J Math Imaging Vis (2010) 37: 183–203
DOI 10.1007/s10851-010-0201-y

On Variational Curve Smoothing and Reconstruction

Yu Wang · Desheng Wang · A.M. Bruckstein

Published online: 15 April 2010

Abstract In this paper we discuss and experimentally compare variational methods for curve denoising, curve smoothing and curve reconstruction problems. The methods are based on defining suitable cost functionals to be minimized, the cost being 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.

Keywords Curve smoothing · Curve reconstruction · Variational methods · Level sets · Graph-cuts · Corner detection

1 Introduction

Curve smoothing is an interesting problem from both practical and theoretical views. It is a popular pre-processing and post-processing step in addressing various computer vision problems. As an example, following segmentation object boundaries have many zigzags and to do further analysis the curve must be smoothed. Another important and related problem is curve reconstruction from an unorganized set of data points. Figure 1 is an illustration of these two problems.

Surface reconstruction problems from 3D points or noisy 3D triangulated surfaces received a lot of attention from the computer graphics and CAD communities due to the importance of this problem in an era when 3D scanners are commonly used by various industries. Therefore many of the issues discussed in this paper have been the subject of intense investigation in the important context of surface fitting, reconstruction and smoothing. For example, Hornung and Kobbelt [15] proposed to perform surface reconstruction based on a variational functional that measures the surface fit to the data by integrating a distance function to the given 3D point set. This is, of course, a Geodesic Active Surface functional [8, 9, 35]. By minimizing a functional with TV regularization and L^1 data fidelity, Zach et al. [33] presented an approach to reconstruct 3D models

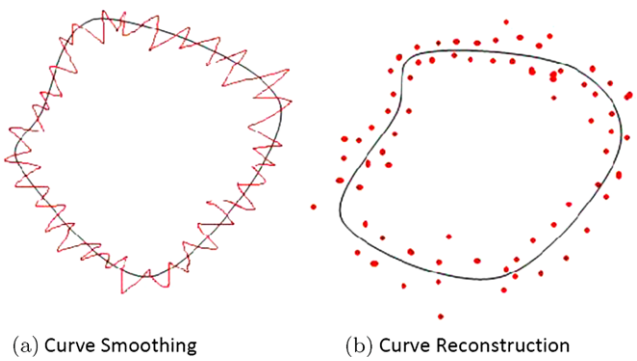


Fig. 1 Curve smoothing and reconstruction

Professor A.M. Bruckstein's work was supported in part by an NTU joint visiting professorship at the School of Physical and Mathematical Sciences and the Institute for Media Innovations. The work is supported in part by the NTU start-up grant M58110011, Singapore MOE ARC 29/07 T207B2202 and NRF 2007IDM-IDM 002-010.

Y. Wang · D. Wang
Division of Mathematical Sciences, School of Physical and Mathematical Sciences, Nanyang Technological University, Singapore, Singapore

Y. Wang
e-mail: wang0312@ntu.edu.sg

D. Wang
e-mail: desheng@ntu.edu.sg

A.M. Bruckstein (✉)
Department of Computer Science, Technion IIT, Haifa, Israel
e-mail: freddy@cs.technion.ac.il

from range images. Lempitsky and Boykov in [22] proposed a so-called "touch-expand" graph-cuts algorithm to surface fitting. Combining flux with a regularization term, an idea developed in [19, 20] enabled their method to reconstruct fine details by graph-cuts. The above-mentioned, and many more papers that deal with surface smoothing and reconstruction proposed a wealth of ideas aimed to solve difficult topological and geometric issues that arise in 3D. These works might be regarded as 3D generalizations of the issues we deal with in this paper and hence one might argue that the simpler 2D problems have already been dealt with, as particular cases of 3D problems. While this might be true in some instances, the 3D studies necessarily address a wealth of issues that cannot arise in 2D, and hence we should be able to do more in the planar case by carefully considering the simpler geometry and fully exploiting it. Furthermore we can readily test new ideas in the 2D case and sometimes export them to the higher dimensional cases. In our view, this paper servers a dual purpose: it is a survey of ideas on variational methods in the context of curve smoothing and reconstruction and a showcase of experimental tests exploring some new ideas that involve combination of functionals and data adaptive selection of weighting parameters in a context where the results can be readily obtained and assessed.

To address curve smoothing and curve reconstruction problems in a variational framework we define "distance" functionals that measure the similarity of curves from raw data of curves or from sets of points. Several options are available, the most natural one, in our opinion, being based on the definition of a "distance field" based on the data and integrating the field intensity along the curve. Other options could involve the Hausdorff distance between the geometric objects of interest or Frechet distances, but these are much harder to deal with. The "distance field" approach is also better suited at incorporating various types of weighting functions with adaptive distances that may take into consideration knowledge of spatially varying and data dependent noise (see e.g. heteroscedastic problems [18]).

Using distance functions to measure similarity to data is not a new idea. The idea appears in various papers that deal with image segmentation of edge integration [5, 8, 9, 16, 17, 23, 32] or in using shape priors for object detection and tracking [10, 25]. The distance functions defined in image analysis measured distances to edge regions or high gradient points and were often used to define metrics implicitly based on the 2D image analyzed. We shall here also consider an alternative approach which represents closed curves via inside-outside indicator functions and measures distance between curves by integrating functionals measuring the area of "symmetric differences" based on such indicator functions, see e.g. [7]. These different similarity measures yield different functionals whose details will be discussed in Sect. 2. As an addition to the basic "similarity to data" term we must add to our functional a smoothing term, which, following [27], will be the total variation of the curve. Geometrically, this quantity turns out to simply be the curve's length.

We shall analyze methods for planar curve smoothing and reconstruction, both in order to assess the properties of such variational methods in the basic and relatively simple 2D case, which has many practical applications (post-processing segmentation results, and various geometric problems like determining the minimal length curves contained in a band) and test new ideas that might be beneficial for all the other applications in 3D and even video processing. Our work tests two important functionals and the combination of these functionals and thoroughly compares the basic numerical implementations in these fundamental 2D problems—something that to the best of our knowledge was not done before.

To carry out the minimization, level set methods are commonly considered. However, level set methods have some well documented problems, such as converging to local minima and long iteration times. To address such problems one may use narrow-band algorithms and design an adaptive method to save the computational cost [30]. Another popular choice is the use of "graph-cuts based" methods. This approach, based on min-cut/max-flow algorithms, has the merit that global minima are efficiently attained. In this paper we apply graph cuts on both symmetric-difference based and on distance-field based measures and compare the results with the level-set implementation. For symmetric-differences-based functionals the graph cuts approach is generally faster than the standard level set implementation. When the graph-cuts method is used in conjunction with the active contour model, narrow-band implementations are commonly used to effectively eliminate the trivial global minima and reduce computational cost. As a matter of fact, the narrow-band implementation is equivalent to solving a modified active contour model, which profitably combines the merits of symmetric-difference-based and distance-function-based measures. Moreover, instead of using uniform relative weight parameters, data dependent parameters are proposed, which thereby enable the method to preserve corners.

This paper is organized as follows. In Sect. 2 we introduce functionals for curve smoothing and reconstruction problems. In Sect. 3 we briefly review the level set implementation and related numerical schemes. In Sect. 4, we discuss the implementation of graph-cuts method and a description of the graph construction is briefly revised. The implementation details are presented in Sect. 5. In Sect. 6, we discuss the curve reconstruction application and its main challenges. In Sect. 7 several variations and many examples are presented. Then some concluding remarks are made in Sect. 8.

2 The Curve Smoothing and Reconstruction Functionals

The Rudin-Osher-Fatemi approach for signal and image denoising has had great success in recent years [27]. This denoising framework can be readily adapted to different situations. A particularly interesting scenario is curve, as opposed to function, smoothing: the noise is assumed to be a perturbation of some smooth underlying curve and the user might be interested in smoothing out the rough curve. Another interesting problem is curve reconstruction which addresses the issue of reconstructing a curve from a set of unordered sample points without any prior knowledge except the fact that the points are noisy samples of a curve. These two problems are closely related and in this paper we treat them as different instances of the same variational formalism.

The standard ROF approach to denoising is: given a noisy function f we seek a smooth version u by minimizing the cost functional

$$E_{\mathrm{ROF}}(u,\lambda)=\int_{R^2}|\nabla u|dxdy+\lambda\int_{R^2}(u-f)^2dxdy$$

where $\lambda>0$ is a relative weight parameter to be selected. In this functional, the first term measures the smoothness of the solution, while the second term represents the similarity between given data and the solution. Following this idea, all curve denoising functionals will comprise the same basic components. As mentioned in [7], the area of symmetric difference of the indicator functions defined by the simple closed curves can also be used to measure the distance between two curves. This functional is

$$E_1(C,\lambda)=\int_{\Omega}|\nabla\mathbf{1}_C|d\Omega+\lambda\int_{\Omega}|\mathbf{1}_C-\mathbf{1}_{C_0}|d\Omega \tag{1}$$

where Ω is the region of interest, and $\mathbf{1}_C$ and $\mathbf{1}_{C_0}$ are the indicator functions of two curves, C_0 the given one and C the smoothed curve we seek. In this case, the second term is exactly the difference of the area enclosed "between" two curves while the first term is the perimeter of the smoothed curve we seek. The merit of this functional is that the optimal curves are global minimizers of the E_1. As mentioned in the Introduction, this functional has been adopted in various applications, such as reconstructing 3D model from range images [33]. In fact, the indicator functions of the solution curve and the given curve in (1) can be replaced by signed distance functions as well [30], which is convenient for level-set implementation. However, this functional is not readily adapted to the problem of curve or surface reconstruction from point clouds, in which case indicating all interior and exterior points based on a nonexistent curve or surface (an object that is being sought by us) is essentially impossible. Therefore, an ideal functional that can handle these problems should rely only on the data points and/or on the induced topological information.

Provided that we represent a curve as a level-set of a bivariate function, and in particular via the distance function induced by itself in the plane, the distance between two curves can be written in the form,

$$\mathrm{dist}(C^0,C)=\oint\varphi^0(C(\tau))|C_\tau|d\tau+\oint\varphi(C^0(\tau))|C^0_\tau|d\tau$$

where φ^0 is the distance function to a given "data" curve $C^0(\tau)$ and φ is the distance function to the expected smooth "output" curve $C(\tau)$. Considering only the first part of this distance measuring the closeness of $C(\tau)$ to $C^0(\tau)$ via its induced distance field φ^0, we get the following functional

$$\mathrm{dist}_{C^0}(C)=\int_0^1\varphi^0(C(\tau))|C_\tau|d\tau.$$

Adding to the distance to C^0 a functional measuring the L_1 norm of the curve, i.e. $\int|C_\tau|d\tau$ (which is the curve length) yields

$$E_2(C,\alpha,\beta)=\int_0^1(\alpha+\beta\varphi^0(C(\tau)))|C_\tau|d\tau \tag{2}$$

where α,β are parameters (constants or functions) to be selected. This functional is formally identical to the geodesic active contours model, the main problem of which is that global minimizers are isolated points trivial in most applications, the meaningful solutions corresponding to local minimizers. This is not a big problem for PDE-based methods which always converge to a local minimizer near properly selected initial solution, but would be a disaster for straightforward graph-cuts implementations which always yield the global minimizers.

It is interesting to note that, like in the image segmentation problem, for simply closed curve smoothing we have here two types of methods: some using a "region based" functional as given by (1); and others "edge based" functionals, as exemplified by (2). The fact that each of them has advantages and drawback leads to the natural idea of combining them together so as to combine the advantages of both functionals and jointly avoid their drawbacks. This leads to the consideration of

$$E_3(C,\alpha,\beta,\lambda)=\oint_C\alpha ds+\int_{\Omega}\lambda|\mathbf{1}_C-\mathbf{1}_{data}|d\Omega+\oint_C\beta\varphi^0(C)ds,$$

where $\mathbf{1}_{data}$ is the indicator function of the data set. It is worth pointing out that for curve smoothing problem $\mathbf{1}_{data}$ is just the indicator function of the given curve C_0, and for the curve reconstruction problem the way of defining it depends on the determination of a narrow band containing the data

points whose details will be given in Sect. 6. Changing the line integral to area integral, we get

$$E_3(C,\alpha,\beta,\lambda) = \int_\Omega \big(\alpha + \beta\varphi^0(C)\big)|\nabla \mathbf{1}_C| d\Omega + \int_\Omega \lambda |\mathbf{1}_C - \mathbf{1}_{data}| d\Omega. \quad (3)$$

If $\lambda = 0$, above functional becomes the one used in [15, 32]. When $\lambda = \text{const}$ and $\beta = 0$, it turns out to be the functional used in [33]. If λ has the form

$$\lambda(x) = \begin{cases} 0 & \text{if } \varphi^0(x) \le c, \\ \infty & \text{if } \varphi^0(x) > c, \end{cases}$$

where c is a constant, (3) becomes the functional involved in the narrow-band implementation of the graph-cuts-based-method [15, 32]. It can be seen that various modifications can be incorporated in the functional (3), each of them just corresponding to a certain confidence function $\lambda(x)$, which "encodes" that to what extent we rely on the data-induced topological information. On the other hand, the different choices of the $\lambda(x)$ may result in different global minima for the same problem, e.g. the width of the "narrowband" will affect the final reconstruction result. Moreover, α, β, λ could be functions depending on the data curve. For example, the curves obtained by minimizing functional E_3 may lose meaningful corners. It hence would be natural to use a large similarity parameter in a corner region and small one elsewhere to remedy this problem. Therefore, a simple choice is to let the similarity parameter or regularity parameter depend on certain "cornerness" measures of the data curve, e.g. a function of the local averaged curvature. Here, the functional would be

$$E_4(C) = \int_\Omega \big(\alpha_{data}(x) + \beta\varphi^0(C)\big)|\nabla \mathbf{1}_C| d\Omega + \int_\Omega \lambda_{data}(x) |\mathbf{1}_C - \mathbf{1}_{data}| d\Omega.$$

By proper choice of the weight functions $\alpha_{data}(x)$ and $\lambda_{data}(x)$, it will then be possible to preserve important features of curves. In Sect. 5, some such choices are discussed and implementation details are given and the corresponding experiments are shown in Sect. 7.

Standard level set methods can be applied to all functionals. In the next section we shall briefly discuss the possibility of various level-set implementations and for simplicity we shall assume a constant relative weight parameter.

3 Level-Set-Methods for Functional Minimization

We shall here briefly review the level set method implementation of the functional (1) and (2), and refer to [7] and [5] for more details. Let ϕ be the indicator function of the curve we seek and φ be the indicator function of the given curve. Then, the minimization flow corresponding to functional (1):

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

The evolution equation can be discretized with the straightforward explicit finite difference scheme:

$$\frac{\phi^{n+1}-\phi^n}{\delta t} = D_x^- F_x^{+,n} + D_y^- F_y^{+,n} - \lambda\frac{\phi^n-\varphi^n}{\sqrt{(\phi^n-\varphi^n)^2+\epsilon}} \quad (4)$$

where

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

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

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

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

and ϵ is a very small constant usually set to be $1e-10$. It is worth mentioning that due to the CFL condition, explicit finite difference schemes always suffer from restrictively short time steps. step in (4) is at least of order $\min((dx)^2, (dy)^2)$. That means this algorithm's complexity is of order almost $O(N^3)$, where N is the number of grids points. More efficient numerical schemes such as AOS, primal-dual model of the TV-norm can, of course, can be used. A comprehensive discussion is beyond the scope of this paper. For details see [6, 16, 17, 21].

For the functional (2), the corresponding Euler-Lagrange equation is:

$$(\lambda\langle\nabla\varphi^0, N_c\rangle - (\lambda\varphi^0 + 1)\kappa_c) = 0$$

where the N_c is the unit normal of the curve and κ_c is the curvature (mean curvature for surface). Let $v = -\lambda\langle\nabla\varphi^0, N_c\rangle + (\lambda\varphi^0 + 1)\kappa_c$, therefore the curve should evolve according to

$$\frac{\partial C(\tau,t)}{\partial t} = vN_c.$$

By means of level-sets function [24] language, $N_c = -\frac{\nabla\phi}{|\nabla\phi|}$, $\kappa_c = \operatorname{div}(\frac{\nabla\phi}{|\nabla\phi|})$ where ϕ is the level set function of the curve C, and above equation can be rewritten as

$$\phi_t = v|\nabla\phi| = (\lambda\varphi^0 + 1)|\nabla\phi|\mathrm{div}\left(\frac{\nabla\phi}{|\nabla\phi|}\right) + \lambda\nabla\varphi^0\cdot\nabla\phi.$$

To discretize the convection term $\nabla\varphi^0\cdot\nabla\phi$ the upwind scheme is needed,

$$K^n = \max(D_x^+\varphi^0, 0)D_x^-\phi^n + \min(D_x^+\varphi^0, 0)D_x^+\phi^n + \max(D_y^+\varphi^0, 0)D_y^-\phi^n + \min(D_y^+\varphi^0, 0)D_y^+\phi^n$$

and the nonlinear diffusion term $|\nabla\phi|\mathrm{div}(\frac{\nabla\phi}{|\nabla\phi|})$ can be approximated by central finite difference scheme,

$$\sqrt{(D_x^0)^2 + (D_y^0)^2}(D_x^- F_x^{+,n} + D_y^- F_y^{+,n}),$$

where $D_x^0 = \frac{\phi_{i,j+1}-\phi_{i,j-1}}{2dx}$, $D_y^0 = \frac{\phi_{i+1,j}-\phi_{i-1j}}{2dy}$. Again combining with forward time divided difference the finite difference scheme for the functional (2) reads:

$$\frac{\phi^{n+1}-\phi^n}{\delta t} = (\lambda\varphi^0 + 1)\sqrt{(D_x^0)^2 + (D_y^0)^2} \times (D_x^- F_x^{+,n} + D_y^- F_y^{+,n}) + \lambda K^n. \quad (5)$$

The CFL time step restriction of above scheme is also of order $\min((dx)^2, (dy)^2)$. We need point out that both schemes (4) and (5) are the first order in time. Although higher order schemes such as TVD RK [28, 29] methods exist, the improved temporal accuracy does not seem to make a significant difference in our application.

Due to the non-convexity of the energy functional, GAC models may get stuck at local minima and hence GAC is highly sensitive to the initial condition. Evidently the global minima for the GAC model corresponds to single points which make no sense for our applications. However, there are many good fixes to this problem. The graph cuts method overcomes this problem with minor modifications. Functional (3) will overcome the drawback of GAC functional and its level set implementation is the combination of the scheme (4) and (5) [30]. We shall next discuss the graph-cuts implementation for functional (1), (2) and their combination (3).

4 Graph-Cuts for Global Minimization

In this section, a description of how to construct a graph where each cut represents a state of energy functional will be given.

Suppose $G = (V, E)$ is a directed graph with nonnegative edge weights that has two special vertices (called terminals), namely, a source s and a sink t. An s-t-cut $C = \{S, T\}$ is a partition of the vertices in V into two disjoint sets S and T such that $s \in S$ and $t \in T$. The cost of the cut is the sum of costs of all edges that go from S to T:

$$c(S, T) = \sum_{u\in S, v\in T, (u,v)\in E} c(u, v).$$

The minimum s-t-cut problem is to find a cut C with the smallest cost. Due to the theorem of Ford and Fulkerson this is equivalent to computing the maximum flow from the source to sink. Note that a cut is a binary partition of the graph and may be viewed as a labeling. Actually our problem can be thought of as a binary partition problem as well. This can be easily seen from the functional (1) in which the curve is identified by the inside-outside indicator. Thus, to find a solution is equivalent to seek a partition of the graph nodes which represents the inside-outside indicator of the optimal curve. Using Cauchy-Crofton formula, it is also possible to translate GAC model to a binary problem, as shown in [3]. It is to be noted that this approach necessarily introduces some metrication errors of the magnitude of the order of grid size that will, however, not bother us too much. Therefore our aim is to construct the graph whose minimum cut or partition is the minimizer of the energy functional.

Let us consider a graph $G = (V, E)$ with $V = \{v_1, v_2, \ldots, v_N, s, t\}$. Each cut on G has some cost; therefore, G represents the energy function mapping from all cuts on G to the set of nonnegative real numbers. Any cut can be described by n binary variables $x_1, \ldots, x_n$ corresponding to vertices in G (excluding the source and the sink): $x_i = 0$ when $v_i \in S$ and $x_i = 1$ when $v_i \in T$. Therefore, the energy functional which G represents can be viewed as a function of n binary variables: $\mathcal{E}(x_1, \ldots, x_n)$ which is equal to the cost of the cut defined by the configuration $x_1, \ldots, x_n (x_i \in \{0, 1\})$.

4.1 Graph Construction of Energy Functional

Normally, there are two types of edges in the graph: N-links and T-links. N-links connect neighboring nodes; T-links connect nodes with terminals. In this subsection we shall state how to use these two kinds of links to represent energy functional. Assume our energy functional has the following form as in [19, 20],

$$\mathcal{E}(x_1, x_2, \ldots, x_n) = \sum_i E^i(x_i) + \sum_{i,j} E^{i,j}(x_i, x_j).$$

The term E^i corresponds to similarity term or T-link. For instance, in the functional (1) after discretizing similarity integration by summation E^i has the form

$$E^i(x_i(v_i)) = |\mathbf{1}(v_i)_C - \mathbf{1}(v_i)_{C_0}|$$

where v_i is the grid point. Therefore, $E^i(x_i)$ either is 1 or 0.

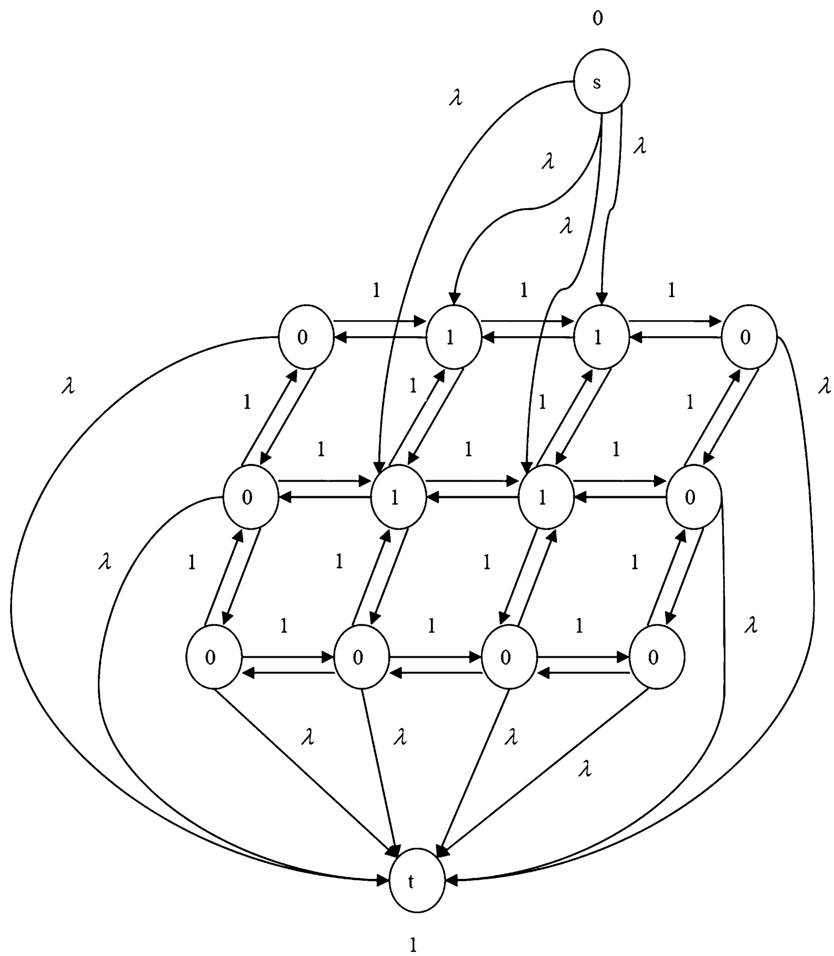


Fig. 2 An example of graph construction

While, the term $E^{i,j}$ corresponds to N-link and has the flavour of the smoothness term such as $\int_\Gamma ds$. There are several ways to view this term. Arc length can be expressed by total variation of inside-outside indicator function of the curve

$$|\nabla \mathbf{1}| = \sqrt{(\mathbf{1}_x)^2 + (\mathbf{1}_y)^2}$$

since this term is not easy to be discretized by graph, the following anisotropic expression is used to measure the smoothness

$$|\nabla \mathbf{1}|_{ani} := (|\mathbf{1}_x| + |\mathbf{1}_y|).$$

Note that for $|\nabla \mathbf{1}|_{ani}$ and $|\nabla \mathbf{1}|$ the following inequalities hold

$$\frac{\sqrt{2}}{2}|\nabla \mathbf{1}|_{ani} \le |\nabla \mathbf{1}| \le |\nabla \mathbf{1}|_{ani},$$

which imply $|\nabla \mathbf{1}|_{ani}$ is a proper smoothness measure. If we use finite difference scheme to discretize $|\nabla \phi|$ on uniform grids we get an isotropic discretization

$$|\nabla \mathbf{1}| \approx \frac{1}{h}\sqrt{(\mathbf{1}_{m,n} - \mathbf{1}_{m,n-1})^2 + (\mathbf{1}_{m,n} - \mathbf{1}_{m-1,n})^2}$$

and the corresponding anisotropic discretization is given by

$$|\nabla \mathbf{1}|_{ani} := \frac{1}{h}(|\mathbf{1}_{m,n} - \mathbf{1}_{m,n-1}| + |\mathbf{1}_{m,n} - \mathbf{1}_{m-1,n}|).$$

In this case the term $E^{i,j}$ has form $|\mathbf{1}_i - \mathbf{1}_j|$ where i and j are indices of the nodes and it can be represented by a bi-directed edge between the node i and j. Above observation is only valid for 4-neighbourhood. Of course, we can use higher order finite difference scheme or unstructured grids to approximate $|\nabla \mathbf{1}|$ and get corresponding anisotropic smoothness term $|\nabla \mathbf{1}|_{ani}$, but this will involve three or more terms. Actually, by means of the Cauchy-Crofton formula dicretization can be also carried out by dividing angles. The reader is refereed to [3, 4] for more discussions on discretizing the arc-length on other neighbourhood systems. Figure 2 is an example of a complete 3 by 4 graph of the functional (1). In this graph the given curve has characteristic function

$$\begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}.$$

As shown in Fig. 2, the nodes which have function value 1 connect to s with an edge of capacity λ while the nodes whose function value are 0 have an edge with t. The bi-directed edges between grids points having capacity 1 represent the $E^{i,j}$.

Turning to geodesic active contour functional (2), it is just the shortest path under the Riemann metric and has the form

$$\psi(C(\tau)) = \oint \sqrt{[x', y']G[x', y']^t}\, d\tau$$

where

$$G = \begin{bmatrix} (\lambda\varphi^0(C(\tau)) + 1)^2 & 0 \\ 0 & (\lambda\varphi^0(C(\tau)) + 1)^2 \end{bmatrix}.$$

Thus by Cauchy-Crofton formula [3, 4] the weight of N-links in GAC functionals is just multiplied by $(\lambda\varphi^0(C(\tau)) + 1)$. Therefore, for functionals (2) and (3) the N-link edges control both similarity and smoothness terms, and T-links just play a role like area constraints or boundary condition.

4.2 Min-Cut Algorithm

The efficiency of the min-cut algorithm is crucial for our applications. In combinatorial optimization there are several polynomial algorithms for min-cut/max-flow. Most of the algorithms fall in one of the following groups: algorithms based on Ford-Fulkerson style "augmenting paths" [13] and Goldberg-Tarjan style "push-relabel" methods [14]. Standard augmenting paths based algorithms, such as Dinic algorithm [12], work by pushing flow along non-saturated paths from the source to the sink until the maximum flow in the graph is achieved. An important fact in Dinic's algorithm is the use of breadth-first search to find the shortest paths from s to t on the residual graph, which significantly improves the theoretical running time. The worst case running time complexity for Dinic's algorithm is $O(mN^2)$ where N is the number of nodes and m is the number of edges in the graph. By clever implementation, "push-relabel" algorithm can be improved to $O(N^3)$. Although the worst complexity of min-cut algorithm is the same with explicit level-set implementation, this worst case is seldom attained in our application. Actually, in all our examples the min-cut algorithm attains the order $O(N^\alpha)$ where $1 \le \alpha \le 2$. Due to their efficiency and reliability we shall use standard push-relabel algorithms in this paper.

5 Implementation Details

There are several implementation issues which need to be pointed out. The first one is the way to tag interior and exterior regions. For the functional (1), we need characteristic functions to represent curves. This is not an issue if curves come from segmentation results since partition is already there. However, most often curve is defined by an ordered set of points. In this case the curve is linearly approximated by a polygon. Then our problem becomes a classic computational geometry problem: determining whether a point is in a polygon. Two rather different methods for solving this problem are available: counting ray crossings and computing "winding" numbers. Because of its simplicity and efficiency we use the ray crossing method. Basic idea of ray crossings is that if a point p is in the polygon a ray from p will intersects polygon boundary odd times otherwise even times. If data set is unordered points it is hard to locate the boundary of the curve by any means and such ray crossing computation is impossible in this scenario. Therefore, we conclude that the curve based functional (1) is not appropriate for curve reconstruction.

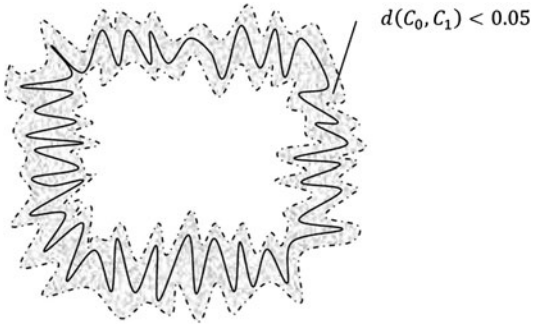


Fig. 3 Dashed is the band boundary; solid line is the given curve. The distance of the point in the shadow region to given curve is less than δ. Candidate curve should be bounded by the shadow region

The second issue is about computing distance function. Since for the functional (2) and (3) we need compute the distance function corresponding to the given curve or data points. Again, if data is defined by an ordered point set then the simplest way to do this is for each grid point computing the distance to all segments and setting the smallest as the final distance. This method does not work for unorganized points and is very slow if points set is large. Generally, we can use a sweeping method [26, 34] to compute distance function. This method has complexity $O(N)$ for uniform grids and $O(N\log N)$ for unstructured grids, where N is the grids number.

Since the global minimum for the GAC model (2) corresponds to a point, direct application of the graph-cuts method will always yield a point which makes no sense in our application. To eliminate this trivial solution, certain Dirichlet boundary conditions [2] need to be imposed. Since desired curve is always near the given curve or data points, it should fall in a narrow band around the given curve or data points as illustrated in Fig. 3. The choice of the bandwidth δ depends on the noise level and the resolution. Based on these observations, [15, 32] used a narrow-band implemen-

tation: (a) If the point is in the narrow band, the weight of the N-link is set as usual; set the weight of the T-link to be zero. (b) If point is not in the narrow band and it is in the interior region enclosed by curve for sure, the point is connected with s and the T-link is set to be a large number. On the other hand, connect points which are certainly outside the curve with t and set T-link to be a large number as well.

Actually, seeking minimum s-t-cut on the graph described above means minimizing the functional (3)

$$E_3(C,\alpha,\beta,\lambda) = \int_{\Omega} \big(\alpha + \beta\varphi^0(C)\big)|\nabla \mathbf{1}_C| d\Omega + \int_{\Omega} \lambda |\mathbf{1}_C - \mathbf{1}_{data}| d\Omega.$$

For curve smoothing problem, $\mathbf{1}_{data}$ in the functional (3) can be computed by ray crossing method, and relative weight function $\lambda(x)$ can be just a constant. For curve reconstruction problems, 1_{data} and $\lambda(x)$ need special treatments, which will be discussed in Sect. 6.

5.1 Corner Preserving Strategy

Next, we present our corner-preserving strategy. To preserve corners the first concern will be designing a good corner detector. Estimating the discrete curvature could yield a good corner detector, but note that the raw data is contaminated by noise, and hence point-wise evaluation or approximation of the curvature is not reliable. Inspired by the neighborhood idea proposed in [31], an *averaging technique* is utilized in this paper to design a robust corner detector.

For an arbitrary polygon $P = \{P_1 P_2 \cdots P_n\}$, denote I_P by its indicator function and A_{x_i} by the angle at point P_i. Then the following observation is true,

$$A_{x_i} = \lim_{t\to 0} \int_{R^2} 2\pi I_P \bullet G_t(x_i, y) dy$$

where $G_t(x_i, y) = \frac{1}{2t}\exp(-\frac{\pi\|y-x_i\|^2}{2t})$. When t is small enough, it is reasonable to use the following approximation

$$A_{x_i} \approx \int_{R^2} 2\pi I_P \bullet G_t(x_i, y) dy.$$

Moreover, for a curve/surface approximated by linear polygon/polyhedra, the following equality holds

$$\sum_i (\pi - A_{x_i}) = 2\pi,$$

therefore, defect angle $\pi - A_{x_i}$ is used to approximate curvature in many cases. notice that

$$2\pi \int_{R^2} (1 - I_P) \bullet G_t(x_i, y) dy = 2\pi - A_{x_i},$$

thus

$$\int_{R^2} (1 - I_P) \bullet G_t(x_i, y) dy - \int_{R^2} I_P \bullet G_t(x_i, y) dy = 1 - \frac{A_{x_i}}{2\pi} - \frac{A_{x_i}}{2\pi}.$$

Therefore the measure of cornerness can be obtained as

$$\tau(x_i) = \frac{1}{2} \int_{R^2} (1 - 2I_P) \bullet G_t(x_i, y) dy.$$

This approximation can overcome the influence of the noises, if variance of the Gaussian is large enough to cover the noise level. In other words, by tuning parameter t, measure $\tau(x_i)$ is able to characterize corners in different scales. The similar ideas can be found in [1, 11]. For a given t, the larger $|\tau(x_i)|$ means the shaper corner in the scale t. However, this formula is computational expensive since when the variance of Gaussian function is large the computation has to be carried on whole domain so as to evaluate the integral accurately. Note that the above quantity actually only measures how different a point is from being on a straight line which has an angle π. Therefore the quantity can also be,

$$\hat{\tau}(x_i) = \frac{|\int_{R^2} G_t(x_i, y) dy|}{\min(g_1, g_2)},$$

where

$$g_1 = \left| \int_{R^2} (1 - I_P) \bullet G_t(x_i, y) dy \right|,$$

and

$$g_2 = \left| \int_{R^2} I_P \bullet G_t(x_i, y) dy \right|.$$

$\hat{\tau}(x_i)$ has value 2 for straight line and becomes larger when the corner becomes shaper. The integration is replaced by summation and R^2 is replaced by the domain of interests Ω,

$$\hat{\tau}_h(x_i) = \frac{|\sum_{y\in\Omega} G_t(x_i, y)\delta y|}{\min(g_1, g_2)},$$

where

$$g_1 = \left| \sum_{y\in\Omega} (1 - I_P) \bullet G_t(x_i, y)\delta y \right|,$$

and

$$g_2 = \left| \sum_{y\in\Omega} I_P \bullet G_t(x_i, y)\delta y \right|.$$

Furthermore, in practical experiments $G_t(x)$ can be simply replaced by 1 and the summation can be only computed in a small neighborhood $\mathcal{N}$ which is determined by the variance or the noise level while numeric errors in numerator

and denominator are somewhat canceled. Thus, in practice the formula may be,

$$\hat{\tau}_h(x_i) \approx \frac{|\sum_{y\in\mathcal{N}} I_P(y)|}{\min(|\sum_{y\in\mathcal{N}}(1-I_P(y))|, |\sum_{y\in\mathcal{N}}(I_P(y))|)}.$$

By this formula $\hat{\tau}_h$ is equal to infinity for the points inside or outside the curve, and therefore to remedy this problem all points not close to the curve take the value 2 the same as the points on straight line. We use the following Algorithm to compute corner measure $\hat{\tau}_h$.

Algorithm (Determination of $\hat{\tau}_h(x)$) For every node x_i in the computational domain Ω:

1. Choose a reference circle $\mathcal{N}(x_i, r)$ centered at x_i with a radius r;
2. Count the number of nodes, $N = N^+ + N^-$, contained inside the reference circle $\mathcal{N}(x_i, r)$ where N^+ denotes the number of nodes with indicator value 1 in $\mathcal{N}(v_i, r)$ and N^- that with indicator value 0. Set $N_{min} = \min\{N^+, N^-\}$;
3. If $N_{min} = N^+$ and $I(x_i) = 1$ *or* $N_{min} = N^-$ and $I(x_i) = 0$, then set $\hat{\tau}_h(x_i) = N/N_{min}$, otherwise, $\hat{\tau}_h(x_i) = 2$;

Once the "cornerness" measure is available the λ in the functional

$$E_4(C) = \int_\Omega \big(\alpha + \beta\varphi^0(C)\big)|\nabla \mathbf{1}_C| d\Omega + \int_\Omega \lambda_{data}(x)|\mathbf{1}_C - \mathbf{1}_{data}|^2 d\Omega,$$

can be set to be large in regions near corners and small in flat regions. Thus, we specify a non-homogeneous "attraction potential" making corners more attractive. In our study, $\lambda(x)$ was taken to be,

$$\lambda(x_i) = \begin{cases} \lambda_1 & \text{if } f(\hat{\tau}_h) \le 0.4\max(f(\hat{\tau}_h)) \\ \lambda_2 & \text{if } f(\hat{\tau}_h) > 0.4\max(f(\hat{\tau}_h)) \end{cases} \tag{6}$$

$$f(s) = e^{w(2-s)^2}, \tag{7}$$

where λ_1, λ_2, and w are some parameters. The reason that $f(s)$ is selected as above is because it changes rapidly in corner regions and is able to make the corner more distinguishable. Once these parameters are selected, the $\lambda(x)$ is determined.

However, the above-mentioned computation relies on the indicator function which is not available for unordered points. In the curve reconstruction application the corner detector should only be based on geometric information residing in the data points. Thus, the approach described below outlines an alternative corner detector. Let $P = \{P_1, P_2, \ldots, P_n\}$ denote the set of unordered points, where order does not imply connectivity. Provided the point set is sampled from a closed curve and dense enough, for each point P_i the neighbourhood $\mathcal{N}(P_i, r)$ should contains some points P_j different from P_i, where $\mathcal{N}f(P_i, r)$ is the circle centered at P_i with radius r. Define the average direction vector as,

$$\mathbf{v}(P_i) = \frac{1}{N} \sum_{P_j\in B(P_i,r) \text{ and } P_j\neq P_i} \frac{P_j - P_i}{|P_j - P_i|}, \tag{8}$$

where N is the number of points in the circle $B(P_i, r)$. When P_i's are on a straight line $\mathbf{v}(P_i)$ will be close to zero vector, and as P_i is a corner point $\mathbf{v}(P_i)$ points to the direction of the angle bisector. Therefore, the cornerness measure for P_i can be defined as the length of the average direction vector,

$$\tau_2(P_i) = |\mathbf{v}(P_i)|. \tag{9}$$

Moreover, if r is large enough this measure can suppress the influences of noises as well. Utilizing this cornerness measure, the corner preserving strategy for reconstruction is either to decease regularity parameter α around detected corners or modify the distance filed induced by points set. There are several ways to implement these ideas, the precise choice of which is immaterial so long as corners are made more attractive (an illustrative example is shown in Sect. 7).

6 Application to Curve Reconstruction from Scattered Points

As mentioned above, reconstruction curve from an unordered set of points is an interesting and challenging problem, and there is no well defined, unique solution for this problem. To construct a curve that is a good approximation of the data set, a reconstruction procedure should be able to deal with complicated topologies and geometries, as well as with noise and nonuniformity of sampling. This problem is closely related to curve smoothing, since both problems rely on the definition of meaningful distance measures. The functional (1) measures distance between two curves by area difference. As we pointed out before this is a proper distance measurement for the case of curve smoothing but in the reconstruction scenario an indicator function becomes meaningless since we can not completely mark the interior and exterior regions. In this case, some "induced" topological information can be made available for the regions far from the points, and near the point sets only the geometrical or distance information is readily usable. Hence the functional (2) which only relies on distance information can readily handle the curve reconstruction problem. However, as discussed in Sect. 2, the modified functional (3) is more stable than (2), and can be minimized by graph-cuts techniques. To address reconstruction problem by functional (3), our first task will

be to properly define the $\mathbf{1}_{data}$ indicator function. This will be done by defining a "narrow band" containing the data points

$$S_{band} = \{x | \varphi^0(x) \leq c\} \tag{10}$$

where $\varphi^0(x)$ is the unsigned distance function induced by the point set. By the definition of the narrow band, the weight function λ is set to

$$\lambda(x) = \begin{cases} 0 & \text{if } x \in S_{band}, \\ \infty & \text{if } x \notin S_{band}. \end{cases} \tag{11}$$

In practice, the bandwidth c depends on sampling density and noise level. For example, provided the sample points are quite sparse, c should be relatively large so that the narrow-band will be a connected watertight region. Generally, the set S_{band} separates R^2 into one outside domain and possible several inside domains. $\mathbf{1}_{data}$ will be set to 0 outside and to 1 for the points in the inside domains. For the points belonging to S_{band}, the indicator function can be set to an arbitrary value since $\lambda(x)$ is zero for them, and the term $\lambda|\mathbf{1}_C - \mathbf{1}_{data}|$ will add nothing to the energy cost. Essentially, this is the narrow band implementation used in various applications.

As described in Sect. 5.1, based on proposed corner detectors (9) it is also possible to develop corner preserving strategy for curve reconstruction. A simple way is to decrease regularity parameter α around the detected corners. Because τ_2 in (9) is defined on the data points not on grids points, one needs impose corner information on grid points by locating the nearest grid points x_i. Thus, $\alpha(x)$ may have the form,

$$\alpha(x_i) = \begin{cases} \alpha_1 & \text{if } x_i \text{ is in the corner regions} \\ \alpha_2 & \text{otherwise} \end{cases} \tag{12}$$

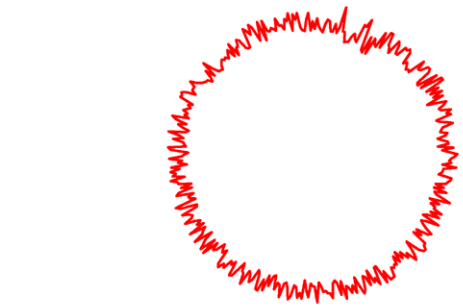
where α_1 and α_2 $(\alpha_1 < \alpha_2)$ are some constants to be selected. Another way is to modify the distance field around detected corners. For example, the distance field can be generated by a function $(\frac{\text{dist}}{c})^n$ in a neighbourhood of the detected corners, where c is the constant used to define $\lambda(x)$ in (11). In both cases, the solution curve will prefer to pass through detected corners in order to decrease energy.

7 Numerical Experiments and Results

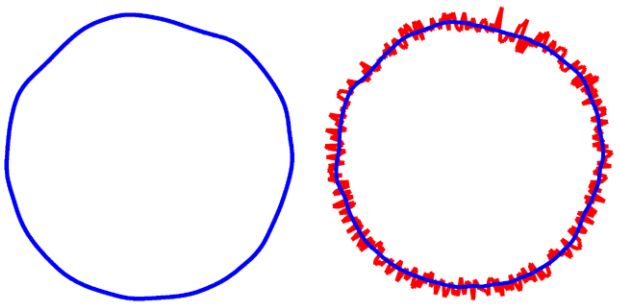
All the experiments reported herein were performed on a PC with Intel Xeon CPU of 3.33 GHz and 4 GB memory. To do graph cuts we use the push relabel algorithm, see [14].

The first example is a noisy circle as shown in Fig. 4(a). The curve is smoothed using the functional (1)

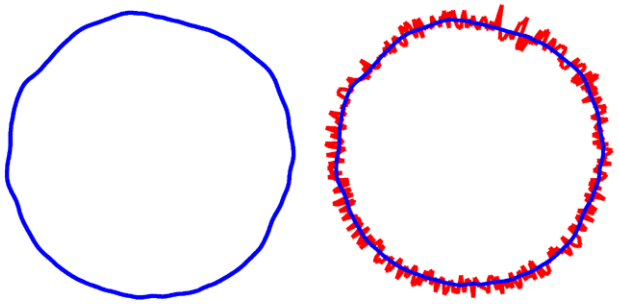
$$E_1(\phi, \lambda) = \int_\Omega |\nabla \mathbf{1}_C| d\Omega + \lambda \int_\Omega |\mathbf{1}_C - \mathbf{1}_{C_0}| d\Omega,$$



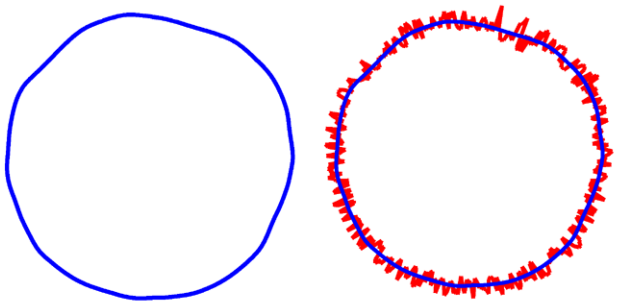
(a) Noisy Circle



(b) The result minimizing functional 1 and the comparison of the noisy curve and smoothed curve

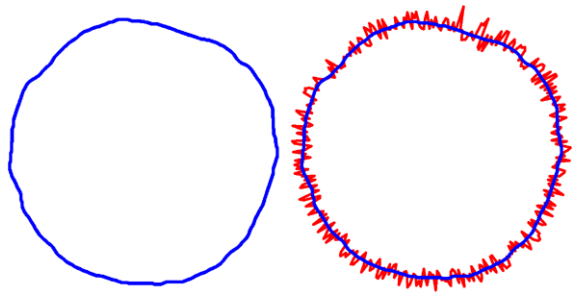


(c) The result minimizing functional 2 and the comparison of the noisy curve and smoothed curve

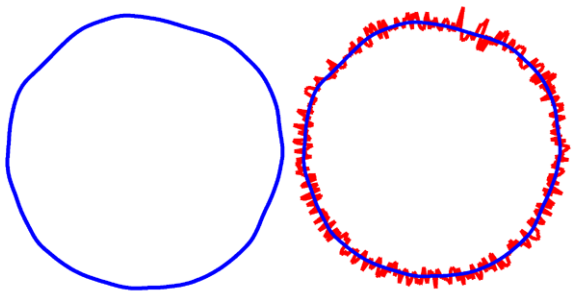


(d) The result minimizing combined functional 3 and the comparison of the noisy curve and smoothed curve

Fig. 4 The resulting curves smoothed for the same data and three different functionals. (**b**) The smoothed curve with parameters $\lambda = 0.05$ and $\alpha = 1$ in functional (1). (**c**) The smoothed curve with parameters $\beta = 5$ and $\alpha = 1$ in the functional (2). (**d**) The smoothed curve with parameters $\lambda = 0.05$, $\beta = 3$ and $\alpha = 1$ in functional (3)



(a) Smoothed Circle produced by the graph-cut method



(b) Smoothed Circle produced by the level set method

Fig. 5 The first column contains the smoothed curve and the second column includes the mixture of the smoothed curves and noisy curves; parameters in functional (3) are $\alpha = 1$, $\beta = 3$ and $\lambda = 0.05$ and 32 neighbor system is used

functional (2)

$$E_2(C, \alpha, \beta) = \int_0^1 (\alpha + \beta\varphi^0(C(\tau)))|C_\tau| d\tau,$$

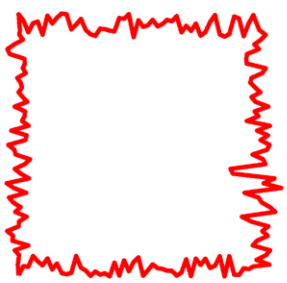
and functional (3)

$$E_3(C, \alpha, \beta, \lambda) = \int_\Omega \big(\alpha + \beta\varphi^0(C)\big)|\nabla \mathbf{1}_C| d\Omega + \int_\Omega \lambda|\mathbf{1}_C - \mathbf{1}_{data}| d\Omega.$$

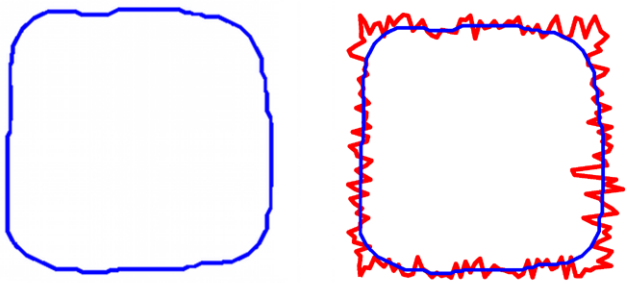
First, three functionals are minimizing by the level set method on a 400×400 uniform mesh. Here, forward difference is used for time discretization, time step being 1×10^{-6} and reinitialization is performed every 50 steps. Figure 4(b) shows the smoothed circle obtained by minimizing functional (1) and the comparison with the noisy circle. Figure 4(c) shows the results minimizing functional (2) and Fig. 4(d) presents the results minimizing the functional (3). It can be observed that the result in Fig. 4(b) is slightly smoother than the result in Fig. 4(c), and the result in Fig. 4(d) is more like the result in Fig. 4(b) because of the choices of the parameter λ and β. For comparison the results of the graph cut method and the level set method are presented together in Fig. 5, in which both smoothed curves are obtained by minimizing the functional (3) with the same parameters as the one in Fig. 4(d) and 32-neighbourhood system is used for the graph cut method. It can be observed that under the same resolution, the level set method presents better result than the graph cuts method because of the metrication errors of graph-discretization. However, the computation time of the level set method is much longer than the graph cut method, as can be clearly seen from the statistics of CPU time listed in Table 1.

Table 1 CPU time (in seconds) of the level set method and the graph cut method for the noisy circle example

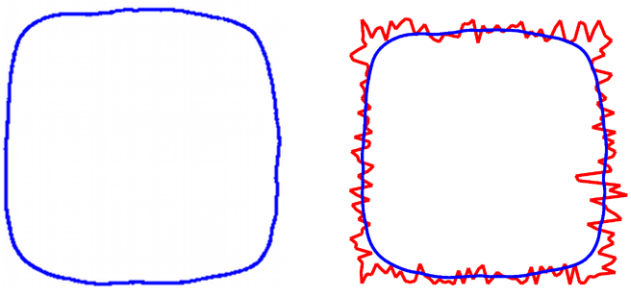
	200 Resolution	400 Resolution
Level set method	117.12 s	1326.23 s
Graph cuts	1.9 s	7.9 s



(a) Noisy rectangle



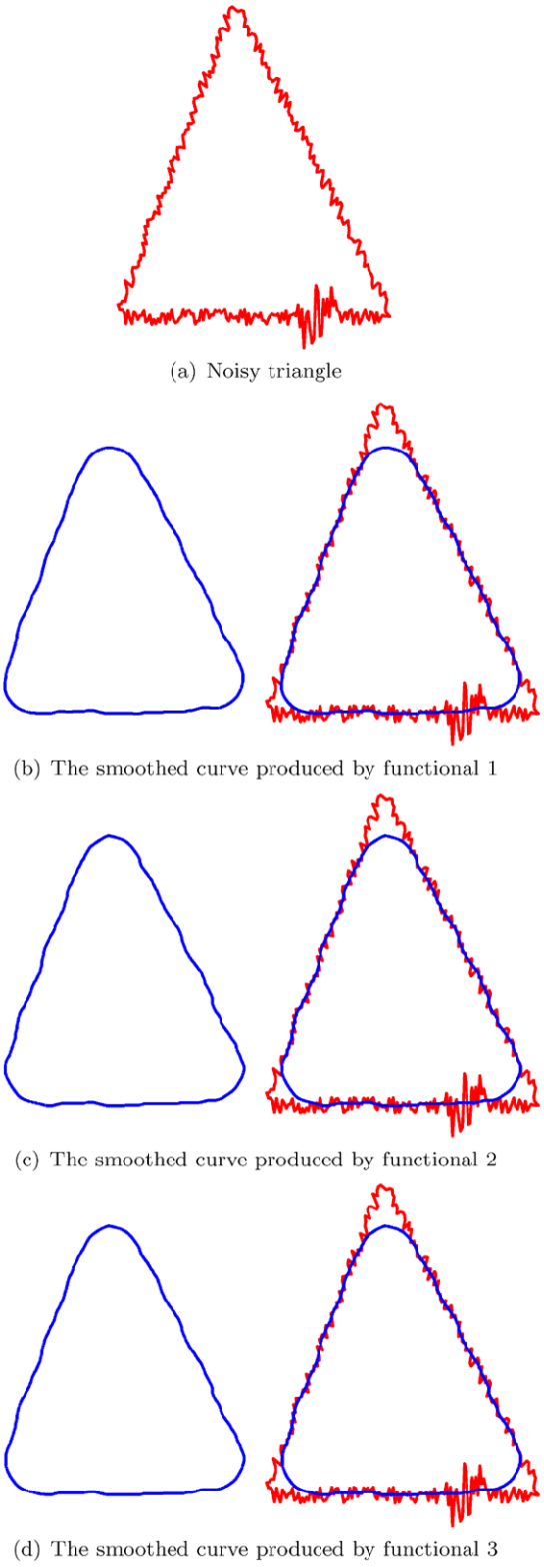
(b) The result minimizing functional (3)



(c) The result minimizing functional (3)

Fig. 6 In (**b**) and (**c**), the first column contains the smoothed curve and the second column includes the mixture of the smoothed curves and noisy curves. Parameters in functional (3) are $\lambda = 0.05$ and $\beta = 8$ and 64 neighbor system is used

Figure 6 gives an example of a noisy square which is also processed by graph-cuts and level set methods. In this example the 64-neighbourhood system is used for graph



(a) Noisy triangle

(b) The smoothed curve produced by functional 1

(c) The smoothed curve produced by functional 2

(d) The smoothed curve produced by functional 3

Fig. 7 A noisy triangle example. From the second row to the last row, the first column contains the smoothed curves and the second column includes the mixture of the smoothed curves and noisy curves; parameters in the functional (1) are $\lambda = 0.03$ and $\alpha = 1$; parameters in the functional (2) are $\beta = 4$ and $\alpha = 1$ while narrow band has width 0.06. In model (3) $\lambda = 0.02$, $\beta = 4$, and $\alpha = 1$. 64 neighbor system and 800×800 resolution are used in all methods

Table 2 CPU time of graph-cuts results of two functionals with different neighborhoods. Resolution is 800×800 and similarity parameters in two model (1) and (2) are 0.02 and 4 respectively while regularity parameter is $\alpha = 1$ for both functionals

Neighborhoods	Time	
	Functional 1	Functional 2
4	11.32 s	9.96 s
16	28.73 s	23.11 s
64	86.31 s	85.16 s

cuts method and the metrication errors reduce considerably. Since the result obtained by graph cut method is actually only the labellings and the curve is obtained by connecting discrete girds, the resulting curve in Fig. 6(b) is still not as smooth as the one in Fig. 6(c) obtained by the level set method, which is the interpolation of zero level-set of the signed distance function.

The third example considers a noisy triangle as shown in Fig. 7, where the results obtained by minimizing three different functionals are illustrated. In this example the graph cuts method is applied on 800×800 grids with a 64 neighbor system. Figure 7(b) presents the result of the functional (1) and Fig. 7(c) gives the result of the functional (2). It can be observed that two smoothed curves have almost the same quality. As discussed before, to solve the functional (2) by graph cut, the narrow band implementation is needed. The width of the band is selected to be 0.2 in this example. Table 2 lists the CPU times consumed by minimizing functionals (1) and (2) with different neighbor systems, from which we can see that to minimize the functional (2) is slightly faster than the functional (1). Figure 7(d) shows the smoothed curve produced by the functional (3), which quite resembles the result in Fig. 7(c) because here the parameter β is dominated. However, to solve the functional (3) there is no need for narrow band implementation and the CPU time is almost as same as the functional (1).

In order to quantitatively compare above results, we use the following way to measure the difference of the smoothed curve and the original curve. We compute

$$D_{C_0}(C) = \frac{\sum |\mathbf{1}_C - \mathbf{1}_{C_0}|}{\sum \mathbf{1}_{C_0}} \tag{13}$$

where C_0 is the original curve and C is a noisy curve or smoothed curve. This normalized symmetric area difference measures the relative difference of C and C_0, and obviously the smaller it is, the closer two curves are. Statistics listed in Table 3 are normalized symmetric area differences computed for the examples in Figs. 5 and 7, in which the noisy curves and the smoothed curves are compared with the original curves. To save space, the original curves are omitted. In Table 3, the differences between three functionals are quantitatively compared, where the Circle example is processed

Table 3 The normalized symmetric area difference. Two examples are used to study the difference between three functionals. In each example, the statistics are computed for the noisy curve, the smoothed curve minimizing Functional 1, Functional 2 and Functional 3, respectively. The parameters are chosen as before

	Noisy	Fun. 1	Fun. 2	Fun. 3
Circle	0.0723	0.0259	0.0261	0.0258
Triangle	0.0412	0.0358	0.0395	0.0343

Table 4 The normalized symmetric area difference. Two examples are used to study the difference between Level-set and Graph-cuts implementation. In each example, the statistics are computed for the results of Level-set method and Graph-cuts method respectively. The parameters are chosen as same as before

	Noisy	Level-set	Graph-cuts
Circle	0.0723	0.0259	0.0407
Rectangle	0.0886	0.0434	0.0512

via the level-set implementation while the results of the Triangle example are obtained with the graph-cuts implementation. From Table 3, several conclusions can be drawn. First, according to the proposed similarity measure the smoothed curves are closer to the original curves than the noisy curves. Second, the results of Functional 3 are slightly better than those of Functional 1 and Functional 2. Also, from the quantitative comparison one can observe that the smoothing effects of the Triangle example are not as good as the Circle example. This is because the corners of the Triangle are necessarily rounded by the smoothing process. Therefore, in order to obtain good smoothing effects while also preserving the sharp corners, data-dependent parameters will have to be used. We will further investigate the corner preserving strategies with some examples and quantitatively study them later. In the same fashion, we compare the results of the level-set method and the graph-cuts method. The examples in Figs. 5 and 6 are used and the corresponding statistics are listed in Table 4. It can be seen that, under the same resolution, the results of the graph-cuts method is worse than those of the level-set method due to the metric errors and discretization artifacts.

In the above examples, by choosing appropriate parameters the three functionals provided the similar results. However, when the geometry of curves become complicated, the differences between the optimal results with respect to functional (1) and functional (2) can be quite significant. Therefore, we next consider an example of a curve which is the digitized outline of a cat with lots of zig-zags, as shown in Fig. 8(a). The second row in Fig. 8 shows the results of level-set implementations while the third row presents the results of graph-cuts implementations of the optimizations of the functionals we consider. Figure 8(b) and (c) presents the results of functional (2) with different curve initialization. In Fig. 8(b), the initial curve was a small circle inside the cat, and after 200,000 iterations the evolution is stuck at a local minimum, as shown. In Fig. 8(c), the initial solution is chosen as a large circle around the cat and after 500,000 iterations the evolution stops. It is obvious that with a large circle as the initial guess the final solution is much better than taking the small circle as the initial solution. However, both of them missed the concave part of the curve. Figure 8(d) presents the result of functional (1). Because of the existence of the extra fidelity term, functional (1) is not so sensitive to the initial solutions. Moreover, the fidelity term plays the role of an external driving force which did push the solution into the concave part. There is no initialization issue for graph-cuts implementation, however using graph-cut to minimize functional (2), one has to resort to narrowband implementations. Figure 8(e) and (f) presents the results of functional (2), both of which are solved by graph-cuts, for different widths of the narrowband. With a very large bandwidth, the result (Fig. 8(e)) misses all elongated parts. Com-

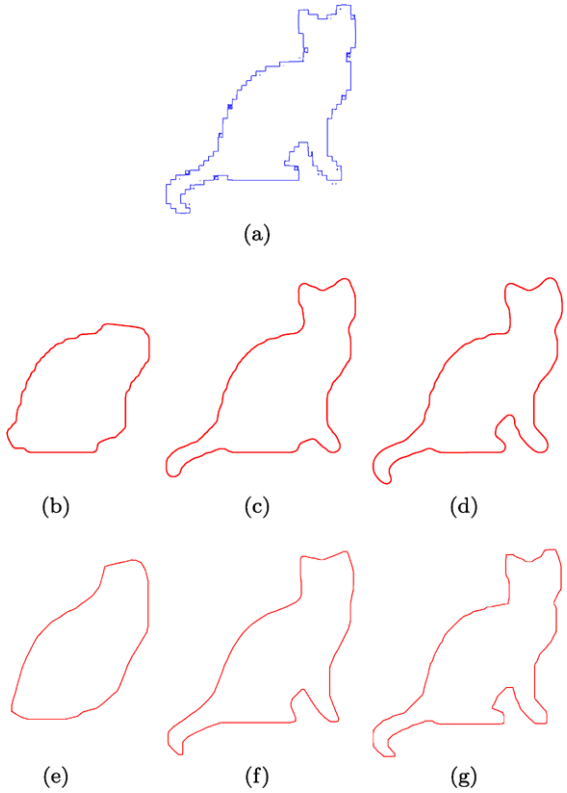


Fig. 8 A cat with a lot of zig-zags. (**a**) The noisy curve; (**b**) the level-set solution of functional (2) with a small circle as the initial solution; (**c**) the level-set solution of functional (2) with a large circle as the initial solution; (**d**) the level-set solution of functional (1); (**e**) the graph-cuts solution of functional (2) with a large narrowband; (**f**) the graph-cuts solution of functional (2) with a relatively small narrowband; (**g**) the graph-cuts solution of functional (1). 400×400 resolution is used for level-set implementations and 800×800 resolution is used for graph-cuts implementations

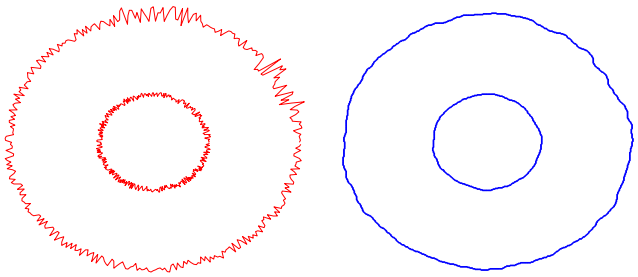


Fig. 9 A noisy annulus example. The first figure presents the noisy curve and the second figure shows the result minimizing model (3); 64 neighborhoods are used and similarity parameter $\lambda = 0.06$, $\beta = 4$ and $\alpha = 1$

pared to Fig. 8(e), the result in Fig. 8(f) is much better due to a smaller bandwidth. Figure 8(g) gives the curve minimizing functional (1), and no narrowband needed for this functional. For functional (1) both the level-set implementation and the graph-cut implementation can yield good results for complex geometries. For functional (2), the results of the level-set implementation depend on initialization, and a good initial guess not only accelerates the algorithm but also enables it to yield the desired results; the graph-cuts implementation relies on a proper choice of the narrowband, and too thick a narrowband may cause the loses of the elongated parts while too thin a narrowband may not provide smooth curves. As discussed before, although functional (1) performs better than functional (2) in the curve smoothing examples, it is not applicable to the curve reconstruction tasks. The combination of these two functionals, functional (3) and functional 4 avoid their individual limitations and have wider applicability.

The next example is a noisy annulus. The result is shown in Fig. 9. To do this example with the functional 2 one can let points between two circles be connected with source and both boundary of the domain and the center of the inner circle be connected with sink. In our implementation the T-links are automatically determined by region growing method.

The next examples consider noisy curves with more complicated geometry. Hereafter all examples will be implemented based on the functional 3 and $\alpha = 1$ unless specified otherwise. It can be observed from Fig. 10 that the smaller the similarity parameter is chosen, the smoother curve is obtained but also the more details are lost. We remark that with different parameters the computation costs of the algorithm are somewhat different. Table 5 gives the CPU time consumed during smoothing Fig. 10 with different parameters and narrow bands. Here we only use 64 neighborhoods within narrow band and 4 neighborhoods outside the narrow band. It can be observed from Table 5 that this adaptive implementation considerably reduced the computation cost. However, narrow band should be wide enough so that at least solution curve is included.

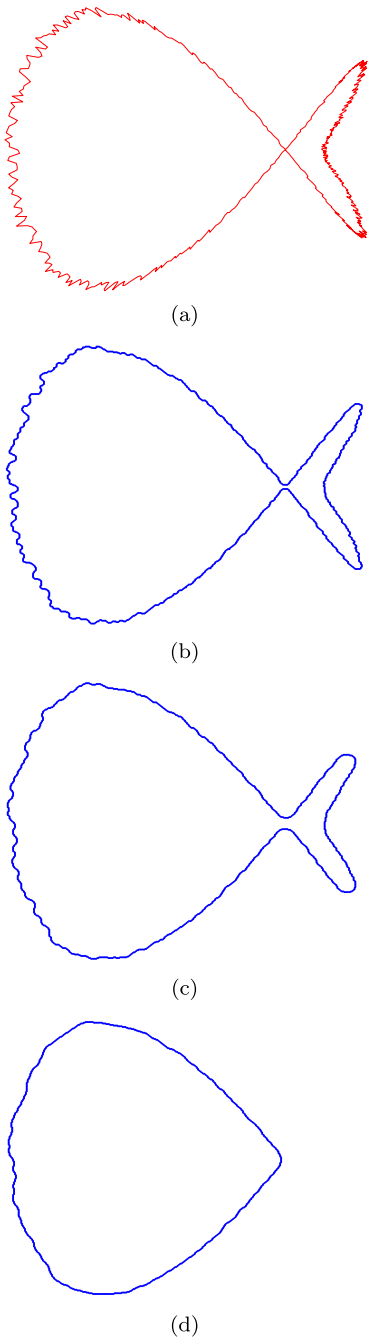


Fig. 10 An example with complicated geometry; 64 neighborhoods are used and $\beta = 0.0$ in all results but with different parameter λ. (**a**) The noisy curve; (**b**) the curve smoothed with $\lambda = 0.2$; (**c**) the curve smoothed with $\lambda = 0.1$; (**d**) the curve smoothed with $\lambda = 0.05$

The example in Fig. 10 is formed by a circle intersecting a noisy rectangle. This is a more challenging example. From Fig. 10 we can see that circle and rectangle separate when the similarity parameter is decreased and since the intersection regions for the two shapes are very small, it is hard to determine the correct local topology.

Table 5 CPU time with different parameters under 800 × 800 resolution. 64 neighborhood is chosen within narrow band

λ	Time	
	Bandwidth <1	Bandwidth <0.5
0.2	73.11 s	35.45 s
0.1	76.12 s	41.86 s
0.05	111.43 s	58.24 s

The "Squirrel surrounded by grass" example in Fig. 12 tests the smoothing of curves resulting from image segmentation. The direct image segmentation results have a lot of small features and the boundary of the object often has zigzags. It can be observed that in the last picture of Fig. 12, all the insignificant features are removed by our method.

The main disadvantage of the functional (1) and (2) is that resulting curves may lose meaningful corners. Therefore, the functional (4) is used to handle the examples with sharp corners,

$$E_4(C) = \int_{\Omega} \big(\alpha + \beta\varphi^0(C)\big)|\nabla \mathbf{1}_C| d\Omega + \int_{\Omega} \lambda_{data}(x)|\mathbf{1}_C - \mathbf{1}_{data}| d\Omega,$$

where $\lambda(x)$ is computed as described in Sect. 5.1. The example in Fig. 13 is a noisy eight-sided star, which has obvious sharp corners. Figure 13(c) shows the smoothed curve with constant similarity parameters $\lambda = 0.5$ and $\beta = 5$. Although all corners are preserved, the curve is not smooth enough. By decreasing λ to 0.03 the curve becomes much smoother but also loses all corners as shown in Fig. 13(d). However, using $\lambda(x)$ computed by the method described in Sect. 5.1 the result presented in Fig. 13(e) is smooth enough as well as preserves all sharp corners. The contour of the function $\lambda(x)$ is also illustrated in Fig. 13(f), which clearly demonstrates that all corners are correctly identified. Here we use the normalized symmetric area difference to study the proposed corner preserving strategy. Let C_0 denote the original curve shown in Fig. 13(a) and C_1, C_2, C_3, and C_4 denote the curves shown in Fig. 13(b)–(e) respectively. The normalized area differences for them are $D_{C_0}(C_1) = 0.0641$, $D_{C_0}(C_2) = 0.0542$, $D_{C_0}(C_3) = 0.1640$, $D_{C_0}(C_4) = 0.0231$ respectively. C_1 is the noisy curve, $D_{C_0}(C_1)$ actually measures the noise level and it is 0.0641 for this example. C_2 is the smoothed curve as shown in Fig. 13(c), from which it can be seen that C_2 is a reasonable approximation of the original curve, and this is also justified by the fact $D_{C_0}(C_2) = 0.0542 < 0.0641$. Although C_3 in Fig. 13(d) is very smooth, it is not a good approximation since visually it loses some important features of the original curve and quantitatively it not close to the original curve $D_{C_0}(C_3) = 0.1640$. C_4 is an excellent approximation of the original curve, which can be seen from the figure clearly and is proven by the normalized area difference as well, $D_{C_0}(C_4) = 0.0231$, the smallest value among these curves. All statistics are listed in Table 6. This example shows that our corner preserving strategy indeed works well and a relatively complex example will be presented next.

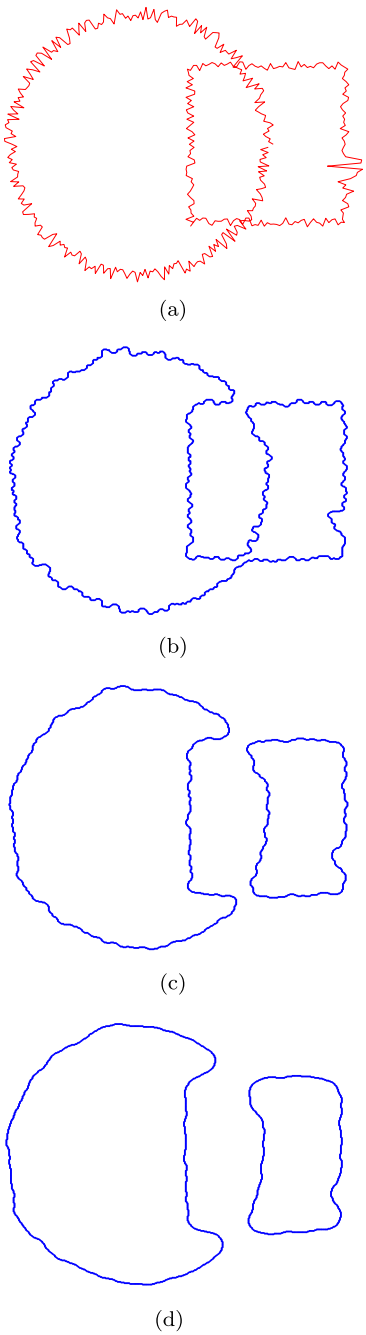


Fig. 11 An example formed by a noisy circle intersected a noisy rectangle; 64 neighborhoods are used and $\beta = 0.0$ in all results but with different similarity parameters λ. (**a**) The noisy curve; (**b**) the result with $\lambda = 0.2$; (**c**) the result with similarity $\lambda = 0.1$; (**d**) the result with $\lambda = 0.05$

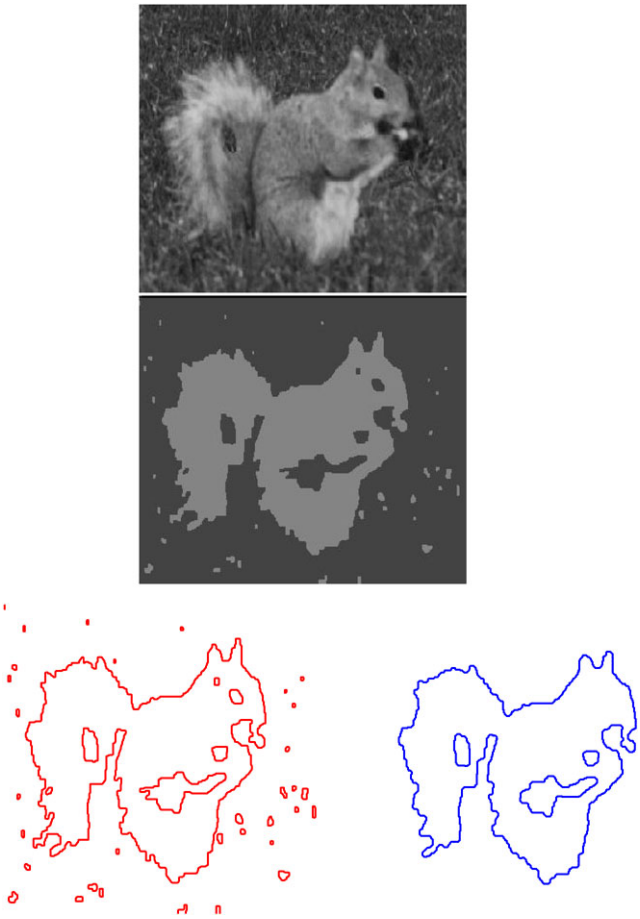


Fig. 12 Image segmentation result. (**a**) Original image; (**b**) segmentation result; (**c**) boundary curve of the segmentation; (**d**) smooth curve processed by our method

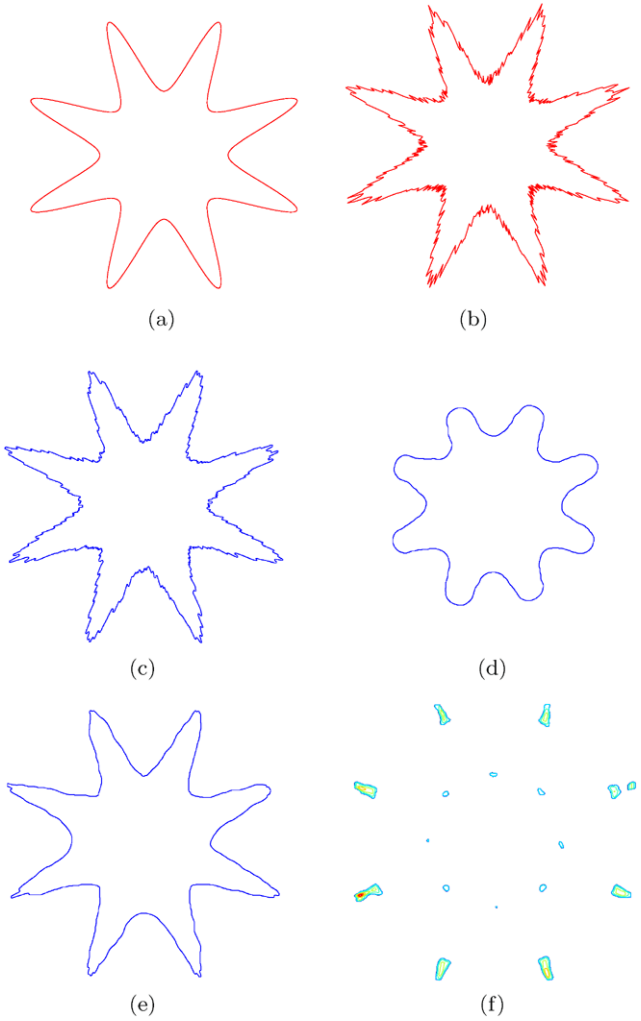


Fig. 13 Noisy eight-sided star example. (**a**) The original curve; (**b**) the noisy curve; (**c**) the smoothed curve produced with parameters $\lambda = 0.5$ and $\beta = 5$; (**d**) the smoothed curve produced with parameters $\lambda = 0.03$ and $\beta = 5$; (**e**) the smoothed curve produced with varying $\lambda(x)$ and $\beta = 5$, where $\lambda(x)$ is computed with $\lambda_1 = 0.5$, $\lambda_2 = 0.5$ in (12), $w = 4$ in (7) and $r = 0.1$ for reference circle; (**f**) the contour of the function $\lambda(x)$ in (**d**)

Table 6 Comparison of the normalized area difference

Curves	C_1	C_2	C_3	C_4
Area Difference	0.0641	0.0542	0.1640	0.0231

The example in Fig. 14 is a result from image segmentation, a logo of "NTU". Figure 14(a) and (b) presents segmentation result and the corresponding boundary curve. Figure 14(c) shows the smoothed curve with constant similarity parameters $\lambda = 0.03$ and $\beta = 5$. It can be observed that not only the corners are missed but also the elongated parts in letter "N" and "U". Figure 14(d) presents the result with the varying similarity function, which is smooth and preserves most important features. Again, one can see the correctly identified important features from the contour of the function $\lambda(x)$ shown in Fig. 13(e).

The next three examples are the results of curve reconstruction, where bandwidth is chosen as $c = 0.1$ and the similarity parameter $\beta = 20$. Figure 15 is the curve recovered from points which have strong noise at certain parts. We can observe that the noisy clouds are gracefully ignored by our method. Figure 16 depicts a cloud of points outlining an elephant shape. The point cloud is very noisy but our method recovers a meaningful result.

Figure 17 shows a cloud of points sampled from a Chinese character which has obvious corners. In this example, the proposed corner preserving strategy is clearly demonstrated. Figure 17(a) gives the set of noisy points. Figure 17(b) illustrates the reconstructed curve with constant regularity parameter $\alpha = 1$ while Fig. 17(c) presents the reconstructed curve with regularity parameter $\alpha = 0.01$ around corner regions and $\alpha = 1$ otherwise. Figure 17(d) gives the comparison of two reconstructed curves, clearly manifesting that the curve reconstructed by the proposed corner preserving strategy has sharper corners and better visual effects. Figure 17(e) shows the contour of the proposed

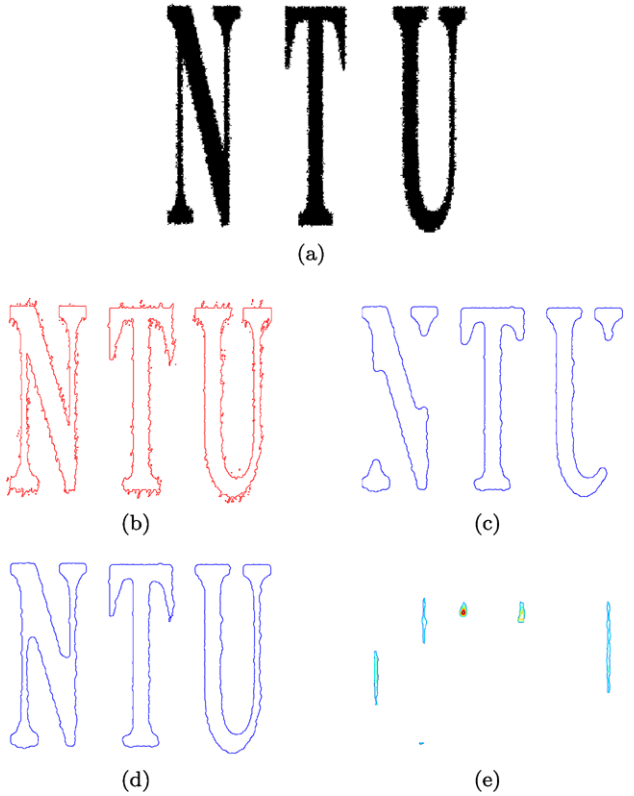


Fig. 14 Segmentation result of an image containing words "NTU". (**a**) The segmentation result; (**b**) the noisy boundaries; (**c**) the smoothed curve produced with parameters $\lambda = 0.03$ and $\beta = 5$; (**d**) the smoothed curve produced with varying $\lambda(x)$ and $\beta = 5$, where $\lambda(x)$ is computed with $\lambda_1 = 0.03$, $\lambda_2 = 0.5$ in (12), $w = 4$ in (7) and $r = 0.1$ for the reference circle; (**e**) the contour of the function $\lambda(x)$ in (**d**)

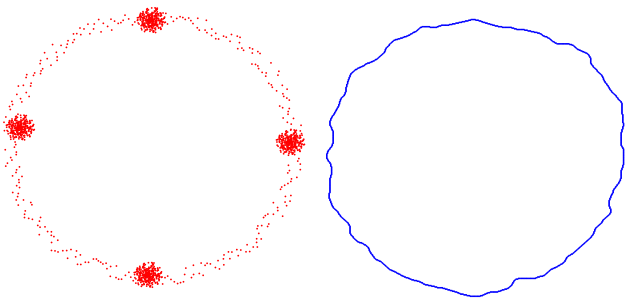


Fig. 15 Curve reconstruction from noisy points data. The points in the first figure are the noisy data and the curve shown in the second figure is the reconstruction result

corner detector (9), in which the locations of all corners are correctly identified.

Furthermore, by choosing different distance fields, the variational model proposed in this paper can actually deal with a large range of applications. The example shown in Fig. 18 demonstrates that our method can readily produce the shortest path within a bounded region between two curves. In this example, the distance field is generated by

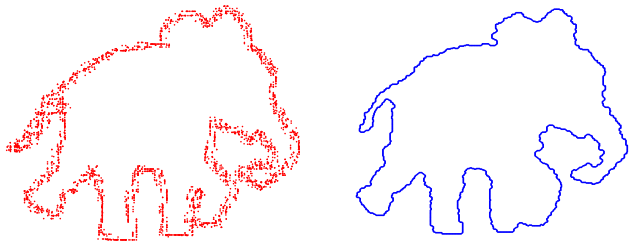


Fig. 16 Curve reconstruction from noisy points data. The points in the first figure are the noisy data and the curve shown in the second figure is the reconstruction result

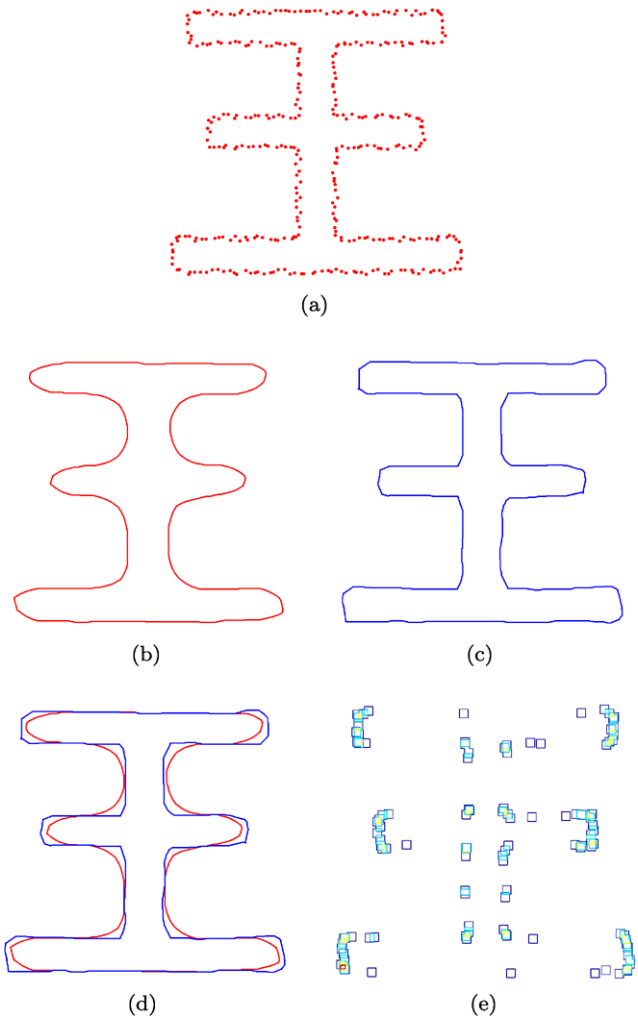
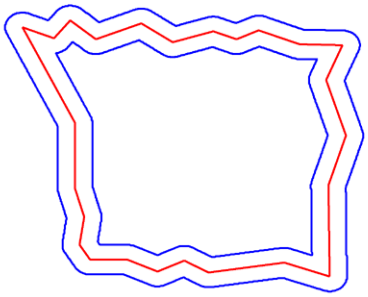


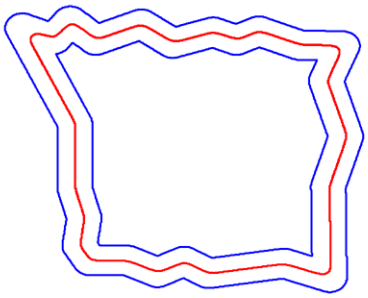
Fig. 17 Curve reconstruction from points sampled from a Chinese character. (**b**) The reconstructed curve with constant regularity parameter; (**c**) the reconstructed curve with varying regularity parameter; (**d**) mixture of the curves in (**c**) and (**d**); (**e**) the contour of the corner detector function

the formula

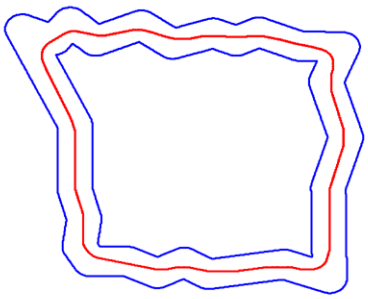
$$D = \left(\frac{\text{dist}}{\mathbf{B}}\right)^n$$



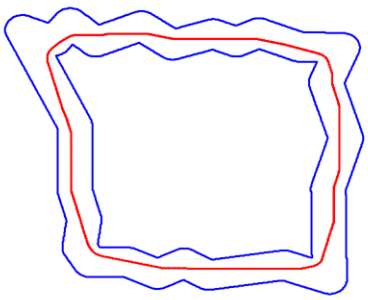
(a) Original Curve within the Band:$dist \leq 0.05$



(b)



(c)



(d)

Fig. 18 Shortest path within a bounded region. Distance function is $(\frac{\text{dist}}{\mathbf{B}})^n$. (**a**) A bounded region generated by a curve with width 0.05; (**b**) the solution curve with $\lambda = 0, \beta = 20, n = 3$; (**c**) the solution curve with $\lambda = 0, \beta = 4, n = 3$; (**d**) the solution curve with $\lambda = 0, \beta = 4, n = 6$

where **B** is the width of the band which is here set to 0.05, dist is the unsigned distance function and n is an integer constant. We see that bigger n yields smoother curves. Figures 19 and 20 are two examples which show that our method can determine the shortest polygonal path in a bounded region defined between two nested polygons.

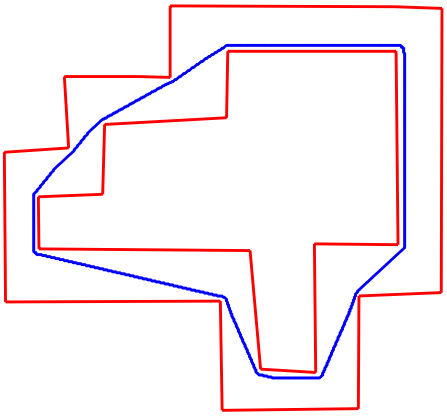


Fig. 19 Shortest polygonal path in a bounded region, obtained by "variational smoothing"

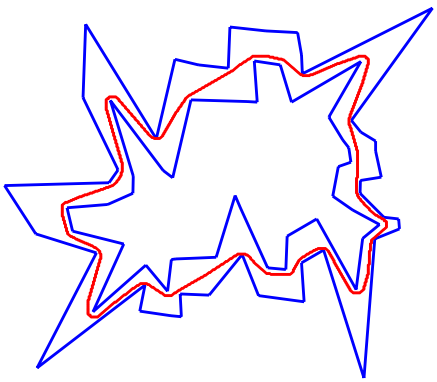


Fig. 20 Shortest path in a more complicated region between two polygons

As an interesting generalization, the distance field naturally solves heteroscedastic probabilistic curve reconstruction problems [18], as described below. Consider the samples of a planar curve corrupted by independent additive Gaussian noises whose covariance matrices vary according to the location of the sample points in space as follows:

$$\mathbf{v} = \mathbf{v}_0 + n,$$

where $n \sim \mathcal{N}(0, \Sigma_{\mathbf{v}_0})$ is an independent Gaussian noise vector with zero mean and point-specific covariance ($\mathbf{v} = (x, y)$, $\mathbf{v}_0 = (x_0, y_0)$). Therefore, for a given sample point $\mathbf{v}_0$, the position of point $\mathbf{v}$ is a random vector with Gaussian distribution $\mathcal{N}(\mathbf{v}_0, \Sigma_{\mathbf{v}_0})$. Provided the covariance matrix is given for each point, the question is to find the original point (x_0, y_0) given that (x, y) is observed. Thus, the following conditional density function is of interest,

$$p(\mathbf{v}_0|\mathbf{v}) = \frac{p(\mathbf{v}|\mathbf{v}_0)p(\mathbf{v}_0)}{\int_{R^2} p(\mathbf{v}|\mathbf{v}_0)p(\mathbf{v}_0)dx_0dy_0} \tag{14}$$

describing the distribution of the curve sample given the observation point $\mathbf{v}$. Noticed that, for each possible (x_0, y_0) and a fixed sample point (x, y), the density function $p(\mathbf{v}|\mathbf{v}_0)$ is a function of (x_0, y_0) proportional to

$$\frac{1}{\sqrt{(2\pi)|\Sigma_{\mathbf{v}_0}|^{1/2}}} \exp\left(-\frac{1}{2}(\mathbf{v}-\mathbf{v}_0)'\Sigma_{\mathbf{v}_0}^{-1}(\mathbf{v}-\mathbf{v}_0)\right).$$

Without prior knowledge about the original curve, it is reasonable to assume that $p(\mathbf{v}_0)$ in the formula (14) is a uniform distribution in a bounded region, which means each point in the region is equally likely to be a point on the curve that is sampled. Under this assumption, the conditional density function generated by an observed point can be computed by formula (14), in which $p(\mathbf{v}|\mathbf{v}_0)$ is a function of (x_0, y_0), $p(\mathbf{v}_0)$ a constant, and $\int_{R^2} p(\mathbf{v}|\mathbf{v}_0)p(\mathbf{v}_0)dx_0dy_0$ a scaling.

After the conditional density function is determined for each sample point, the influence potential of the points $\{\mathbf{v}_i\}$ can be defined as

$$A(x, y) := \frac{1}{T}\sum_i p((x, y)|\mathbf{v}_i), \tag{15}$$

where $\mathbf{v}_i$'s are the T sample points. This density function describes how likely points in the plane are the original samples of a curve given points we observe. Therefore, the problem of finding the original curve becomes a process of seeking curves having the largest cumulative probability, which is equivalent to the following optimization problem

$$\max_{C(x,y)} \oint_C A(x, y)ds. \tag{16}$$

By adding a constant and changing the sign of the function, the maximization problem (16) is changed to the minimization problem (17)

$$\min_{C(x,y)} \oint_C \psi ds, \tag{17}$$

where

$$\psi = \max(A(x, y)) - A(x, y).$$

Since ψ is non-negative, functional $\oint_C \psi ds$ looks exactly like the GAC functional discussed before, in spite of the fact that the "distance" field is now defined by the probability density function instead of geometric distance. In contrast to the simple geometric approach, the probability density function is able to incorporate prior knowledge on the noise distribution. For example, assuming that $\mathbf{v}_0$ uniformly distributes in a band around the sample points, the resulting density function will be consistent with narrowband graph-cuts implementation. Furthermore if the location of the point $\mathbf{v}_0$ is roughly known, one can assume $\mathbf{v}_0$ is subject to a Normal

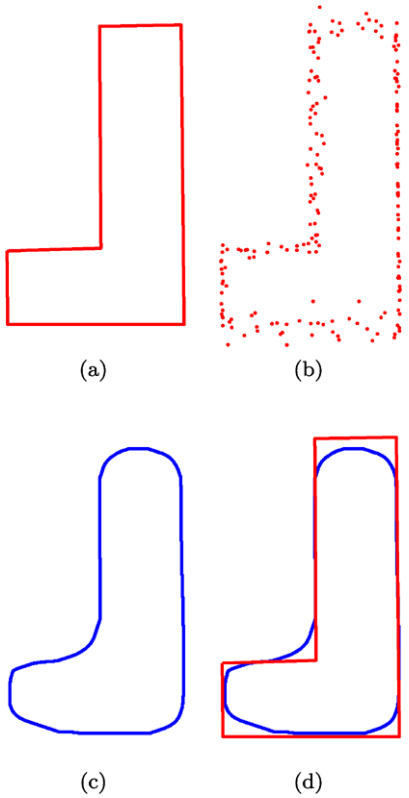


Fig. 21 An example of L-shape polygon. (**a**) The original curve; (**b**) the noisy sample; (**c**) the smoothed curve maximizing sum probability; (**d**) the comparison of the curve in (**c**) and the original curve

distribution with $\mathbf{v}_0$ as mean. We next present several examples to demonstrate the capability of the above proposed framework to nicely handle heteroscedastic cases.

The examples considered assume that each sample point generates a probability or influence field, and the whole influence field is of form (15). In the sequel, a influence function defined in (15) will be called "sum probability". In the examples considered the 2-D Gaussian distributions around each point are given by

$$\frac{1}{\sqrt{2\pi|\Sigma_{\mathbf{v}_0}|}} \exp\left(-\frac{1}{2}(\mathbf{v}-\mathbf{v}_0)^T\Sigma_{\mathbf{v}_0}^{-1}(\mathbf{v}-\mathbf{v}_0)\right)$$

where $\mathbf{v}$ is the observed point and $\mathbf{v}_0$ is the grid point. The covariance matrix Σ is given by

$$\Sigma_{\mathbf{v}_0} = \begin{bmatrix} \sigma_x^2 & \rho\sigma_x\sigma_y \\ \rho\sigma_x\sigma_y & \sigma_y^2 \end{bmatrix}.$$

Our first example shown in Fig. 21 is an L-shape polygon. In this example the following spatially varying covariance is used to generate the noisy sample points $\sigma_x = 0.05 + |\sin(5\pi x)|/30$, $\sigma_y = 0.05 + |\cos(5\pi y)|/30$, $\rho = (x^2 + y^2)/4$. The first row of Fig. 21 includes the original curve and the noisy samples. It can be seen that the noise

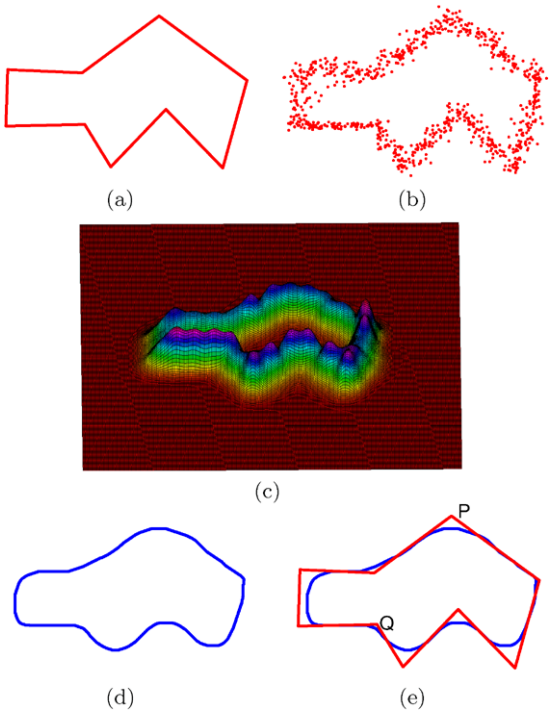


Fig. 22 A polygon example. (**a**) The original curve; (**b**) the noisy samples; (**c**) the influence field generated by the noisy samples; (**d**) the smoothed curve; (**e**) the comparison of the smoothed curve and original curve

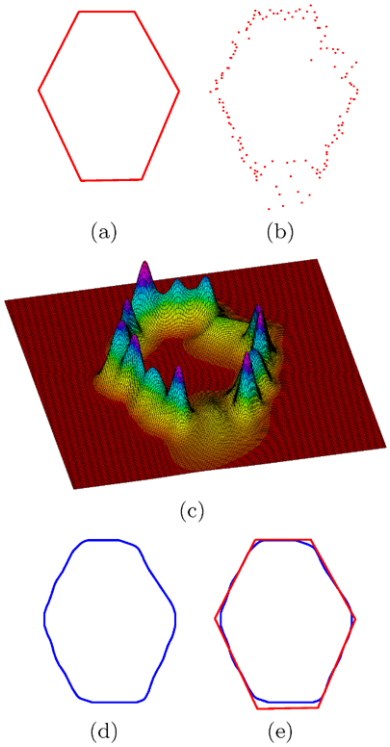


Fig. 23 An example of six-sided polygon. (**a**) The original curve; (**b**) the noisy samples, which have relative large noise level on both upper left and bottom sides; (**c**) the influence field generated by the noisy samples; (**d**) the smoothed curve; (**e**) the comparison of the smoothed curve and original curve

varies at different locations. The second row shows the curve obtained by maximizing the sum probability.

The example in Fig. 22 is of a curve with very noisy samples. The first row of the Fig. 22 contains the original curve and the noisy samples. The influence field generated can be observed in Fig. 22(c). The result shown in Fig. 22(d) and (e) show that the noise effect is removed. It is clear that the variance of Gaussian distribution drastically changes from point to point. This means that we trust more the data-points in certain regions. For example in Fig. 22(d) we can observe that at point P the curve is far from the corner because of large variance and at point Q the curve is close to the corner because of small variance. We note that instead of using homogeneous and isotropic distance fields, inhomogeneous and anisotropic distance fields allow us much more flexibility in modelling the effects of noise and in incorporating prior knowledge.

The last example shown in Fig. 23 is a six-sided polygon. Figure 23(a) gives the original curve, and Fig. 23(b) shows the noisy samples, which has relative large noise level on both upper left and bottom sides. Figure 23(c) presents the influence field generated by noisy samples, which has two evident dents on upper left and bottom sides on account of the large variance of the sample points. Therefore, the regions with large noise level are less trusted, which is the effect desired. Figure 23(d) and (e) presents the smoothed curve and the comparison with the original curve.

8 Concluding Remarks

In this paper we survey general variational frameworks for curve smoothing and reconstruction problems. Compared with the functional (2), the functional (1) is insensitive to the initial guess and can be minimized by graph-cuts techniques yielding global minima. The distance measure used in the functional (1) is not suitable to curve reconstruction problem, to which the distance-field based functional (2) is readily applicable. By combing the merits of functionals (1) and (2), the functional (3) is able to gracefully handle curve reconstruction and smoothing within the same process. To preserve corners the similarity parameters or regularity parameters can be made dependent on "cornerness measures", two of which are proposed and were successfully tested in this paper. By introducing a probability distance field, functional (3) is also able to deal with applications involving heteroscedastic noises. Various examples were presented to demonstrate the flexibility, effectiveness and robustness of the proposed joint functional. Exporting some of the ideas developed in this paper to surface smoothing and reconstruction problems will be one of our future projects. A thorough study of corner preserving regularizations is also an interesting subject for future investigations.

References

1. Alvarez, L., Morales, F.: Affine morphological multiscale analysis of corners and multiple junctions. Int. J. Comput. Vis. **25**, 95–107 (1997)
2. Boykov, Y., Jolly, M.-P.: Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images. In: IEEE International Conference on Computer Vision, vol. 1, pp. 105–112 (2001)
3. Boykov, Y., Kolmogorov, V.: Computing geodesic and minimal surfaces via graph cuts. In: International Conference on Computer Vision, vol. 1, pp. 26–33 (2003)
4. Boykov, Y., Kolmogorov, V.: An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. IEEE Trans. Pattern Anal. Mach. Intell. **26**, 1124–1137 (2004)
5. Caselles, V., Kimmel, R., Sapiro, G.: Geodesic active contours. Int. J. Comput. Vis. **22**, 61–79 (1997)
6. Chan, T.F., Golub, G.H., Mulet, P.: A nonlinear primal-dual method for total variation-based image restoration. SIAM J. Sci. Comput. **20**, 1964–1977 (1999)
7. Chan, T.F., Esedoḡlu, S., Nikolova, M.: Algorithms for finding global minimizers of image segmentation and denoising models. SIAM J. Appl. Math. **66**, 1632–1648 (2006)
8. Cohen, L.D., Cohen, I.: Finite element methods for active contour models and balloons for 2D and 3D images. IEEE Trans. Pattern Anal. Mach. Intell. **15**, 1131–1147 (1993)
9. Cohen, L.D., Kimmel, R.: Global minimum for active contour models: a minimal path approach. Int. J. Comput. Vis. **24**, 57–78 (1997)
10. Cremers, D., Soatto, S.: A pseudo-distance for shape priors in level set segmentation. In: 2nd IEEE Workshop on Variational, Geometric and Level Set Methods in Computer Vision, pp. 169–176 (2003)
11. Deriche, R., Giraudon, G.: A computational approach for corner and vertex detection. Int. J. Comput. Vis. **10**, 101–124 (1993)
12. Dinic, E.A.: Algorithm for solution of a problem of maximum flow in networks with power estimation. Sov. Math. Dokl. **11**, 1277–1280 (1970)
13. Ford, L., Fulkerson, D.: Flows in Networks. Princeton University Press, Princeton (1962)
14. Goldberg, A.V., Tarjan, R.E.: A new approach to the maximum-flow problem. J. Assoc. Comput. Mach. **35**, 921–940 (1988)
15. Hornung, A., Kobbelt, L.: Robust reconstruction of watertight 3D models from non-uniformly sampled point clouds without normal information. In: Eurographics Symposium on Geometry Processing, pp. 41–50 (2006)
16. Kimmel, R.: Numerical Geometry of Images: Theory, Algorithms, and Applications. Springer, Berlin (2004)
17. Kimmel, R., Bruckstein, A.M.: Regularized Laplacian zero crossings as optimal edge integrators. Int. J. Comput. Vis. **5**, 225–243 (2003)
18. Kiryati, N., Bruckstein, A.M.: Heteroscedastic hough transform (HtHT): an efficient method for robust line fitting in the 'errors in the variables' problem. Comput. Vis. Image Underst. **78**, 69–83 (2000)
19. Kolmogorov, V., Boykov, Y.: What metrics can be approximated by geo-cuts, or global optimization of length/area and flux. In: IEEE International Conference on Computer Vision, pp. 564–571 (2005)
20. Kolmogorov, V., Zabih, R.: What energy can be minimized via graph cuts? IEEE Trans. Pattern Anal. Mach. Intell. **26**, 147–159 (2004)
21. Krishnan, D., Lin, P., Tai, X.-C.: An efficient operator splitting method for noise removal in images. Commun. Comput. Phys. **1**, 847–858 (2006)
22. Lempitsky, V., Boykov, Y.: Global optimization for shape fitting. In: IEEE Conference on Computer Vision and Pattern Recognition, pp. 17–22 (2007)
23. Lombaert, H., Sun, Y., Grady, L., Xu, C.: A multilevel banded graph cuts method for fast image segmentation. In: IEEE International Conference on Computer Vision, vol. 1, pp. 259–265 (2005)
24. Osher, S., Sethian, J.A.: Fronts propagating with curvature dependent speed: algorithms based on Hamilton-Jacobi formulations. J. Comput. Phys. **79**, 12–49 (1988)
25. Paragios, N., Rousson, M., Ramesh, V.: Matching distance functions: a shape-to-area variational approach for global-to-local registration. In: Lecture Notes in Computer Science, 7th European Conference on Computer Vision, pp. 775–789 (2002)
26. Qian, J., Zhang, Y.-T., Zhao, H.-K.: Fast sweeping methods for eikonal equations on triangular meshes. SIAM J. Numer. Anal. **45**, 83–107 (2007)
27. Rudin, L.I., Osher, S., Fatemi, E.: Nonlinear total variation based noise removal algorithms. Physica D **60**, 259–268 (1992)
28. Shu, C.W., Osher, S.: Efficient implementation of essentially non-oscillatory shock capturing schemes. J. Comput. Phys. **77**, 439–471 (1988)
29. Shu, C.W., Osher, S.: Efficient implementation of essentially non-oscillatory shock capturing schemes II. J. Comput. Phys. **83**, 32–78 (1989)
30. Wang, Y., Song, S., Tan, Z., Wang, D.: Adaptive variational curve smoothing based on level set method. J. Comput. Phys. **228**, 6333–6348 (2009)
31. Wang, J., Ju, L., Wang, X.: An edge-weighted centroidal Voronoi tessellation model for image segmentation. IEEE Trans. Image Process. **18**, 1844–1858 (2009)
32. Xu, N., Bansal, R., Ahuja, N.: Object Segmentation Using Graph Cuts Based Active Contours, vol. 2, pp. 46–53 (2003)
33. Zach, C., Pock, T., Bischof, H.: A globally optimal algorithm for robust TV-L^1 range image integration. In: IEEE International Conference on Computer Vision, pp. 14–21 (2007)
34. Zhao, H.: A fast sweeping method for eikonal equations. Math. Comput. **74**, 603–627 (2005)
35. Zhao, H., Osher, S., Merriman, B., Kang, M.: Implicit and non-parametric shape reconstruction from unorganized points using variational level set method. Comput. Vis. Image Underst. **80**, 295–319 (2000)