
(a) Noisy rectangle



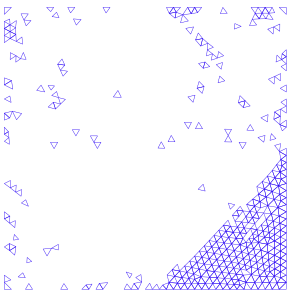
(b) Smoothed curve on mesh1



(c) Smoothed curve on mesh2



(d) Smoothed curve on mesh3

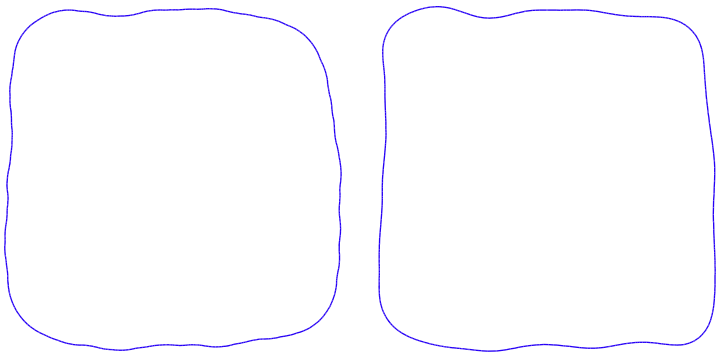


Experiment 4. Refinement for corner preserving.

In this experiment, a noisy triangular curve is taken as an example and variant refinement strategies will be investigated. Appropriate choices of similarity parameters and adaptive mesh refinement help to keep corners.

First, we consider the cases when λ is small, which are shown in Fig. 11. It can be seen that the noisy triangle shown in Fig. 11(a) is smoothed on a very coarse mesh contained in Fig. 11(c). The corresponding smooth curve is shown in Fig. 11(b), where $\lambda = 3$. Next, the mesh is refined around the smoothed curve as in Fig. 11(e), and smoothing is conducted. It can be clearly observed that the finer mesh improves the smoothness of the curve. However, the three corners are smoothed out.

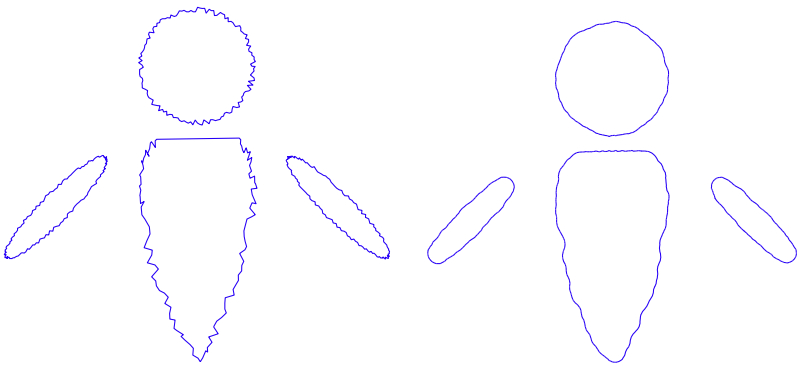
In order to keep the corners, the similarity parameter λ is increased to 50 and the smoothing is conducted on four different meshes. The first mesh is of uniform size and shown in Fig. 12(c), on which a smooth curve is obtained as shown in Fig. 12(b).



(a) Noisy rectangle smoothed on the adaptive mesh refined around the curve

(b) Noisy rectangle smoothed on the adaptive mesh with further refinement around the corners

Fig. 15. Noisy rectangles smoothed on different type of adaptive meshes.



(a) Noisy robot

(b) Smoothed robot

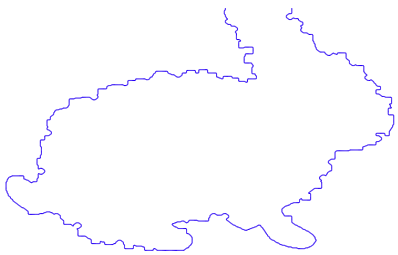


Table 6
CPU time comparison.

	Robot		Rabbit	
	Uniform mesh	Adaptive mesh	Uniform mesh	Adaptive mesh
Node	46271	13095	46271	8913
CPU time (s)	128.22	15.36	108.53	13.47
Energy	6.52	6.49	4.84	4.89

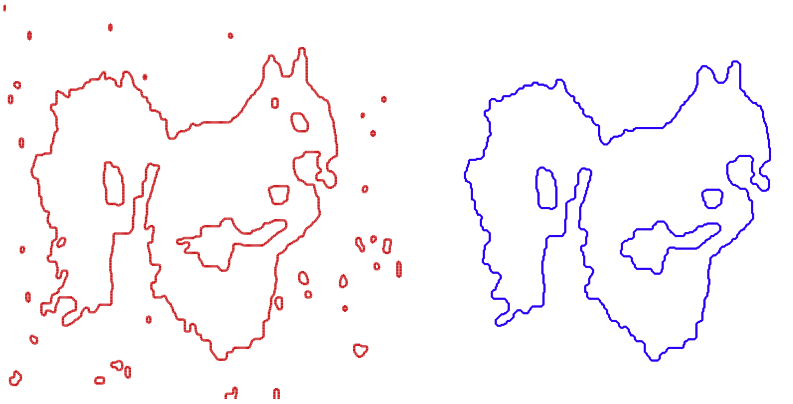


Fig. 17. Smooth contours from image segmentation.

The three corners are all cut off. Then, a finer uniform mesh is used and the obtained curve preserves the corners much better, but with less smoothness, which is illustrated in Fig. 12(d) and (e). Based on the observation, curvature-based adaptive meshing could be of the aid to keep the corners, in the sense that finer meshes can be used in the region where the curvature is relatively larger and coarser mesh can be used in the region where the curvature is smaller. The first curvature-based adaptive mesh is shown in Fig. 13(c), where the triangulation is refined locally in the region around the three corners where the curvature is relatively larger. The curve obtained on the mesh has a fake corner. However, after further refinement both in the corner area and the region around the fake corner, the smoothed curve resolves the fake corner and preserves the three corners very well, which is shown in Fig. 13(e). The nice property of keeping the corners through curvature-based mesh refinement demonstrates that adaptive meshing not only saves the computational cost, but also assists in corner preserving, which will be further demonstrated in the following examples.

Experiment 5. Another example for corner keeping.

The strategy developed in the previous experiment is used here to handle the noisy rectangle introduced in Experiment 3. First, the noisy curve is smoothed with $\lambda = 3$ on a uniform mesh shown in Fig. 14(e). Then, the mesh sizes of the nodes lying in the circles around the four corners are halved. And the similarity parameter λ is modified to be 30 around the corners and $\lambda = 2$ elsewhere. Fig. 14(f) shows the mesh after the first refinement and Fig. 14(c) includes the corresponding smoothed curve. Finally, the triangular mesh is further refined as shown in Fig. 14(g). The final curve is presented in Fig. 14(d). Comparing with the result in Fig. 7(g), the solution curve is smoother and more regular in the sense that the four corners are well kept and the other parts of the curve is much more straight, which is clearly demonstrated in Fig. 15.

Experiment 6. Two more complicated curves.

The sixth experiment considers two noisy curves with more complicated geometries. Fig. 16 contains the noisy and smoothed curves. The solutions show that the proposed adaptive curve smoothing method can handle complicated curves very well. The CPU times on uniform and adaptive meshes are given in Table 6 and the data clearly illustrates the effectiveness of the method. It should be pointed out that as solution curves on adaptive and uniform meshes have minute difference, only the solution curves on adaptive meshes are presented.

Experiment 7. Curves from image segmentation.

The last experiment handles a noisy curve obtained from image segmentation. Fig. 17a contains the contours after segmentation and the smoothed curve is shown in Fig. 17(b). For simplicity, the data statistics including the CPU time, energy and mesh node numbers are omitted here. All the computational results show that the proposed approach is as effective as in the previous examples. Especially, the capability of the method to deal with noisy input and to handle topological changes is clearly demonstrated.

6. Conclusions and future work

A novel adaptive finite volume method is proposed and tested for variational curve smoothing. The method is based on level set implementation on triangular meshes in an easy-to-implement and yet very effective way. The numerical examples show that the method is both efficient and effective. It is quite straightforward to extend the method to variational surface smoothing, which is under current investigations. In order to preserve the corners, curvature related high order terms [27,28] and anisotropic diffusion [4,29] method are also under consideration. A posteriori error estimate based refinement strategy will be one of the future working directions, and appropriate metrics to measure the effectiveness of the adaptation procedure will be developed in the future as well.

Acknowledgments

The authors would like to thank the referees for valuable suggestions and comments.

References

[1] G. Taubin, A signal processing approach to fair surface design, in: Proceedings of SIGGRAPH, 1995, pp. 351–358.
[2] M. Desbrun, M. Meyer, P. Schröder, A.H. Barra, Implicit fairing of irregular meshes using diffusion and curvature flow, in: Proceedings of SIGGRAPH, 1999, pp. 317–324.
[3] Y. Wang, D. Wang, A. Bruckstein, On graph-cuts based curve smoothing and reconstruction, submitted for publication.
[4] U. Clarenz, U. Diewald, M. Rumpf, Anisotropic geometric diffusion in surface processing, in: Proceedings of IEEE Visualization, 2000, pp. 397–405.
[5] V. Caselles, R. Kimmel, G. Sapiro, Geodesic active contours, International Journal of Computer Vision 22 (1) (1997) 61–79.
[6] S. Osher, J.A. Sethian, Fronts propagating with curvature dependent speed: algorithms based on Hamilton–Jacobi formulations, Journal of Computational Physics 79 (1988) 12–49.
[7] B. Milne, Adaptive Level Set Methods Interfaces, Tech. Rep., University of California, Berkeley, Department of Mathematics, 1995.
[8] T.J. Barth, J.A. Sethian, Numerical schemes for the Hamilton–Jacobi and level set equations on triangulated domains, Journal of Computational Physics 145 (1) (1998) 1–40.
[9] C.Q. Hu, C.W. Shu, Weighted essentially non-oscillatory schemes on triangular meshes, Journal of Computational Physics 150 (1999) 97–127.
[10] I. Sazonov, D. Wang, O. Hassan, K. Morgan, N. Weatherill, A stitching method for the generation of unstructured meshes for use with co-volume solution techniques, Computer Methods in Applied Mechanics and Engineering 195 (2006) 1826–1845.
[11] H.K. Versteeg, W. Malalasekra, An Introduction to Computational Fluid Dynamics: The Finite Volume Method, second ed., Pearson Education Limited, Harlow, Essex CM20 2JE, England, 2007.
[12] T. Chan, S. Esedoglu, M. Nikolova, Finding the global minimum for binary image restoration, in: IEEE International Conference on Image Processing, 2005, pp. 121–124.
[13] S. Osher, F. Santosa, Level set methods for optimization problems involving geometry and constraints. I. Frequencies of a two-density inhomogeneous drum, Journal of Computational Physics 171 (1) (2001) 272–288.
[14] H. Borouchaki, F. Hecht, P.J. Frey, Mesh gradation control, International Journal for Numerical Methods in Engineering 43 (1998) 1143–1157.
[15] P.J. Frey, About surface remeshing, in: Proceedings of the 9th International Meshing Roundtable, 2000, pp. 123–156.
[16] P.J. Frey, H. Borouchaki, Geometric surface mesh optimization, Computing and Visualization in Science 1 (1998) 113–121.
[17] D. Wang, O. Hassan, K. Morgan, N.P. Weatherill, EQSM: an efficient high quality surface grid generation method based on remeshing, Computer Methods in Applied Mechanics and Engineering 195 (2006) 5621–5633.
[18] D. Wang, O. Hassan, N.P. Weatherill, K. Morgan, Enhanced remeshing from STL files and its application to surface grid generation, Communications in Numerical Methods in Engineering 23 (2007) 227–239.
[19] P.J. Frey, P.-L. George, Mesh Generation: Application to Finite Elements, second ed., ISTE Publishing Company, 2008.
[20] J. Thompson, B.K. Soni, N.P. Weatherill, Handbook of Grid Generation, CRC Press, LLC, Boca Raton, FL, 1999.
[21] H. Borouchaki, P.L. George, Parametric surface meshing using a combined advancing-front gerneralized Delaunay approach, International Journal for Numerical Methods in Engineering 49 (2000) 233–259.
[22] P.M. Knupp, Achieving finite element mesh quality via optimization of the Jacobian matrix norm and associated quantities. Part I – A framework for surface mesh optimization, International Journal for Numerical Methods in Engineering 48 (2000) 401–420.
[23] D. Wang, X. Wang, A three-dimensional adaptive method based on the iterative grid redistribution, Journal of Computational Physics 199 (2004) 423–436.
[24] D. Wang, A new method for efficient generation of high quality surface triangular meshes, Communications in Computational Physics 1 (4) (2006) 716–735.
[25] Q. Du, D. Wang, Boundary recovery for three dimensional conforming Delaunay triangulation, Computer Methods in Applied Mechanics and Engineering 193 (2004) 2547–2563.
[26] J. Qian, Y.-T. Zhang, H.-K. Zhao, Fast sweeping methods for Eikonal equations on triangular meshes, SIAM Journal on Numerical Analysis 45 (1) (2007) 83–107.
[27] M. Elsey, S. Esedoglu, Analogue of the Total Variation Denoising Model in the Context of Geometry Processing, Tech. Rep., University of California, Berkeley, Department of Mathematics, 2007.
[28] M. Eigensatz, R.W. Sumner, M. Pauly, Curvature-domain shape processing, Computer Graphics Forum 27 (2) (2008) 241–250.
[29] T. Tasdizen, R. Whitaker, P. Burchard, S. Osher, Geometric surface smoothing via anisotropic diffusion of normals, in: Proceedings of the Conference on Visualization, IEEE Computer Society, 2002, pp. 125–132.