Computer Vision and Image Understanding 116 (2012) 1178–1194

Contents lists available at SciVerse ScienceDirect

Computer Vision and Image Understanding

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

The active geometric shape model: A new robust deformable shape model and its applications ☆

Quan Wang *, Kim L. Boyer

Signal Analysis and Machine Perception Laboratory, Department of Electrical, Computer, and Systems Engineering, Rensselaer Polytechnic Institute, Troy, NY 12180, USA

ARTICLE INFO

Article history:
Received 9 January 2012
Accepted 24 August 2012
Available online 3 September 2012

Keywords:
Active geometric shape model
Parametric equation
Cubic spline contour
Graph cuts segmentation

ABSTRACT

We present a novel approach for fitting a geometric shape in images. Similar to active shape models and active contours, a force field is used in our approach. But the object to be detected is described with a geometric shape, represented by parametric equations. Our model associates each parameter of this geometric shape with a combination of integrals (summations in the discrete case) of the force field along the contour. By iteratively updating the shape parameters according to these integrals, we are able to find the optimal fit of the shape in the image. In this paper, we first explore simple cases such as fitting a line, circle, ellipse or cubic spline contour using this approach. Then we employ this technique to detect the cross-sections of subarachnoid spaces containing cerebrospinal fluid (CSF) in phase-contrast magnetic resonance (PC-MR) images, where the object of interest can be described by a distorted ellipse. The detection results can be further used by an *s*–*t* graph cut to generate a segmentation of the CSF structure. We demonstrate that, given a properly configured geometric shape model and force field, this approach is robust to noise and defects (disconnections and non-uniform contrast) in the image. By using a geometric shape model, this approach does not rely on large training datasets, and requires no manual labeling of the training images as is needed when using point distribution models.

1. Introduction

We consider two classes of problems: (1) detect and segment an object in the image, where we have some prior knowledge about the shape of the object and (2) fit a geometric shape to the image. These two problems become very similar when the shape of the object to be detected can be described in a geometric form. In this section, we first review the model-based image analysis methods, which are developed to solve the first class of problems. Then we talk about existing techniques for solving the second class of problems. We will show how our work combines the two endeavors to solve a more difficult problem: detect and segment an object described by a geometric shape without using training data.

1.1. Model-based image analysis

Model-based image analysis is a popular approach to extract high-level information from images. By incorporating prior knowledge, which can be learned from training data or hypotheses of the specific problem, model-based methods offer good performance and robustness in a number of domains.

For most model-based methods, the two major steps are to train the parameters of the model as part of the system design and then, in operation, to fit the model to images. Examples include the well known active shape models (ASMs) [1] and active appearance models (AAMs) [2] proposed by Cootes, where the prior knowledge is obtained statistically by performing a principal component analysis (PCA) on the covariance matrix of a point distribution model (PDM). Another popular model is Kass' active contour model (snakes) [3], which fits smooth closed curves to images using an energy minimization criterion. By defining an internal energy and an external energy, which correspond to the smoothness and data faithfulness of the model respectively, and by fitting the model iteratively, the resulting curve is expected to achieve an optimal balance between smoothness and accuracy. In 1997, Xu and Prince improved the active contour model by applying a new external force, known as the gradient vector flow (GVF) [4]. In 2006, Gotardo et al. introduced the interframe smoothness energy and the Fourier domain interpretation [5] into the active contour model, thus generalizing this model for time lapse image sequence analysis [6]. These models are widely used in various applications, such as image segmentation [7], object tracking [8], face recognition [9], neuron tracing [10], and the detection of pathologies in medical images [6].

However, adding prior knowledge to these models has always been a difficult and interesting issue. Statistical models, such as

☆ This paper has been recommended for acceptance by Albert C.S. Chung.
* Corresponding author.
E-mail addresses: wangq10@rpi.edu (Q. Wang), kim@ecse.rpi.edu (K.L. Boyer).

ASMs and AAMs, are based on point distributions, and therefore are not suitable for shapes without salient feature points such as angles, inflection points, or points of salient curvature because of the inaccuracy of manual annotation. Besides, when a large training dataset is not available, statistical models should not be used in any case. Because the original active contour models only use smoothness and data faithfulness to fit the model, it is essential to incorporate prior knowledge in the form of initial conditions, data constraints, or constraints on model shape parameters into the model fitting procedures [11].

Our active geometric shape model (AGSM) combines geometric shapes with deformability. In our model, the shapes of interest are usually represented by parametric equations. Fitting the shape to the image then becomes a problem of finding the set of parameters of the shape that best fits the image data. By taking different line integrals of a force field generated from the image, we associate each shape parameter with a force or torque. Then we establish our iterative update criteria for each shape parameter according to the forces or torques. By adding constraints to the shape parameters, the resulting shape will always be acceptable, thus avoiding unexpected shapes at all times. With a carefully selected force field and form of integrals, this model is robust to severe noise and defects such as disconnections and non-uniform contrast in the image. In this paper, the active geometric shape model is used to detect the CSF structures in PC-MR images, where the shape has no salient points, little training data is available, and the images are usually very noisy. We represent the shape of interest as a distorted ellipse having five parameters. By adjusting the parameters according to the force field, our model converges accurately to the CSF structure.

1.2. Geometric shape fitting

There are various ways to fit a geometric shape to data. The best known methods are least squares and weighted least squares [12]. However, it is generally difficult to analytically solve the least squares fit for a complicated shape, and it is not suitable for shape detection in the image, where the data cannot be directly used as the input to the least squares problem.

Another famous technique is the Hough transform, which solves the geometric shape fitting problem by a voting procedure in the parameter space [13]. However, for a geometric shape with N parameters, the voting procedure is performed in an N-dimensional parameter space, and the vote of each data is an $(N-1)$-dimensional manifold, which could be very complicated itself. By making use of the directional information associated with edges, Ballard's Hough transform for analytic curves reduced the dimension of each vote to $N-2$ [14]. But the cost of the Hough transform still increases exponentially as the number of parameters of the shape increases. Ballard also generalized the Hough transform for arbitrary shapes represented by a set of boundary points $\{\mathbf{x}_B\}$ relative to some reference origin $\mathbf{y}$ [14]. This technique uses a fixed 4-dimensional parameter space $\{x_c, y_c, s, \theta\}$ for any shape, where (x_c, y_c) is the reference origin of the shape, s is the scale factor, and θ is the rotation factor. However, the generalized Hough transform still performs a brute-force search in the scaling and rotation space, and fails to consider other deformation patterns apart from size and orientation.

In our active geometric shape model, each parameter is updated at each iteration. As the number of parameters increases, the cost of our model only increases linearly. Besides, unlike Hough transform, our model relies only on the force field derived from the image, and this image can be either gray-level or binary.

2. Methods

If the object of interest can be described by a geometric shape, then the features of the object can be described by the parameters of the geometric shape model. For example, a straight line can be described by two parameters: the distance s from the line to the origin, and the orientation θ of the vector from the origin to the closest point on this line. A circle can be described by three parameters: the radius r and the coordinates of the center (x_c, y_c). For an ellipse in standard orientation (aligned with the coordinate axes), there are four parameters: the semi-major axis length a, the semi-minor axis length b, and the coordinates of the center (x_c, y_c). And for an ellipse with arbitrary orientation, the orientation ϕ is another parameter. In this section, we will first review the concept of *force field*, which is widely used for deformable models. Next we will discuss the active geometric shape models for the above mentioned four cases. Then we will propose two more complicated shapes: the distorted ellipse and the cubic spline contour.

2.1. Deformable models and force field

Both active shape models and active contour models are applied by configuring the model according to a certain vector force field over the image, with each vector pointing to edges or ridges of regions of interest in the image. In active shape models, each model point (also called *landmark*) moves along the force field in each iteration [1]. In active contour models, the snakes move along the gradient of the external energy, which is equivalent to a force field [3]. A gray-level image $I(x, y)$ can be viewed as a function of continuous position variables (x, y). To lead the deformable model toward step edges, the external energy can be defined as [15]

$$E_{\text{ext}}^{(1)}(x, y) = -\|\nabla I(x, y)\|^2, \tag{1}$$

$$E_{\text{ext}}^{(2)}(x, y) = -\|\nabla (G_\sigma(x, y) * I(x, y))\|^2, \tag{2}$$

where ∇ is the gradient operator and $G_\sigma(x, y)$ is the 2-dimensional Gaussian function with standard deviation σ. In (2), the image is blurred using a Gaussian function to reduce the noise and increase the capture range of the force field. If we require the deformable model to converge to the gray-level maxima, then appropriate definitions of external energy include

$$E_{\text{ext}}^{(3)}(x, y) = -I(x, y), \tag{3}$$

$$E_{\text{ext}}^{(4)}(x, y) = -G_\sigma(x, y) * I(x, y). \tag{4}$$

Gradient vector flow, which is computed as a diffusion of the gradient vectors of the negative external energy derived from the image, is another form of force field, and is expected to have much larger capture range than other force fields [4]. The gradient vector field is defined as the vector field $\mathbf{v}(x, y) = [u(x, y), v(x, y)]$ that minimizes the energy functional

$$\mathcal{E} = \iint \left(\mu(u_x^2 + u_y^2 + v_x^2 + v_y^2) + \|\nabla f\|^2 \|\mathbf{v} - \nabla f\|^2 \right) \mathrm{d}x\mathrm{d}y. \tag{5}$$

Here $f(x, y)$ is the negative of the external energy derived from the image $I(x, y)$, defined as

$$f(x, y) = -E_{\text{ext}}^{(i)}(x, y), \tag{6}$$

where $i = 1, 2, 3$ or 4. In this paper, we use gradient vector flow $\mathbf{v}(x,y)$ as our force field, and set $f(x,y) = -E_{\text{ext}}^{(4)}(x,y)$ in (5). This force field can be solved using calculus of variations, and can be computed by numerical iteration [15].

2.2. Fitting a line

The equation of a straight line can be written as [16]

$$x\cos\theta + y\sin\theta - s = 0, \tag{7}$$

where s is the distance from the line to the origin and θ is the orientation of the vector from the origin to the closest point on this line. To update these two parameters iteratively, we define two integrals (we are going to use the word *integral* as equivalent to *summation* for the discrete case in the present context) that correspond to the geometric interpretation of the parameters. Assume we are working on an image, and the discrete pixels on the line are represented as (x_i, y_i) where $1 \leqslant i \leqslant N$, and $x_i \leqslant x_j$ if $i < j$. Instead of $\mathbf{v}(x, y) = [u(x, y), v(x, y)]$, which is defined as the solution of the optimization problem (5), from now on we use the notation $\mathbf{F}(x, y) = [F_x(x, y), F_y(x, y)]$ for the force field vector at pixel (x, y).

2.2.1. Updating s

For the distance parameter s, we can define an average normal force, which is the integral of the projection of the force field onto the normal vector $[\cos\theta, \sin\theta]^\top$ along the line normalized by the length N:

$$F_n = \frac{1}{N}\sum_{i=1}^{N}\mathbf{F}(x_i, y_i)\cdot\begin{bmatrix}\cos\theta\\ \sin\theta\end{bmatrix}. \tag{8}$$

Here the "·" represents the dot product of two vectors. If F_n is larger than 0, this average normal force tends to push the line further from the origin. If F_n is smaller than 0, it tends to pull the line towards the origin. Defining δs as the step we use to update s in each iteration, and t_s as the tolerance for F_n (a tolerance is a threshold that if the magnitude of force is smaller than it, the parameter will not be updated), we use this criterion to update s:

$$\begin{cases} s_{\text{new}} = s + \delta s & \text{if } F_n > t_s \\ s_{\text{new}} = s - \delta s & \text{if } F_n < -t_s \end{cases}. \tag{9}$$

2.2.2. Updating θ

For the orientation parameter θ, instead of force, we define an average torque which tends to rotate the line around a fulcrum. If we pick (x_k, y_k) as our fulcrum, which is on the current estimate of the line, we can define the average torque T_k as

$$T_k = \frac{1}{N^2}\sum_{i=1}^{N}\text{sgn}(k-i)d_{ik}\mathbf{F}(x_i, y_i)\cdot\begin{bmatrix}\cos\theta\\ \sin\theta\end{bmatrix}, \tag{10}$$

where

$$d_{ik} = \sqrt{(x_i - x_k)^2 + (y_i - y_k)^2} \tag{11}$$

is the distance between (x_i, y_i) and (x_k, y_k), and $\text{sgn}(\cdot)$ is the sign function. Here the normalization factor is $1/N^2$ instead of $1/N$ because when the number of pixels on the line increases, the average distance from (x_i, y_i) to the fulcrum also increases. The fulcrum should be selected to maximize the absolute value of the torque:

$$\tilde{k} = \arg\max_k |T_k|, \tag{12}$$

$$T = T_{\tilde{k}}. \tag{13}$$

Define $\delta\theta$ as the step we use to update θ in each iteration, and t_θ as the tolerance for T. If $\theta \in [0, \pi)$, the criterion for updating θ is

$$\begin{cases} \theta_{\text{new}} = \theta - \delta\theta & \text{if } T > t_\theta \\ \theta_{\text{new}} = \theta + \delta\theta & \text{if } T < -t_\theta \end{cases}. \tag{14}$$

If $\theta \in [\pi, 2\pi)$, just substitute $\delta\theta$ by $-\delta\theta$ in (14). Since (x_i, y_i) is defined such that $x_i \leqslant x_j$ if $i < j$, the sign function in (10) guarantees that the line rotates in the correct direction. Because we are rotating the line around point $(x_{\tilde{k}}, y_{\tilde{k}})$, we need to adjust s according to θ_{new}, which is

$$s_{\text{new}} = x_{\tilde{k}}\cos\theta_{\text{new}} + y_{\tilde{k}}\sin\theta_{\text{new}}. \tag{15}$$

This will ensure that after updating parameters according to the torque about $(x_{\tilde{k}}, y_{\tilde{k}})$, the line still passes through $(x_{\tilde{k}}, y_{\tilde{k}})$. Note that s is updated twice in each iteration: first updated according to the average normal force using (9); then updated as a fix for the updated θ using (15).

Because there are two parameters to update, in each iteration we first compute all forces and torques using current parameters, and then update all parameters using these forces and torques. In this way, the order in which we update different parameters does not affect the updating results. We will follow this philosophy for all other shapes in the following sections.

2.3. Fitting a circle

For a circle, we use the parametric equations

$$\begin{cases} x = x_c + r\cos\theta \\ y = y_c + r\sin\theta \end{cases}, \tag{16}$$

where (x_c, y_c) is the center of the circle, r is the radius, and θ varies in $[0, 2\pi)$. Assume we are working on an image, and each pixel on the circle corresponds to a θ_i value where $1 \leqslant i \leqslant N$, and $\theta_i < \theta_j$ if $i < j$. Let the coordinates of the pixel corresponding to θ_i be (x_i, y_i), or more specifically,

$$\begin{cases} x_i = x_c + r\cos\theta_i \\ y_i = y_c + r\sin\theta_i \end{cases}. \tag{17}$$

2.3.1. Updating x_c and y_c

To move the circle to the expected position, we update its center coordinates x_c and y_c at each iteration. We define four forces, corresponding to horizontal, vertical, diagonal and anti-diagonal movement respectively:

$$F_{ch} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{F}(x_i, y_i)\cdot[1, 0]^\top, \tag{18}$$

$$F_{cv} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{F}(x_i, y_i)\cdot[0, 1]^\top, \tag{19}$$

$$F_{cd} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{F}(x_i, y_i)\cdot\left[\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}\right]^\top, \tag{20}$$

$$F_{ca} = \frac{1}{N}\sum_{i=1}^{N}\mathbf{F}(x_i, y_i)\cdot\left[-\frac{\sqrt{2}}{2}, \frac{\sqrt{2}}{2}\right]^\top. \tag{21}$$

Assuming that δx_c and δy_c are the steps to update x_c and y_c in each iteration, and t_c is the tolerance for the four forces, our criteria for updating x_c and y_c are

$$\begin{cases} x_{c\text{new}} = x_c + \delta x_c & \text{if } F_{ch} > t_c \text{ or } F_{cd} > t_c \text{ or } F_{ca} < -t_c \\ x_{c\text{new}} = x_c - \delta x_c & \text{if } F_{ch} < -t_c \text{ or } F_{cd} < -t_c \text{ or } F_{ca} > t_c \end{cases}, \tag{22}$$

$$\begin{cases} y_{c\text{new}} = y_c + \delta y_c & \text{if } F_{cv} > t_c \text{ or } F_{cd} > t_c \text{ or } F_{ca} > t_c \\ y_{c\text{new}} = y_c - \delta y_c & \text{if } F_{cv} < -t_c \text{ or } F_{cd} < -t_c \text{ or } F_{ca} < -t_c \end{cases}. \tag{23}$$

2.3.2. Updating r

The radius r of the circle corresponds to the average normal force, which tends to expand or shrink the circle along the radial direction. The average normal force is defined as

$$F_n = \frac{1}{N}\sum_{i=1}^{N}\mathbf{F}(x_i, y_i)\cdot\begin{bmatrix}\cos\theta_i\\ \sin\theta_i\end{bmatrix}. \tag{24}$$

If δr is the step we use to update r in each iteration, and t_r is the tolerance for F_n, then the updating criterion is

$$\begin{cases} r_{\text{new}} = r + \delta r & \text{if } F_n > t_r \\ r_{\text{new}} = r - \delta r & \text{if } F_n < -t_r \end{cases}. \tag{25}$$

2.4. Fitting an ellipse in standard orientation

The parametric equations of an ellipse whose major and minor axes are parallel to the x-axis and y-axis, respectively, are

$$\begin{cases} x = x_c + a\cos\theta \\ y = y_c + b\sin\theta \end{cases}, \tag{26}$$

where a is the semi-major axis, b is the semi-minor axis, (x_c, y_c) is the center, and θ varies in $[0, 2\pi)$. Due to the similarity with the parametric equations of a circle, the coordinates of the center (x_c, y_c) can be updated using the same criteria.

2.4.1. Updating a and b

We define two forces F_a and F_b for a and b, representing the horizontal and vertical inward squeeze respectively:

$$F_a = \frac{1}{N_a}\left(\sum_{\frac{3\pi}{4}<\theta_i<\frac{5\pi}{4}} \mathbf{F}(x_i, y_i)\cdot[1, 0]^\top + \sum_{\theta_i<\frac{\pi}{4}\text{ or }\theta_i>\frac{7\pi}{4}} \mathbf{F}(x_i, y_i)\cdot[-1, 0]^\top\right), \tag{27}$$

$$F_b = \frac{1}{N_b}\left(\sum_{\frac{5\pi}{4}<\theta_i<\frac{7\pi}{4}} \mathbf{F}(x_i, y_i)\cdot[0, 1]^\top + \sum_{\frac{\pi}{4}<\theta_i<\frac{3\pi}{4}} \mathbf{F}(x_i, y_i)\cdot[0, -1]^\top\right), \tag{28}$$

where

$$N_a = \sum_{\frac{3\pi}{4}<\theta_i<\frac{5\pi}{4}} 1 + \sum_{\theta_i<\frac{\pi}{4}\text{ or }\theta_i>\frac{7\pi}{4}} 1, \tag{29}$$

$$N_b = \sum_{\frac{5\pi}{4}<\theta_i<\frac{7\pi}{4}} 1 + \sum_{\frac{\pi}{4}<\theta_i<\frac{3\pi}{4}} 1. \tag{30}$$

The integral of F_a is performed on the left quarter and right quarter of the ellipse, and the integral of F_b is on the lower quarter and upper quarter of the ellipse. N_a and N_b are the numbers of pixels in each integral respectively. Given the steps δa and δb, and the tolerance t_a and t_b, we form our updating criteria as

$$\begin{cases} a_{\text{new}} = a - \delta a & \text{if } F_a > t_a \\ a_{\text{new}} = a + \delta a & \text{if } F_a < -t_a \end{cases}, \tag{31}$$

$$\begin{cases} b_{\text{new}} = b - \delta b & \text{if } F_b > t_b \\ b_{\text{new}} = b + \delta b & \text{if } F_b < -t_b \end{cases}. \tag{32}$$

2.5. Fitting an ellipse with arbitrary orientation

For an ellipse with orientation ϕ, the parametric equations become

$$\begin{cases} x = x_c + a\cos\theta\cos\phi - b\sin\theta\sin\phi \\ y = y_c + a\cos\theta\sin\phi + b\sin\theta\cos\phi \end{cases}. \tag{33}$$

The coordinates of the center (x_c, y_c), the semi-major axis a, and the semi-minor axis b can be updated using similar criteria as fitting an ellipse in standard orientation. But note that in (27) and (28), $[1, 0]^\top$, $[-1, 0]^\top$, $[0, 1]^\top$ and $[0, -1]^\top$ should be replaced by $[\cos\phi, \sin\phi]^\top$, $[-\cos\phi, -\sin\phi]^\top$, $[-\sin\phi, \cos\phi]^\top$ and $[\sin\phi, -\cos\phi]^\top$ respectively.

2.5.1. Updating ϕ

For the orientation ϕ we again define an average torque that rotates the ellipse around its center. The average torque about the center is defined as

$$T_c = \frac{1}{N^2}\sum_{i=1}^{N} d_i\,\mathbf{F}(x_i, y_i)\cdot\begin{bmatrix} -\sin(\theta+\phi) \\ \cos(\theta+\phi) \end{bmatrix}, \tag{34}$$

where

$$d_i = \sqrt{(x_i - x_c)^2 + (y_i - y_c)^2} \tag{35}$$

is the distance between (x_i, y_i) and the center (x_c, y_c). Defining $\delta\phi$ as the step we use to update ϕ in each iteration, and t_ϕ as the tolerance for T_c, the criterion for updating ϕ is

$$\begin{cases} \phi_{\text{new}} = \phi + \delta\phi & \text{if } T_c > t_\phi \\ \phi_{\text{new}} = \phi - \delta\phi & \text{if } T_c < -t_\phi \end{cases}. \tag{36}$$

2.6. Fitting a distorted ellipse

As an extension of the ellipse fitting framework, we now develop a new shape called *distorted ellipse*. The distorted ellipse model will be used to describe the shape of the CSF structure in Section 4. The parametric equations of a distorted ellipse are

$$\begin{cases} x = x_c + a\cos\theta \\ y = y_c + b\left(1 - (1 - \sin\theta)^p\right) \end{cases}. \tag{37}$$

Here (x_c, y_c) is the center of the undistorted ellipse, a and b are the semi-major axis and semi-minor axis respectively, p is the distortion parameter, and θ is between 0 and 2π. We give some example shapes of the distorted ellipse model in Fig. 1. For CSF structure detection, we require $p > 1$.

2.6.1. Updating p

To fit this distorted ellipse model to an image, we update x_c, y_c, a and b the same way as we update them for an ellipse in standard orientation (Section 2.4). We define the force F_p for p similarly as F_a and F_b, but only on the most protruding part (the lower eighth in this example) of the distorted ellipse:

$$F_p = \frac{1}{N_p}\sum_{\frac{11\pi}{8}<\theta_i<\frac{13\pi}{8}} \mathbf{F}(x_i, y_i)\cdot[0, 1]^\top, \tag{38}$$

where

$$N_p = \sum_{\frac{11\pi}{8}<\theta_i<\frac{13\pi}{8}} 1. \tag{39}$$

Given the step δp and the tolerance t_p, the updating criterion is

$$\begin{cases} p_{\text{new}} = p - \delta p & \text{if } F_p > t_p \\ p_{\text{new}} = p + \delta p & \text{if } F_p < -t_p \end{cases}. \tag{40}$$

2.7. Fitting an arbitrary smooth closed contour

So far we have discussed how to configure an active geometric shape model for lines, circles, ellipses and distorted ellipses. Now we further generalize our model to more complicated shapes: arbitrary smooth closed contours. To represent a closed contour, unlike the active contour model, which records all points (pixels) along the contour [3], we model the shape using only a limited number of landmarks, and determine the location of other points by cubic spline interpolation [17].

2.7.1. Abstraction of the shape

Assume the center of the shape is (x_c, y_c), and we use N_{lm} landmarks to represent the shape: $P_1, P_2, \ldots, P_{N_{lm}}$. The distance from the landmark P_k to the center is D_k, and the orientation of the vector from the center to P_k is Θ_k, where we further assume

$$\Theta_k = (k-1)\frac{2\pi}{N_{lm}}. \tag{41}$$

Then the coordinates of landmark P_k are

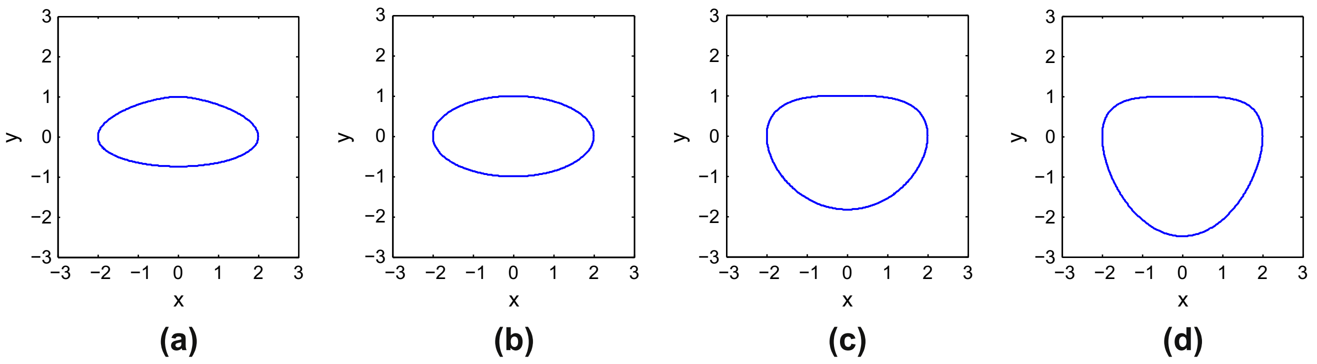


Fig. 1. Example shapes of the distorted ellipse model, where $a = 2$ and $b = 1$, and (a) $p = 0.8$; (b) $p = 1$; (c) $p = 1.5$; and (d) $p = 1.8$.

$$\begin{cases} x_{P_k} = x_c + D_k \cos \Theta_k \\ y_{P_k} = y_c + D_k \sin \Theta_k \end{cases}. \tag{42}$$

For any point Q on the shape, assume its orientation (with respect to the center) is θ_Q, and the distance to the center is d_Q. Then d_Q can be considered as a function of θ_Q, which we denote as

$$d_Q = \eta(\theta_Q). \tag{43}$$

We already know that $D_1 = \eta(\Theta_1), D_2 = \eta(\Theta_2), \ldots, D_{N_{lm}} = \eta(\Theta_{N_{lm}})$, thus given θ_Q, we can obtain d_Q by approximating $\eta(\cdot)$ using cubic spline interpolation [17]. Such shapes will be called *cubic spline contours* in the context. Examples of cubic spline contours are shown in Fig. 2. Note that circle is also a cubic spline contour with $N_{lm} = 1$ and $D_1 = r$.

2.7.2. Updating parameters

For a cubic spline contour described above, if N_{lm} is already given, then our shape model has $N_{lm} + 2$ parameters in total: x_c, y_c and $D = (D_1, D_2, \ldots, D_{N_{lm}})$. x_c and y_c can be updated in a similar way as circles and ellipses – take the integral of the force field along the shape and project it onto four directions (Section 2.3.1). Now we focus on updating each D_k. Assume we are working on an image, and the N pixels (x_i, y_i) on the contour are represented as

$$\begin{cases} x_i = x_c + d_i \cos \theta_i \\ y_i = y_c + d_i \sin \theta_i \end{cases}. \tag{44}$$

We define a force F_{D_k} for each D_k:

$$F_{D_k} = \frac{1}{N_{D_k}} \sum_{\Theta_k - \frac{\pi}{N_{lm}} < \theta_i < \Theta_k + \frac{\pi}{N_{lm}}} \mathbf{F}(x_i, y_i) \cdot [\cos \theta_i, \sin \theta_i]^{\mathsf{T}}. \tag{45}$$

Defining δD as the step we use to update D_k in each iteration, and t_D as the tolerance for D_k, the criterion for updating D_k is

$$\begin{cases} D_{k\text{new}} = D_k + \delta D & \text{if } D_k > t_D \\ D_{k\text{new}} = D_k - \delta D & \text{if } D_k < -t_D \end{cases}. \tag{46}$$

2.8. More about parameter updating

Generally, for a geometric shape model, we associate a parameter α with a force (or torque) F_α. We define the precision of α to be $\delta\alpha$, which is the smallest update step of α allowed in one iteration (in previous sections it is also the only update step being used). We also define the tolerance of F_α to be t_α, which means that if the absolute value of F_α is smaller than t_α, α will not be updated in this iteration. Then the simplest update criterion, as we have already described, would be

$$\begin{cases} \alpha_{\text{new}} = \alpha + \delta\alpha & \text{if } F_\alpha > t_\alpha \\ \alpha_{\text{new}} = \alpha - \delta\alpha & \text{if } F_\alpha < -t_\alpha \end{cases}. \tag{47}$$

The sign before $\delta\alpha$ in (47) can be opposite, depending on the definition of F_α.

However, using a constant update step $\delta\alpha$ is generally inefficient, since the required precision could be very small. An alternative approach is to scale the step with the magnitude of F_α. Since we cannot allow the step to be arbitrarily large, we scale it using a sigmoid-like function $g(\cdot)$. Let $\Delta\alpha$ be the largest step allowed in each iteration; the scaling function $g(F_\alpha)$ must then satisfy

$$g(t_\alpha) = \delta\alpha, \tag{48}$$

$$\lim_{|F_\alpha| \to \infty} |g(F_\alpha)| = \Delta\alpha. \tag{49}$$

Then our updating criterion becomes

$$\alpha_{\text{new}} = \alpha + g(F_\alpha). \tag{50}$$

Available choices of $g(F_\alpha)$ include

$$g_1(F_\alpha) = \frac{2}{\pi} \Delta\alpha \arctan\left(\frac{F_\alpha}{t_\alpha} \tan \frac{\pi\delta\alpha}{2\Delta\alpha}\right), \tag{51}$$

$$g_2(F_\alpha) = \Delta\alpha \tanh\left(\frac{F_\alpha}{2t_\alpha} \ln \frac{\Delta\alpha + \delta\alpha}{\Delta\alpha - \delta\alpha}\right), \tag{52}$$

$$g_3(F_\alpha) = \Delta\alpha F_\alpha \left(F_\alpha^2 + t_\alpha^2 \left(\left(\frac{\Delta\alpha}{\delta\alpha}\right)^2 - 1\right)\right)^{-\frac{1}{2}}, \tag{53}$$

$$g_4(F_\alpha) = \Delta\alpha \left(\operatorname{sgn}(F_\alpha) + \frac{t_\alpha}{F_\alpha}\left(\frac{\Delta\alpha}{\delta\alpha} - 1\right)\right)^{-1}. \tag{54}$$

We provide a plot of these four sigmoid-like scaling functions in Fig. 3. We can see that, among these four functions, with the same input F_α, g_2 has the largest update step, g_3 second largest, and g_4 smallest.

Again, we emphasize that the order in which we update different parameters does not affect the updating results. This is because in each iteration, we always compute all the forces and torques using current parameters first, and then update all parameters with these forces and torques (e.g. Algorithm 1).

2.9. Correction of curvature

When generating the force field, we use Eq. (4) for external energy. While convolving the image with the Gaussian kernel, the local maxima also dislocate to the concave side of a curve. Thus when fitting the geometric shape model in the force field, the shape will converge to the new local maxima instead of the original curves. A similar problem in edge detection was well solved by

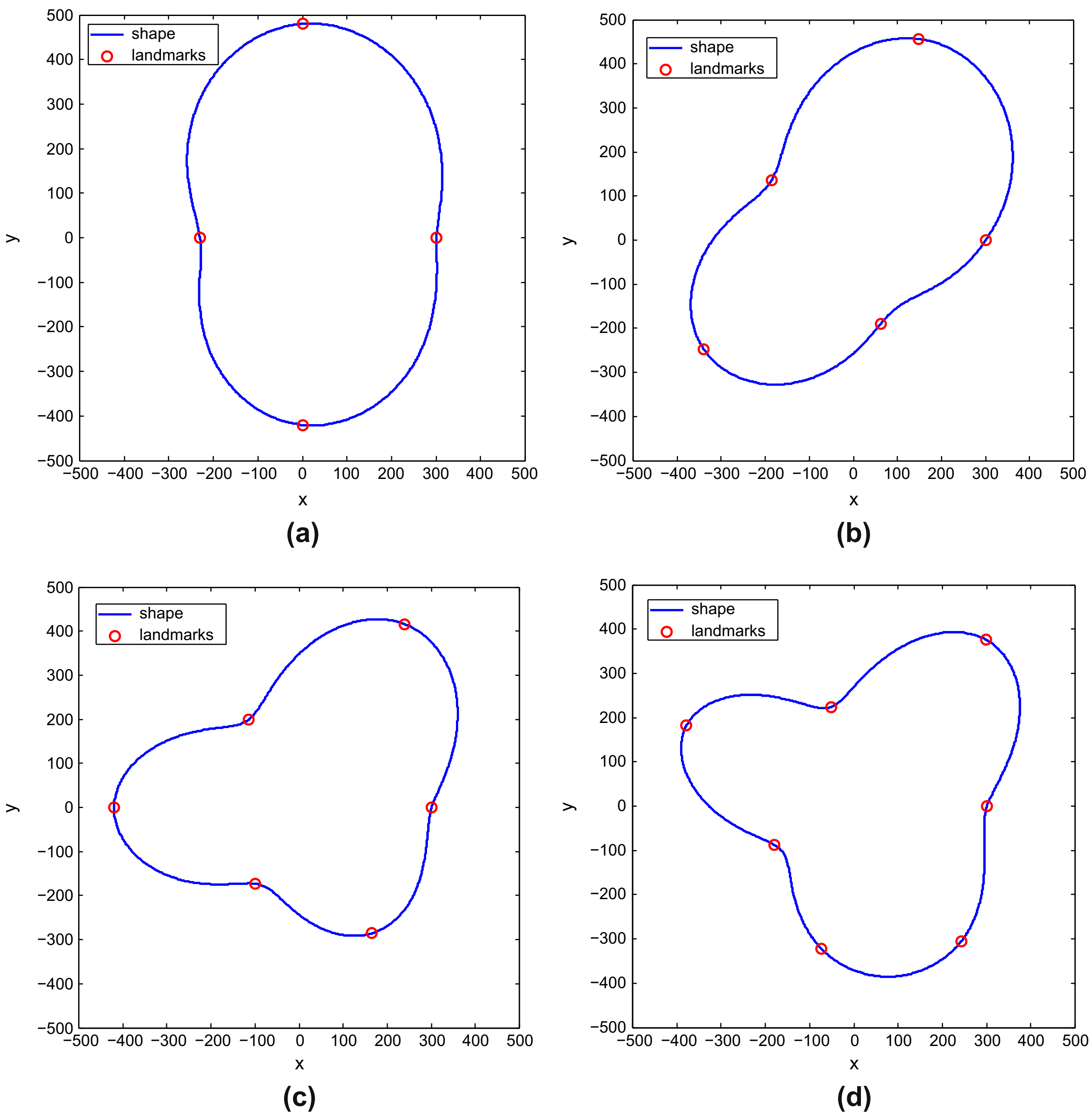


Fig. 2. Examples of cubic spline contours: (a) four landmarks; (b) five landmarks; (c) six landmarks; and (d) seven landmarks.

Bouma et al. in 2005 [18]. We now extend their method to solve our problem by correcting the dislocation according to the local curvature.

2.9.1. Correction for a circle

In the polar coordinate system (ρ, θ), we define a disk with radius R as

$$M(\rho, \theta) = U(R - \rho), \tag{55}$$

where $U(\cdot)$ is the unit step. And we define its convolution with Gaussian kernel $G_\sigma(\rho, \theta)$ as

$$L(\rho, \theta) = G_\sigma * M. \tag{56}$$

The derivative of $M(\rho, \theta)$ in the radial direction is simply

$$M_\rho = -\delta(R - \rho). \tag{57}$$

Then the first order derivative of $L(\rho, \theta)$ in the radial direction can be obtained:

$$L_\rho(\rho, \theta) = G_\sigma * M_\rho = -\frac{R}{\sigma^2} e^{-\frac{R^2+\rho^2}{2\sigma^2}} I_1\left(\frac{\rho R}{\sigma^2}\right), \tag{58}$$

where $I_n(\cdot)$ is the modified Bessel function of the first kind. Now we are interested in finding the positions of the local minima of $L_\rho(\rho, \theta)$. The second order derivative can be derived from L_ρ [18]:

$$L_{\rho\rho}(\rho, \theta) = e^{-\frac{R^2+\rho^2}{2\sigma^2}} \left(-\frac{R^2}{\sigma^4} I_0\left(\frac{\rho R}{\sigma^2}\right) + \left(\frac{\rho R}{\sigma^4} + \frac{R}{\rho\sigma^2}\right) I_1\left(\frac{\rho R}{\sigma^2}\right) \right). \tag{59}$$

If for a radius r, we have $L_{\rho\rho}(r, \theta) = 0$, then r is the dislocated radius of the original disk $M(\rho, \theta)$, or the circle $M_\rho(\rho, \theta)$, whose true radius is R. Then r must satisfy

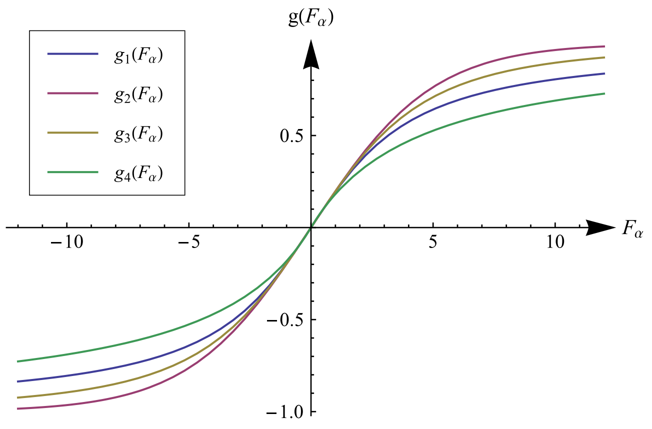


Fig. 3. Plot of the four sigmoid-like scaling functions. Here we set $\delta\alpha = 0.1$, $\Delta\alpha = 1$, and $t_\alpha = 0.5$.

$$\frac{R}{\sigma^2} I_0\left(\frac{rR}{\sigma^2}\right) = \left(\frac{r}{\sigma^2} + \frac{1}{r}\right) I_1\left(\frac{rR}{\sigma^2}\right). \tag{60}$$

If r and σ are given, we can use this equation to solve for R. We represent the solution R as a function of r and σ:

$$R = \Omega(r, \sigma). \tag{61}$$

Now for the circle fitting problem, if the image is blurred by a Gaussian kernel of standard deviation σ, and the radius obtained by fitting to the force field is r, then we should correct the radius to $R = \Omega(r, \sigma)$.

Generally it is difficult to give an analytic expression of the function $\Omega(\cdot)$. But when $r \gg \sigma$, we can still compute R numerically by setting $R^{(0)} = r$ and using the iteration below:

$$R^{(k+1)} = \left(r + \frac{\sigma^2}{r}\right) \frac{I_1\left(\frac{rR^{(k)}}{\sigma^2}\right)}{I_0\left(\frac{rR^{(k)}}{\sigma^2}\right)}. \tag{62}$$

For the modified Bessel functions of the first kind, we know that when $x \gg |n^2 - 1/4|$, $I_n(x)$ can be approximated by [19]

$$I_n(x) = \frac{e^x}{\sqrt{2\pi x}}\left(1 - \frac{4n^2-1}{8x} + \frac{(4n^2-1)(4n^2-9)}{128x^2} + O(x^{-3})\right). \tag{63}$$

Thus (62) can be computed using

$$\frac{I_1(x)}{I_0(x)} \approx \frac{128x^2 - 48x - 15}{128x^2 + 16x + 9}. \tag{64}$$

2.9.2. Correction for an ellipse

For shapes other than circles, such as ellipses, it is difficult to obtain a direct conclusion about the dislocation. Thus we still use the conclusion for the circle, but correct for the radius of curvature locally. For an ellipse, correction for radius of curvature at all positions is still trivial and the resulting shape is not necessarily an ellipse. However, we can approximately correct the semi-major axis a and semi-minor axis b for the curvature only at $\theta = k\pi/2$ where $k = 0, 1, 2, 3$.

Given the parametric equations $x = x(\theta)$ and $y = y(\theta)$, the radius of curvature can be defined as [20]

$$r(\theta) = \frac{(\dot{x}^2 + \dot{y}^2)^{3/2}}{\dot{x}\ddot{y} - \ddot{x}\dot{y}}. \tag{65}$$

To simplify, we assume the major axis and minor axis are parallel to x-axis and y-axis respectively. Due to symmetry, we only need to look at the local radius of curvature around $\theta = 0$ and $\theta = \pi/2$, which are given as

$$r(0) = \frac{b^2}{a}, \tag{66}$$

$$r\left(\frac{\pi}{2}\right) = \frac{a^2}{b}, \tag{67}$$

where a and b are the semi-major axis and semi-minor axis obtained by fitting the active geometric shape model iteratively. Then the corrected semi-major axis a' and corrected semi-minor axis b' can be computed by

$$\frac{b'^2}{a'} = R_1 = \Omega\left(\frac{b^2}{a}, \sigma\right), \tag{68}$$

$$\frac{a'^2}{b'} = R_2 = \Omega\left(\frac{a^2}{b}, \sigma\right), \tag{69}$$

which results in

$$a' = \sqrt[3]{R_2^2 R_1}, \tag{70}$$

$$b' = \sqrt[3]{R_1^2 R_2}. \tag{71}$$

The same idea of only correcting for the curvature at salient points can be also used for more complex shapes.

2.9.3. Correction for a distorted ellipse

For the distorted ellipse model with $p > 1$, the radii of curvature at $\theta = 0,\ \pi/2,\ \pi$ and $3\pi/2$ are

$$r(0) = \frac{p^2 b^2}{a}, \tag{72}$$

$$r\left(\frac{\pi}{2}\right) = +\infty, \tag{73}$$

$$r(\pi) = \frac{p^2 b^2}{a}, \tag{74}$$

$$r\left(\frac{3\pi}{2}\right) = \frac{a^2}{2^{p-1} p b}. \tag{75}$$

Since $r(0) = r(\pi)$ and $r(\pi/2)$ is constant, we only have two constraints to correct three shape parameters a, b and p. Thus to correct for curvature approximately, one method is to correct only a and b, and keep p unchanged. Assuming the original image is blurred by a Gaussian kernel of standard deviation σ, let

$$R_1 = \Omega\left(\frac{p^2 b^2}{a}, \sigma\right), \tag{76}$$

$$R_2 = \Omega\left(\frac{a^2}{2^{p-1} p b}, \sigma\right), \tag{77}$$

then the corrected a' and b' are

$$a' = \sqrt[3]{2^{2(p-1)} R_1 R_2^2}, \tag{78}$$

$$b' = \sqrt[3]{\frac{2^{p-1}}{p^3} R_1^2 R_2}. \tag{79}$$

2.9.4. Correction for a cubic spline contour

For each point (x_i, y_i) on the cubic spline contour, the distance d_i to the center (x_c, y_c) is a function of θ_i:

$$d_i = \eta(\theta_i). \tag{80}$$

Then the radius of curvature at this point is

$$r(\theta_i) = \frac{(\eta^2(\theta_i) + \eta'^2(\theta_i))^{\frac{3}{2}}}{\eta^2(\theta_i) + 2\eta'^2(\theta_i) - \eta(\theta_i)\eta''(\theta_i)}. \tag{81}$$

Table 1
Ground truth parameters, and average absolute values of parameter errors using total least squares method, and active geometric shape models, respectively, over 20 datasets. Each row of this table represents an independent line fitting experiment.

Number of noisy data points	Number of outliers	Ground truth parameters		Total least squares fit		Active geometric shape model fit	
		θ_t	s_t	$\overline{\lvert e_\theta\rvert}$	$\overline{\lvert e_s\rvert}$	$\overline{\lvert e_\theta\rvert}$	$\overline{\lvert e_s\rvert}$
50	0	3.8675	−319.74	0.0044	0.61	0.0053	0.54
50	0	1.1458	285.29	0.0028	0.70	0.0033	0.69
50	5	4.7010	−202.84	0.0250	5.85	0.0063	1.63
50	5	5.1602	−72.03	0.0214	6.06	0.0048	1.99
100	0	0.1708	280.36	0.0032	0.66	0.0042	0.76
100	0	3.4359	−297.27	0.0032	0.51	0.0046	0.54
100	5	0.9093	311.39	0.0052	2.05	0.0024	0.47
100	5	2.9887	−216.62	0.0258	6.92	0.0085	1.95
100	10	3.2245	−265.70	0.0295	5.67	0.0060	1.45
100	10	3.6531	−315.90	0.0177	3.61	0.0049	0.64

To approximately correct for the curvature, we only correct the parameters D_k of the cubic spline contour. For each D_k, if $r(\Theta_k) > 0$, we correct D_k to

$$D'_k = D_k - r(\Theta_k) + \Omega(r(\Theta_k), \sigma). \tag{82}$$

Otherwise if $r(\Theta_k) < 0$, we similarly correct D_k to

$$D'_k = D_k + r(\Theta_k) - \Omega(r(\Theta_k), \sigma). \tag{83}$$

2.10. Optimality of fitting results

So far, we have discussed the procedures of fitting lines, circles, ellipses, distorted ellipses and cubic spline contours to images. These procedures are not direct solutions of minimizing an energy function or maximizing a probability, but are designed using our geometric understanding of the parameters of different shapes. Defining the optimality of a shape parameter set in the image domain is still an open problem. To validate our method, we define our own fitness function of a shape parameter set. For a closed contour (such as a circle, ellipse, or cubic spline contour), its parameter set $\mathcal{P}$ can be divided into two subsets: the size parameter set $\mathcal{P}_1$ and non-size parameter set $\mathcal{P}_2$. For example, for an ellipse, $\mathcal{P}_1 = \{a, b\}, \mathcal{P}_2 = \{x_c, y_c, \phi\}$; for a cubic spline contour, $\mathcal{P}_1 = \{D_1, D_2, \ldots, D_{N_{lm}}\}$, $\mathcal{P}_2 = \{x_c, y_c\}$. Let β be a constant such that $0 < \beta < 1$. Then the parameter set $\mathcal{P}' = (\beta\mathcal{P}_1) \cup \mathcal{P}_2$ represents a shrunken version of the shape, and $\mathcal{P}'' = (\frac{1}{\beta}\mathcal{P}_1) \cup \mathcal{P}_2$ represents an expanded version of the shape. Let (x'_i, y'_i) denote the points on the shrunken shape, (x''_i, y''_i) denote the points on the expanded shape, N' denote the number of points on the shrunken shape, and N'' denote the number of points on the expanded shape. Then we use the summation of the force field magnitude $\|\mathbf{F}(x, y)\|$ along these shape contours to define our fitness function:

$$\mathcal{F}(\mathcal{P}) = \frac{1}{N}\sum_{i=1}^{N}\|\mathbf{F}(x_i, y_i)\| - \frac{1}{2N'}\sum_{i=1}^{N'}\|\mathbf{F}(x'_i, y'_i)\| - \frac{1}{2N''}\sum_{i=1}^{N''}\|\mathbf{F}(x''_i, y''_i)\|. \tag{84}$$

If we choose a β close to 1, then a small value of $\mathcal{F}(\mathcal{P})$ indicates a good fit to the image. If the image $I(x, y)$ is a gray-level image with brighter regions of interest, we can also substitute $\|\mathbf{F}(x, y)\|$ in (84) with $-I(x, y)$. We will show in Section 3 that our parameter updating strategies iteratively decrease the fitness function value.

Besides, the fitness function can be also used to choose the best fit of several trials with different initial conditions. As an example, now we explicitly give our complete ellipse fitting algorithm including correction of curvature and fitness evaluation in Algorithm 1.

We also indicate that for each shape, the choice of forces, torques, and parameter update criteria proposed in our paper is not the unique choice, thus is not the standard answer. How to design and configure the active geometric shape model for a specific shape should be left as an open problem, and depends on the problem to be solved.

Algorithm 1. The ellipse fitting algorithm.

```
input: image $I(x,y)$
output: parameters $x_c, y_c, a, b$ and $\phi$
begin
compute the force field $\mathbf{F}(x,y)$;
for $i \leftarrow 1$ to Num_Of_Trials do
    randomly generate initial parameters;
    for $j \leftarrow 1$ to Max_Num_Of_Iterations do
        compute forces $F_{ch}, F_{cv}, F_{cd}, F_{ca}$ for $x_c$ and $y_c$;
        compute forces $F_a$ and $F_b$ for $a$ and $b$;
        compute the torque $T_c$ for $\phi$;
        update $x_c, y_c, a, b$ and $\phi$;
    end
    compute the fitness function value $\mathcal{F}_i$;
end
find the minimal $\mathcal{F}_i$ of all trials (the $i^*$th trial);
get the resulting parameters of the $i^*$th trial;
correct $a$ and $b$ for curvature;
done
```

3. Experiments

Before we use our models for real data analysis, where the shape is complex, we first show some synthetic data results of lines, ellipses and cubic spline contours (circle fitting is no more difficult than ellipse fitting).

For line, ellipse and cubic spline contour fitting, we first generate random parameters of these shapes as ground truth, then generate data points on these shapes randomly, and add Gaussian noise to the positions of these points. We also randomly generate some outlier points to assess the robustness of our models. From these noisy data points and outliers, we generate a binary image with a black background and white foreground. Then we fit our active geometric shape model to this image and compare the parameters we obtained by fitting the active geometric shape model with the ground truth parameters. For the line fitting, we also compare with the total least squares method.

3.1. Line fitting

In our line fitting experiments, we work on 500×400 binary images. In each experiment, first we generate a ground truth parameter θ_t randomly between 0 and 2π, and then generate the

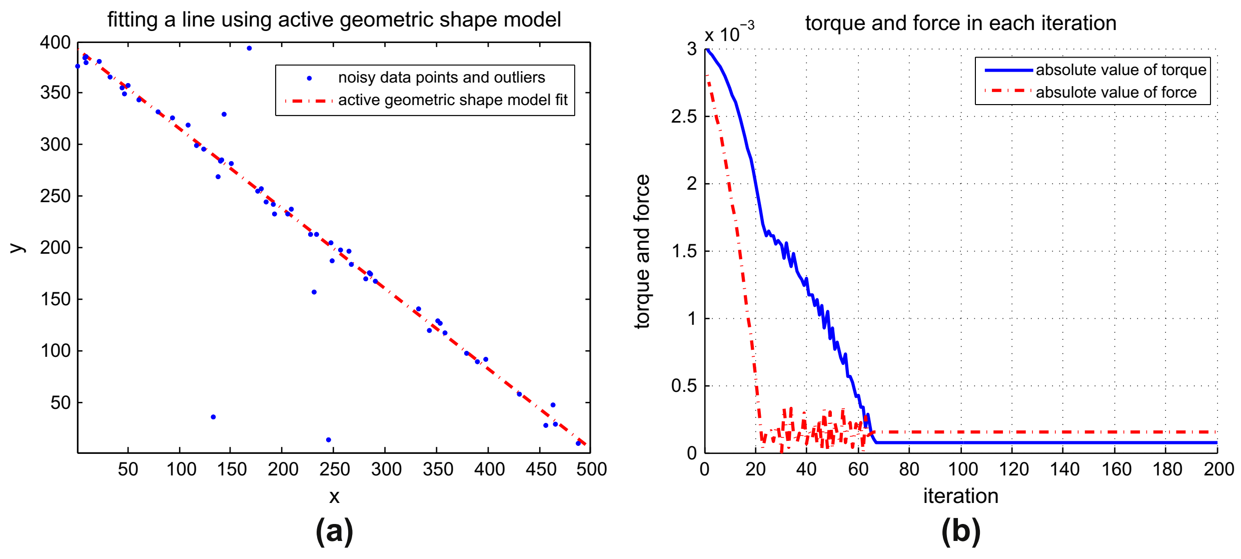


Fig. 4. Line fitting result in one experiment on one dataset. The dataset consists of 50 noisy data points and five outliers. The ground truth parameters are $\theta_t = 0.9169, s_t = 310.81$, and the resulting parameters are $\theta = 0.9092, s = 310.95$. (a) Plot of the line fitting result together with the noisy data points and outliers and (b) force and torque in each iteration.

Table 2
Ground truth parameters, and average absolute values of parameter errors using active geometric shape models over 20 datasets. Each row of this table represents an independent ellipse fitting experiment.

Number of noisy data points	Number of outliers	Ground truth parameters					Active geometric shape model fit				
		x_c	y_c	a	b	ϕ	$\overline{\lvert e_{x_c}\rvert}$	$\overline{\lvert e_{y_c}\rvert}$	$\overline{\lvert e_a\rvert}$	$\overline{\lvert e_b\rvert}$	$\overline{\lvert e_\phi\rvert}$
50	0	236.71	220.40	121.37	75.21	1.5931	0.95	0.85	0.99	1.12	0.0254
50	0	285.16	213.55	95.86	82.55	0.4986	1.01	0.96	1.37	1.16	0.0805
50	5	292.27	223.14	101.59	38.94	2.9342	1.52	0.96	1.98	1.34	0.0179
50	5	288.55	181.77	125.00	74.34	2.5404	1.05	1.00	1.69	1.41	0.0212
100	0	290.32	197.94	111.27	72.18	2.4012	0.65	0.55	0.97	0.63	0.0183
100	0	218.52	197.94	98.34	73.80	0.7966	0.79	0.67	1.01	0.86	0.0292
100	5	284.01	201.97	98.57	81.25	0.4802	0.72	0.67	0.89	0.74	0.0424
100	5	225.05	214.98	97.52	45.12	2.3622	0.97	0.95	1.17	0.65	0.0099
100	10	243.65	229.36	123.53	74.27	2.2988	0.80	0.84	0.80	0.91	0.0131
100	10	238.49	170.91	114.22	42.13	2.7380	0.98	0.57	1.45	1.44	0.0077

ground truth parameter s_t such that the line passes through the middle of the image. With the two ground truth parameters, we generate 20 datasets. Each dataset consists of some outlier points randomly generated in the image, and 50 or 100 data points randomly generated using the ground truth parameters. Then we add Gaussian noise with zero mean and a standard deviation of 5 to both the x and y coordinates of the data points. We generate a binary image with a black background and white foreground for these noisy data points and outliers. To obtain the force field, we use (4) as the external energy where $\sigma = 50$, and use the solution of (5) as our GVF force field, where we set $\mu = 0.1$. Then we start our algorithm with random initial parameters, and set $\delta\theta = 0.2^\circ, \delta s = 0.2$, and the number of iterations to 200.

To assess a line fitting algorithm, we compare the obtained parameters θ and s with the ground truth parameters θ_t and s_t. We compute the average absolute values of the parameter errors over the 20 datasets: $\overline{|e_\theta|}$ and $\overline{|e_s|}$, where

$$e_\theta = \theta - \theta_t, \tag{85}$$

$$e_s = s - s_t. \tag{86}$$

We also use the total least squares method to fit a line to each dataset. For a line defined by Eq. (7), the distance from a point (u, v) to the line is $|u\cos\theta + v\sin\theta - s|$. For the data points and outliers (x_i, y_i) where $i = 1, 2, \ldots, N$, the mean of squared distances M is defined as

$$M = \frac{1}{N}\sum_{i=1}^{N}(x_i\cos\theta + y_i\sin\theta - s)^2. \tag{87}$$

The total least squares method that minimizes M can be obtained by solving the 2D eigenvalue problem [21]

$$\begin{bmatrix} \overline{x^2} - \bar{x}\bar{x} & \overline{xy} - \bar{x}\bar{y} \\ \overline{xy} - \bar{x}\bar{y} & \overline{y^2} - \bar{y}\bar{y} \end{bmatrix}\begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix} = \lambda\begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix} \tag{88}$$

and setting

$$s = \bar{x}\cos\theta + \bar{y}\sin\theta. \tag{89}$$

In Table 1 we give the ground truth parameters θ_t and s_t of each experiment, and the average absolute values of parameter errors over the 20 datasets using the total least squares method, and our active geometric shape model, respectively. We also give the plot of the force (8) and torque (10) in each iteration, and the plot of the line together with the noisy data points and outliers in one experiment on one dataset in Fig. 4.

As a result, we can see that the errors of the parameters using our active geometric shape model are small; and are notably much smaller than those using the total least squares method in the presence of outliers. Thus our active geometric shape model is shown to be effective, accurate, and robust to outliers for the line fitting problem, and is therefore promising for more complicated

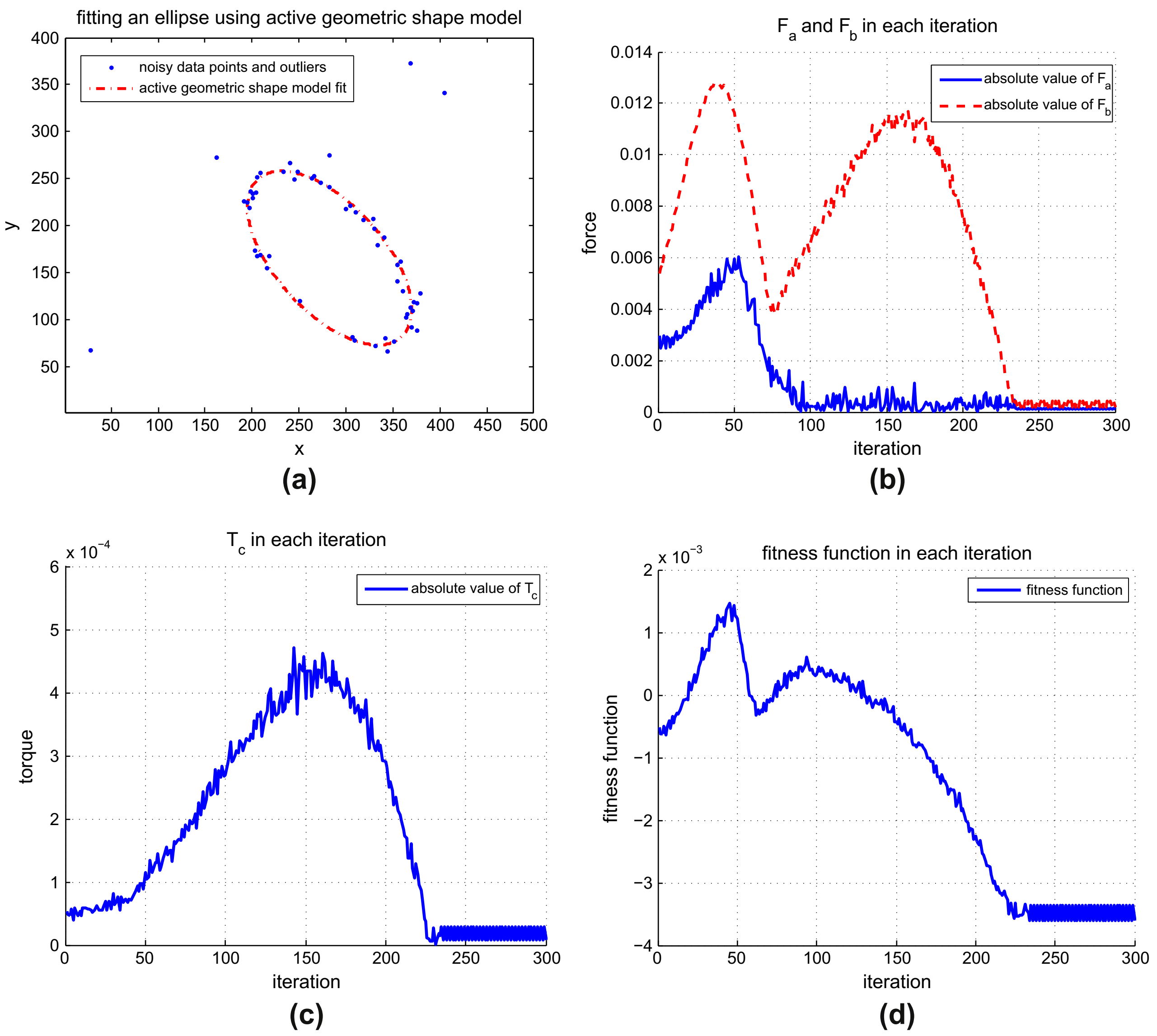


Fig. 5. Ellipse fitting result in one experiment on one dataset. The dataset consists of 50 noisy data points and five outliers. The ground truth parameters are $x_{ct} = 282.41, y_{ct} = 165.22, a_t = 114.64, b_t = 58.14, \phi_t = 2.3179$, and the resulting parameters are $x_c = 282.60, y_c = 165.20, a = 113.58, b = 57.37, \phi = 2.3073$. (a) Plot of the ellipse fitting result together with the noisy data points and outliers; (b) forces on a and b in each iteration (first stage); (c) torque on ϕ in each iteration (first stage); and (d) value of fitness function in each iteration (first stage, $\beta = 0.9$).

shapes for which computing the total least squares would be difficult.

3.2. Ellipse fitting

In our ellipse fitting experiments, we still work on 500×400 binary images. In each experiment, the ground truth parameter ϕ_t is generated randomly between 0 and π, and x_{ct}, y_{ct}, a_t, and b_t are all randomly generated within a wide range. 20 datasets are generated for each experiment. Each dataset includes 50 or 100 data points randomly generated using the ground truth parameters, and Gaussian noise with zero mean and a standard deviation of 5 is added to both x and y coordinates of these data points. Outlier points are also added into these datasets. After generating the black and white image using one dataset, we apply a two-stage ellipse fitting method. In the first stage, we blur the original image using a Gaussian kernel with a large standard deviation $\sigma = 20$ and generate the GVF force field. Then we fit an ellipse to this force field using random initial parameters. In the second stage, we blur the original image using a Gaussian kernel with a small standard deviation $\sigma = 5$ and generate the GVF force field. Then we fit an ellipse again to this force field using the results obtained from the first stage as our initial parameters. By using a large σ in the first stage, the force field has a sufficiently large capture range to locate the ellipse to the rough position. In the second stage, with a small σ, the ellipse will fit to a more accurate position. This two-stage approach can be used as a standard practice for active geometric shape models. Then we correct a and b using (62). In both stages, we set $\delta\phi = 0.2^\circ$ and $\delta x_c = \delta y_c = \delta a = \delta b = 0.2$. The number of iterations is set to 300 in the first stage and 50 in the second stage.

In these 20 experiments, the average running time of our two-stage ellipse fitting algorithm on a 2.53 GHz Intel Core i5 machine is 10.51 s, with standard deviation 1.04 s.[1]

We still use the average absolute values of parameter errors to assess our ellipse fitting results. The ground truth parameters and

[1] This algorithm is currently implemented in MATLAB, and the running time should be significantly improved by recoding in C/C++.

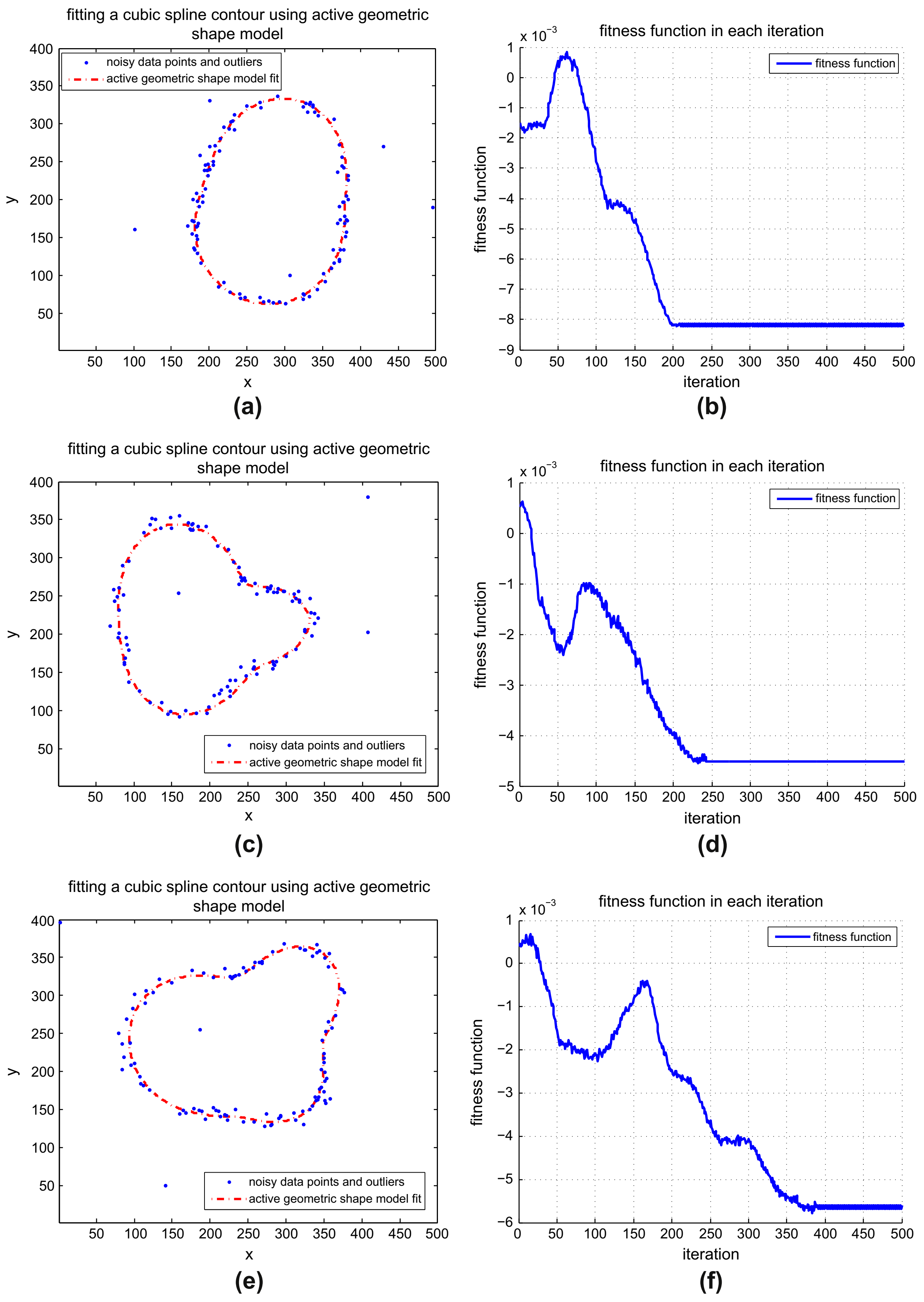
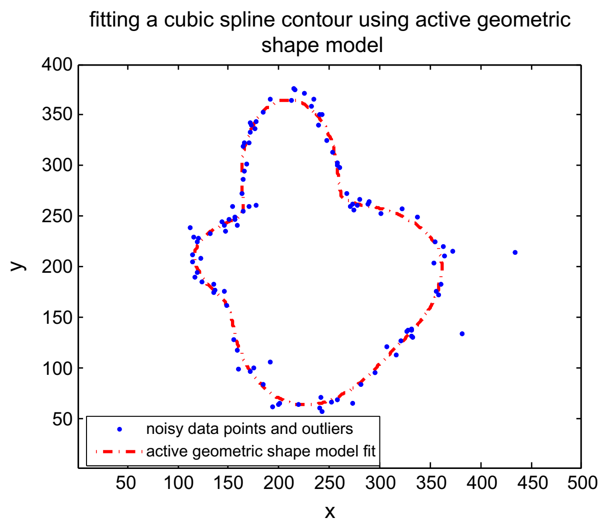
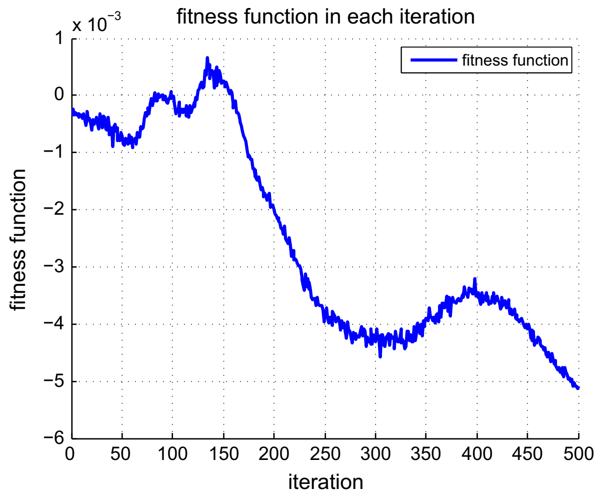


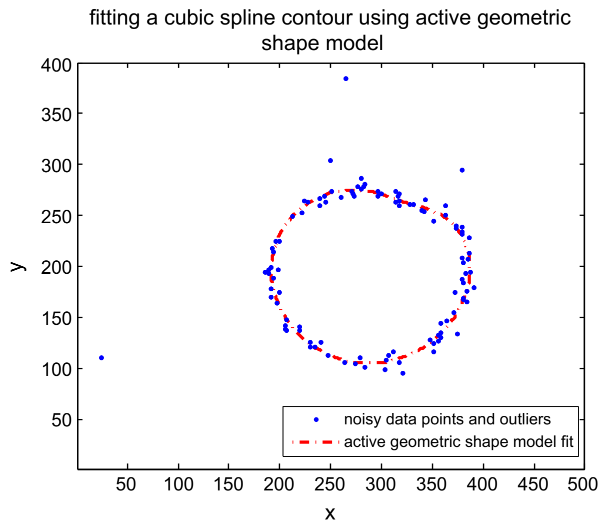
Fig. 6. Cubic spline contour fitting results and fitness function values in each iteration (first stage). Each experiment uses 100 noisy data points and five outliers. (a) and (b) $N_{lm}^{(data)} = 5$, $N_{lm}^{(model)} = 5$; (c) and (d) $N_{lm}^{(data)} = 6$, $N_{lm}^{(model)} = 6$; (e) and (f) $N_{lm}^{(data)} = 7$, $N_{lm}^{(model)} = 7$; (g) and (h) $N_{lm}^{(data)} = 8$, $N_{lm}^{(model)} = 8$; (i) and (j) $N_{lm}^{(data)} = 4$, $N_{lm}^{(model)} = 6$; (k) and (l) $N_{lm}^{(data)} = 5$, $N_{lm}^{(model)} = 7$; (m) and (n) $N_{lm}^{(data)} = 6$, $N_{lm}^{(model)} = 8$; and (o) and (p) $N_{lm}^{(data)} = 6$, $N_{lm}^{(model)} = 4$.



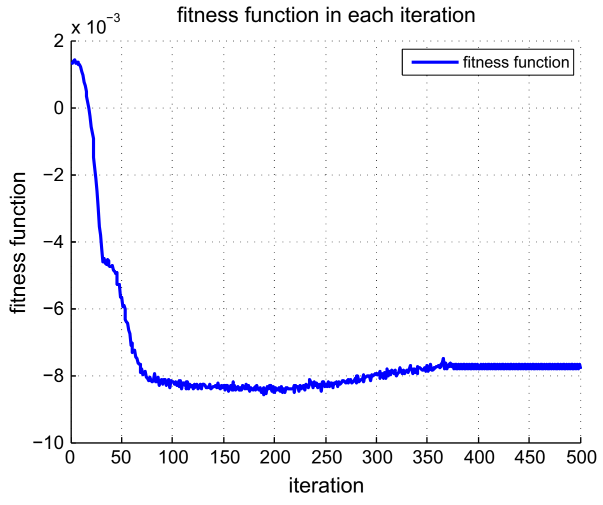
(g)



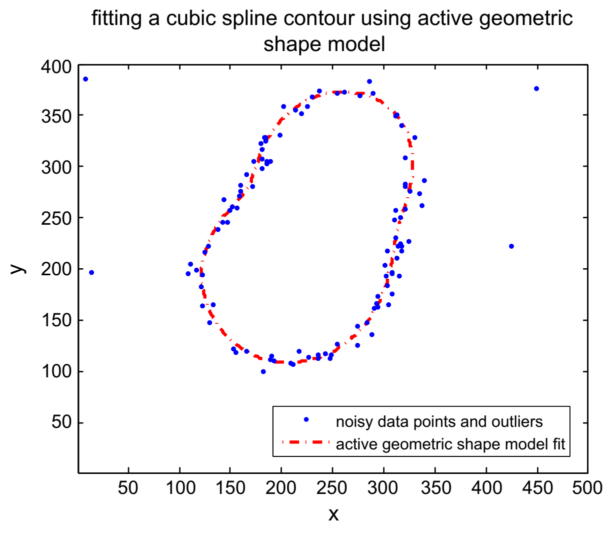
(h)



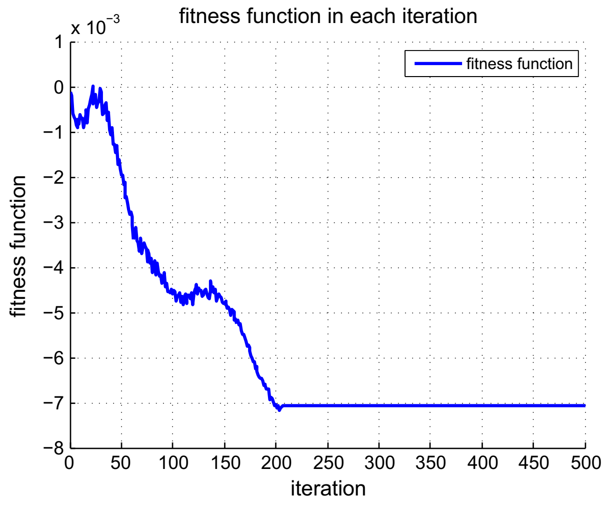
(i)



(j)

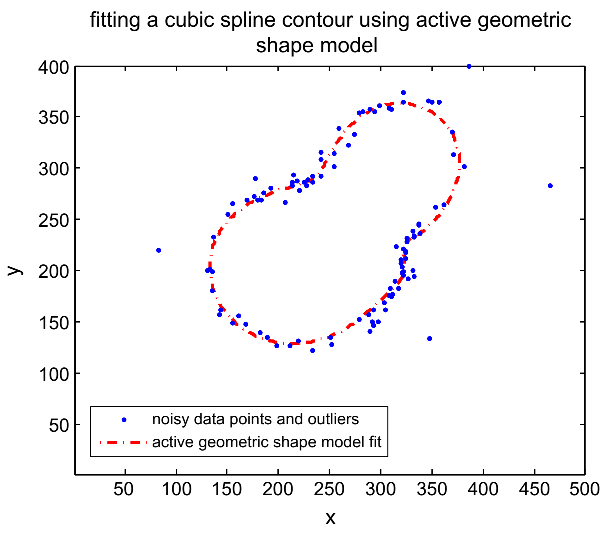


(k)

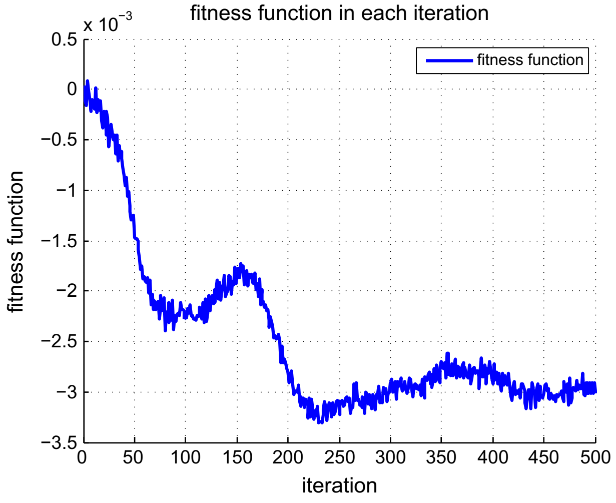


(l)

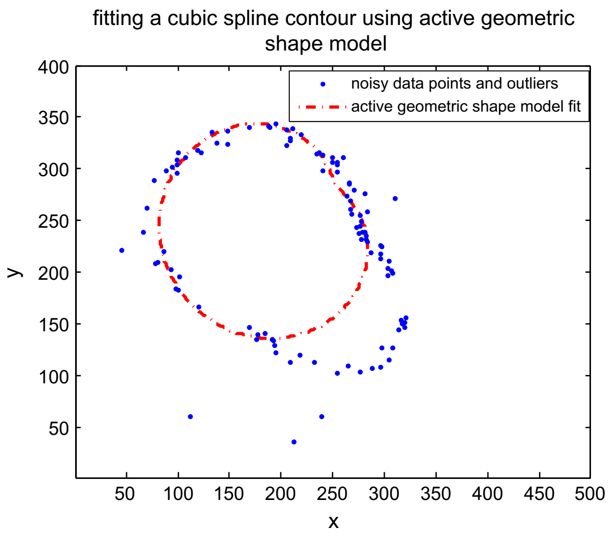
Fig. 6. (*continued*)



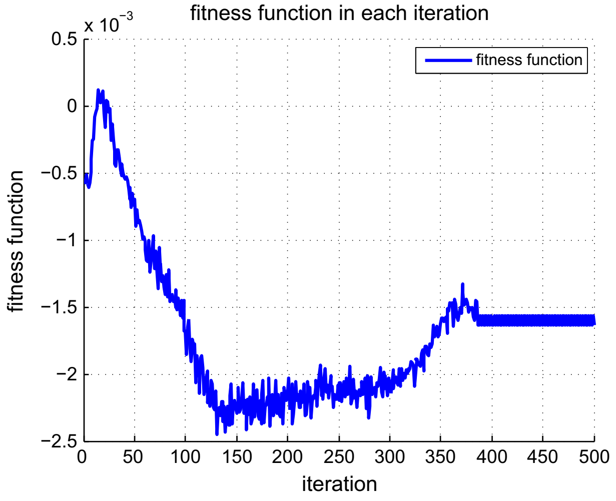
(m)



(n)



(o)



(p)

Fig. 6. (*continued*)

average absolute values of parameter errors over the 20 datasets in each experiment are given in Table 2. The plot of the resulting ellipse together with the noisy data points and outliers, and the plot of the forces (27) and (28), torque (34) and fitness function (84) with $\beta = 0.9$ in each iteration in one experiment on one dataset are given in Fig. 5. We can see that the value of the fitness function decreases in each parameter updating iteration, and that the errors of parameters obtained by our active geometric shape model are small even when there are outliers, which again demonstrates the effectiveness, accuracy and robustness of our method.

3.3. Cubic spline contour fitting

In our cubic spline contour fitting experiments we use similar settings with the ellipse fitting experiments: 500×400 binary images, randomly generated parameters, and two-stage fitting. In each experiment, 100 noisy data points and five outliers are generated for a cubic spline contour with $N_{lm}^{(data)}$ landmarks. Then a cubic spline contour model with $N_{lm}^{(model)}$ landmarks is used to fit the data for 10 times, each time with different randomly generated initial parameters. The resulting parameter set with smallest fitness function value is the final result. We report the results using different $N_{lm}^{(data)}$ and $N_{lm}^{(model)}$ in Fig. 6. We can see that when $N_{lm}^{(model)} \geqslant N_{lm}^{(data)}$, our model is powerful enough to represent the shape. But if $N_{lm}^{(model)} < N_{lm}^{(data)}$ (Fig. 6o and p), the results can be unsatisfactory due to under-modeling, as is expected.

4. Automatic detection and segmentation of CSF

Detection and segmentation of cerebrospinal fluid (CSF) flow in CSF phase-contrast magnetic resonance (PC-MR) images are significant for the study of CSF dynamics, such as estimating the flow rate and wall shear stress [22]. Since the shape of the CSF structure has no salient points, the active shape models, which rely on a large amount of training data and manual labeling, are not suitable for this problem. However, by representing the shape using a distorted ellipse proposed in Section 2.6, our active geometric shape model is very good at detecting the CSF structure in noisy images, while strictly constraining the shape of the model at each iteration (Fig. 7d).

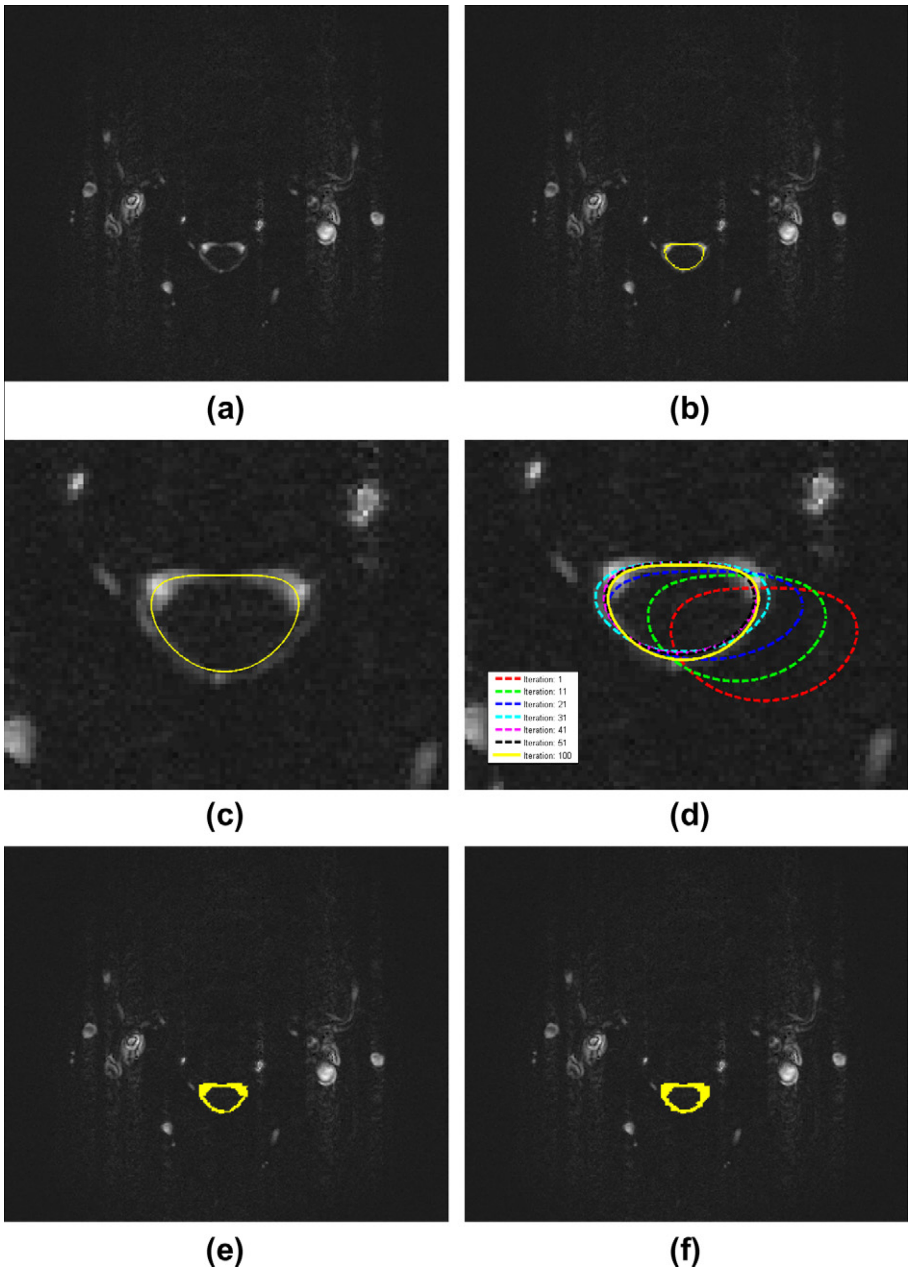


Fig. 7. (a) The correlation map of the CSF image sequence shown in Fig. 8; (b) the distorted ellipse fitting result on the correlation map; (c) a zoom-in view of the distorted ellipse fitting result; (d) intermediate shapes in the distorted ellipse fitting process; (e) the manual segmentation (ground truth) of the CSF structure; and (f) the graph cuts segmentation of the CSF structure.

The description of the data and the method to generate a correlation map from the original image sequence is provided in Appendix A. Here we focus on our distorted ellipse fitting procedures, and how to use the active geometric shape model fitting results to generate a CSF segmentation.

4.1. Detecting the CSF structure

Example *in vivo* CSF PC-MR magnitude images are shown in Fig. 8. The correlation map of the CSF image sequence obtained by using (A.2) and (A.5) is displayed in Fig. 7a. We can see that the contrast is low and non-uniform around the CSF structure, which is located below the center of the correlation map. One solution for such defects in the image is to reduce the variation of the magnitude of the force field at different locations. We can rescale the magnitude with some monotonically non-decreasing function such as the power function. Here we simply normalize the force field vectors so that the magnitude is constant everywhere. By restricting the range of the distorted ellipse shape parameters a, b, and p according to medical knowledge, we can generate some random guess of these parameters as initial shape parameters. If information of the imaging procedure (such as posture) is provided, we are also able to generate some rough guess of the position parameters x_c and y_c as the initial position, which should be close to the true values. Otherwise, we can require a one-click input from the user for the initial position on the image; or we can fit the model many times using different randomly generated initial positions, and use the fitness function to select a best fit. Here we adopt the latter approach. For a distorted ellipse, the size parameter set is $\mathcal{P}_1 = \{a, b\}$, and the non-size parameter set is

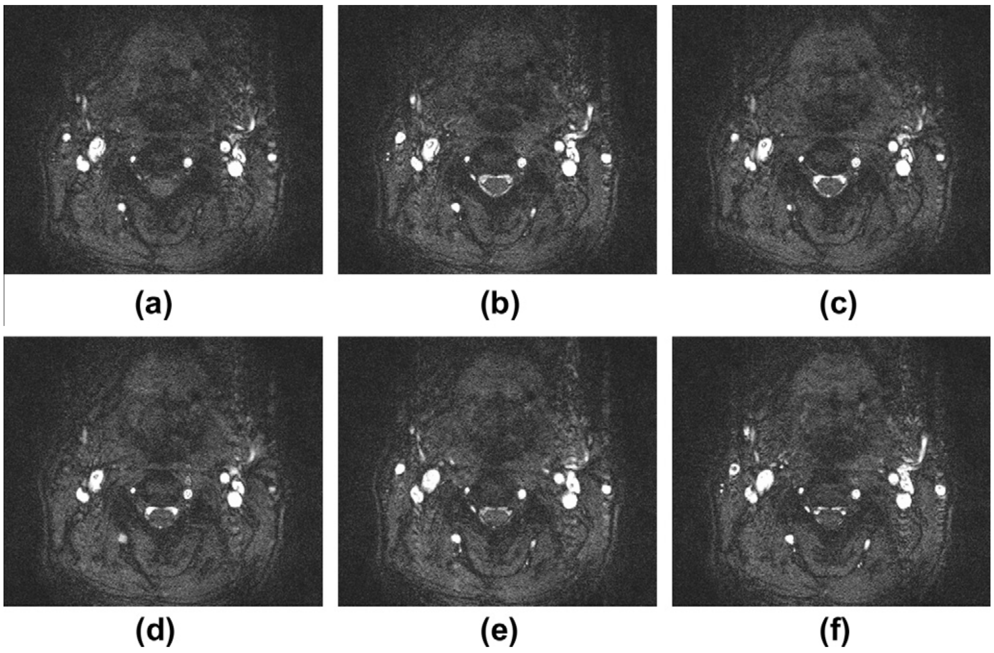


Fig. 8. Example *in vivo* CSF PC-MR magnitude images of one case. (a) Frame 0; (b) frame 4; (c) frame 8; (d) frame 12; (e) frame 16; and (f) frame 20.

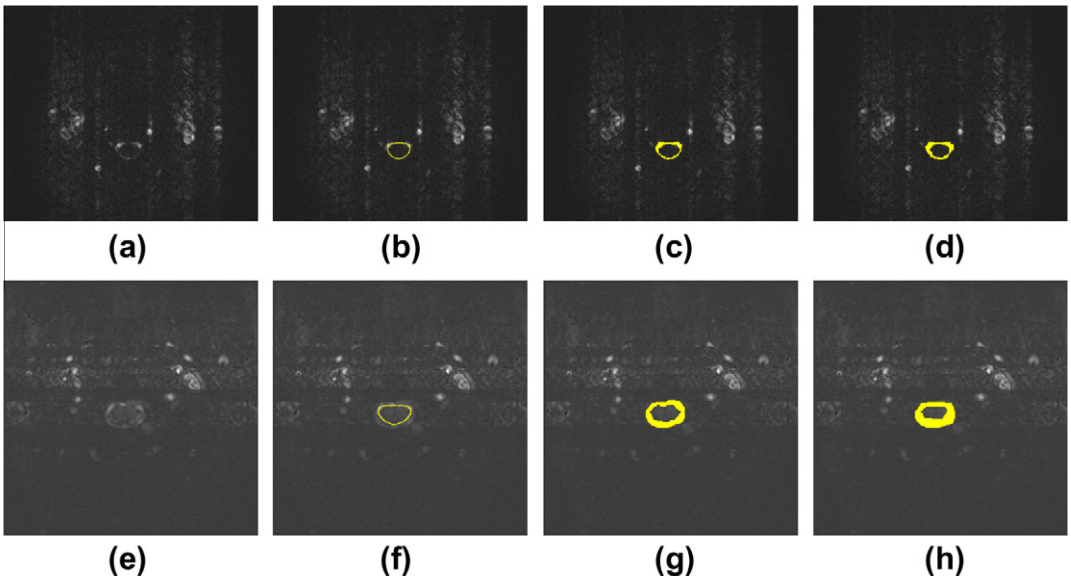


Fig. 9. Distorted ellipse fitting results and graph cuts segmentation results on two cases. Each row is one case. The first column is the correlation map; the second column is the distorted ellipse fitting result; the third column is the ground truth; and the last column is the graph cuts segmentation.

$\mathcal{P}_2 = \{x_c, y_c, p\}$. Similar to Eq. (84), the fitness function of the distorted ellipse model in the correlation map $C(x,y)$ is

$$\mathcal{F}(\mathcal{P}) = -\frac{1}{N}\sum_{i=1}^{N} C(x_i, y_i) + \frac{1}{2N'}\sum_{i=1}^{N'} C(x'_i, y'_i) + \frac{1}{2N''}\sum_{i=1}^{N''} C(x''_i, y''_i). \quad (90)$$

We consider a fit to be good if the value of this fitness function is small. Now we use the normalized GVF as our force field, use 50 randomly generated positions as the initial positions, set $\beta = 0.8$, use (90) to select the best fit, and set $\delta x_c = \delta y_c = 0.2, \delta a = \delta b = 0.2$, $\delta p = 0.002$.

Our resulting parameters of fitting the distorted ellipse model to the correlation map of the case shown in Figs. 7 and 8 are $x_c = 158.37$, $y_c = 93.67$, $a = 14.00$, $b = 5.60$ and $p = 1.716$. The fitting result of the above case is displayed in Fig. 7b and c. Fitting results of another two cases are shown in Fig. 9b and f.

For comparison, we have also tried to detect the CSF structure in the correlation map using active contours [23]. The results of three cases are shown in Fig. 10. In each case, we randomly initialize 40 active contours, and let them evolve in the correlation map. Some contours will evolve partly beyond the image border; and some contours with bad initialization will shrink to a single point and disappear. This is why some contours in Fig. 10 are not closed, and the number of contours is less than 40. Generally, we can see that the active contours fail to converge accurately to the CSF structure. This is because active contours contain no prior knowl-

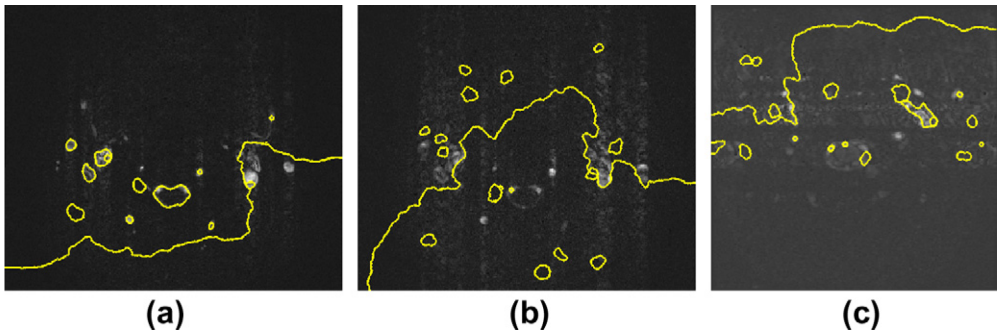


Fig. 10. Comparison: fitting active contours to the correlation maps shown in (a) Fig. 7a; (b) Fig. 9a and (c) Fig. 9e.

edge of a geometrical model. Besides, there is more than one bright region in the correlation map, and there is not a good criterion to select a best fit from all resulting contours.

4.2. Segmentation of the CSF structure

Now we need to segment the correlation map to two regions: the CSF structure and the background. Once we find a distorted ellipse fit in the correlation map, we can claim the pixels on the distorted ellipse contour to belong to the CSF structure. Besides, with a well chosen $0 < \beta' < 1$, $\mathcal{P}' = (\beta'\mathcal{P}_1) \cup \mathcal{P}_2$ and $\mathcal{P}'' = (\frac{1}{\beta'}\mathcal{P}_1) \cup \mathcal{P}_2$ represent a shrunken distorted ellipse and an expanded distorted ellipse respectively (see Section 2.10 for details), and we can also claim the pixels inside the shrunken distorted ellipse and outside the expanded distorted ellipse to belong to the background. Now the problem is to decide which region should the remaining pixels belong to. Boykov's *s*–*t* graph cuts algorithm provides a good solution to this problem [24]. The above mentioned pre-segmented pixels are used to construct the hard constraints (seed points) of the graph. Besides, to calculate the regional energy terms of the graph, the intensity distribution of each region is obtained by approximating the histograms of the two pre-segmented regions with Gaussian mixture models using EM algorithm. For each region, a Gaussian mixture model with three components is used. The relative importance of the regional energy term is set to $\lambda = 0.2$, the camera noise of the boundary energy term is set to $\sigma = 5$, and β' is set to 0.65. Details of the *s*–*t* graph cuts algorithm can be found in [24].

The graph cuts segmentation results of the CSF structures are shown in Figs. 7 and 9 together with the manual segmentations. Using the manual segmentation as the ground truth, our graph cuts segmentation achieves an average Dice similarity coefficient (DSC) of $\mu_{\text{DSC}} = 86.37\%$ on our dataset, while the standard deviation of the DSC is $\sigma_{\text{DSC}} = 3.70\%$. Note that we did not use any of the manual annotations for training (actually there is no training stage), thus this performance is already very satisfactory. Besides, since neither of the detection algorithm or the segmentation algorithm requires any manual input or labeling, our CSF detection–segmentation framework is completely automatic.

5. Discussion

Our active geometric shape model (AGSM) is a powerful approach to fit a geometric shape to image. Once we are able to represent the shape with parametric equations, we can associate the shape parameters with forces and torques defined as integrals of a force field along the shape. We have illustrated the accuracy and robustness of our model by fitting lines, ellipses and cubic spline contours on synthetic data. We also developed a distorted ellipse model to describe the shape of the CSF structure. This model provides an approach to detect the CSF structures in the PC-MR image sequences, and the *s*–*t* graph cuts algorithm based on the detection result is used to segment the CSF structure.

Unlike ASMs [1], our model does not require any manual annotation of landmark points of the shape. Since our active geometric shape model internally has the shape prior, and has a well defined fitness function coming with it, no training is needed for our method. Prior knowledge can be also added by constraining the range, ratio, or updating step of the shape parameters. We have also tried active contours [23] to detect the CSF structure in the correlation map. However, without prior knowledge, the active contours do not converge to the CSF structure robustly, since there is more than one bright region in the correlation map, and the CSF structure appears to be disconnected in the correlation map.

Although our work is mainly inspired by the CSF detection and segmentation problem [22], there are many other promising applications of the active geometric shape model, such as cell segmentation and tracking [25–28], 3D vessel extraction by fitting closed contours to cross-section planes [29], and photographic composition analysis [30–32]. Besides, in our CSF detection and segmentation problem, the shape parameters obtained by fitting the distorted ellipse model to the correlation map are promising features for the detection of pathologies such as hydrocephalus. Another future work is to generalize the active geometric shape model to the 3D space, such as models for spheres, ellipsoids and other closed surfaces.

Acknowledgments

The authors would like to thank Dr. Benjamin Cohen and Prof. Timothy Wei for kindly providing the image dataset and manual annotations of this research.

Appendix A. Correlation map

One case of *in vivo* CSF PC-MR image sequence consists of *N* magnitude images, which reveal the anatomy of the brain section, and *N* corresponding phase images. *N* can be 26 or 32, which differs between different image sequences. In those cases where $N = 26$, the pixel spacing is 0.52 mm, and the image size is 320×270; in the other cases where $N = 32$, the pixel spacing is 0.55 mm, and the image size is 256×256. Both magnitude images and phase images are 16-bit gray-level images. The frames of each image sequence are uniformly sampled over one cardiac cycle.

Since the cardiac cycle can be expected to approximate a sinusoid, we compute the correlation of the pixel intensity at each position in the magnitude image sequence with a sine function (reference function), and use this correlation map for further analysis. Now the number of images in one cardiac cycle is *N*, the reference function at frame *k* with phase ψ is

$$\sin\left(\frac{2k\pi}{N}+\psi\right). \quad (A.1)$$

Assume the original image sequence is denoted as $I_k(x, y)$, where (x, y) is the position and k is the frame number. Then the correlation function of the image sequence at (x, y) with a reference function of phase ψ is

$$C(x,y,\psi)=\frac{\sum_{k=1}^{N}\left(I_k(x,y)-\bar{I}(x,y)\right)\sin(\frac{2k\pi}{N}+\psi)}{\sqrt{\sum_{k=1}^{N}\left(\sin(\frac{2k\pi}{N}+\psi)\right)^2}}, \quad (A.2)$$

where $\bar{I}(x, y)$ is the mean of pixel intensities at (x, y) in the cycle. Note that here the correlation function is not normalized, so background noise that may also be highly correlated with a sine function, but with low amplitude, will be weak. To obtain the final correlation map, we need to choose a phase ψ at each point. Thus we can define the phase as a function of the position, denoted $\psi(x, y)$. Since the phase inside the CSF flow is expected to be continuous and smooth, we expect the magnitude of the spatial derivatives of $\psi(x, y)$ to be small and the correlation map $C(x, y, \psi(x, y))$ to be large at the same time. This can be formulated as an optimization problem:

$$\min_{\psi(x,y)}\iint\left(v(\psi_x^2+\psi_y^2)-C(x,y,\psi)\right)\,dx\,dy, \quad (A.3)$$

where ψ_x and ψ_y are the first order partial derivatives of $\psi(x, y)$ with respect to x and y respectively, and v is a scaling parameter. By using calculus of variations [33], the phase $\psi(x, y)$ can be obtained by solving the Euler–Lagrange equation:

$$\frac{\partial C(x,y,\psi)}{\partial\psi}+2v\nabla^2\psi=0. \quad (A.4)$$

The Euler–Lagrange equation can be solved using a simple numerical method:

$$\psi_{m+1}(x,y)=\psi_m(x,y)+\left(\frac{\partial C}{\partial\psi}+2v\nabla^2\psi_m\right), \quad (A.5)$$

where m is the iteration number. The resulting correlation maps with $v = 0.01$ are shown in Figs. 7a and 9a and e.

References

[1] T.F. Cootes, C.J. Taylor, D.H. Cooper, J. Graham, Active shape models-their training and application, Computer Vision and Image Understanding 61 (1995) 38–59.
[2] T.F. Cootes, G.J. Edwards, C.J. Taylor, Active appearance models, in: IEEE Transactions on Pattern Analysis and Machine Intelligence, Springer, 1998, pp. 484–498.
[3] M. Kass, A. Witkin, D. Terzopoulos, Snakes: active contour models, International Journal of Computer Vision 1 (1988) 321–331.
[4] C. Xu, J. Prince, Gradient vector flow: a new external force for snakes, in: Proceeding of the 1997 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 66–71.
[5] L. Staib, J. Duncan, Boundary finding with parametrically deformable models, IEEE Transactions on Pattern Analysis and Machine Intelligence 14 (1992) 1061–1075.
[6] P. Gotardo, K. Boyer, J. Saltz, S. Raman, A new deformable model for boundary tracking in cardiac MRI and its application to the detection of intra-ventricular dyssynchrony, in: 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, vol. 1, pp. 736–743.
[7] B. van Ginneken, A.F. Frangi, R.F. Frangi, J.J. Staal, B.M.T.H. Romeny, M.A. Viergever, Active shape model segmentation with optimal features, IEEE Transactions on Medical Imaging 21 (2002) 924–933.
[8] N. Paragios, R. Deriche, Geodesic active contours and level sets for the detection and tracking of moving objects, IEEE Transactions on Pattern Analysis and Machine Intelligence 22 (2000) 266–280.
[9] G. Edwards, T. Cootes, C. Taylor, Face recognition using active appearance models, in: H. Burkhardt, B. Neumann (Eds.), Computer Vision – ECCV'98, Lecture Notes in Computer Science, vol. 1407, Springer, Berlin/Heidelberg, 1998, pp. 581–589.
[10] Y. Wang, A. Narayanaswamy, C.-L. Tsai, B. Roysam, A broadly applicable 3-d neuron tracing method based on open-curve snake, Neuroinformatics 9 (2011) 193–217.
[11] T. McInerney, D. Terzopoulos, Deformable models in medical image analysis: a survey, Medical Image Analysis 1 (1996) 91–108.
[12] C.M. Bishop, Pattern Recognition and Machine Learning (Information Science and Statistics), Springer-Verlag, New York, Inc., Secaucus, NJ, USA, 2006.
[13] R.O. Duda, P.E. Hart, Use of the hough transformation to detect lines and curves in pictures, Communications of the ACM 15 (1972) 11–15.
[14] D.H. Ballard, Generalizing the hough transform to detect arbitrary shapes, Pattern Recognition 13 (1981) 111–122.
[15] C. Xu, J.L. Prince, Snakes shapes and gradient vector flow, IEEE Transactions on Image Processing 7 (1998) 359–369.
[16] J. Radon, On the determination of functions from their integral values along certain manifolds, IEEE Transactions on Medical Imaging 5 (1986) 170–176.
[17] J. Stoer, R. Bulirsch, Introduction to Numerical Analysis, Springer Science+Business Media, 2002.
[18] H. Bouma, A. Vilanova, L.J. van Vliet, F.A. Gerritsen, Correction for the dislocation of curved surfaces caused by the psf in 2d and 3d ct images, IEEE Transactions on Pattern Anaysis and Machine Intelligence 27 (2005) 1501–1507.
[19] M. Abramowitz, Handbook of Mathematical Functions, with Formulas, Graphs, and Mathematical Tables, Dover Publications, Incorporated, 1974.
[20] D.M. Wuescher, K.L. Boyer, Robust contour decomposition using a constant curvature criterion, IEEE Transactions on Pattern Analysis and Machine Intelligence 13 (1991) 41–51.
[21] D.A. Forsyth, J. Ponce, Computer Vision: A Modern Approach, Prentice Hall Professional Technical Reference, 2002.
[22] B. Cohen, A. Voorhees, S. Vedel, T. Wei, Development of a theoretical framework for analyzing cerebrospinal fluid dynamics, Cerebrospinal Fluid Research 6 (2009). 2+.
[23] C. Li, C. Xu, C. Gui, M. Fox, Distance regularized level set evolution and its application to image segmentation, IEEE Transactions on Image Processing 19 (2010) 3243–3254.
[24] Y. Boykov, G. Funka-Lea, Graph cuts and efficient n–d image segmentation, International Journal of Computer Vision 70 (2006) 109–131.
[25] D.H. Kim, U.K. Cheang, L. Kohidai, D. Byun, M.J. Kim, Artificial magnetotactic motion control of Tetrahymena pyriformis using ferromagnetic nanoparticles: A tool for fabrication of microbiorobots, Applied Physics Letters 97 (2010) 173702.
[26] Y. Ou, D.H. Kim, P. Kim, M.J. Kim, A.A. Julius, Motion control of tetrahymena pyriformis cells with artificial magnetotaxis: model predictive control (MPC) approach, in: 2012 IEEE International Conference on Robotics and Automation, St. Paul, USA.
[27] Q. Wang, Y. Ou, A.A. Julius, K.L. Boyer, M.J. Kim, Tracking tetrahymena pyriformis cells using decision trees, in: 2012 International Conference on Pattern Recognition, Tsukuba Science City, Japan.
[28] E. Meijering, O. Dzyubachyk, I. Smal, W.A. van Cappellen, Tracking in cell and developmental biology, Seminars in Cell & Developmental Biology 20 (2009) 894–902.
[29] Y. Wang, A. Narayanaswamy, B. Roysam, Novel 4-d open-curve active contour and curve completion approach for automated tree structure extraction, in: 2011 IEEE Conference on Computer Vision and Pattern Recognition, pp. 1105–1112.
[30] L. Yao, P. Suryanarayan, M. Qiao, J. Wang, J. Li, Oscar: On-site composition and aesthetics feedback through exemplars for photographers, International Journal of Computer Vision 96 (2012) 353–383.
[31] M. Freeman, The Photographer's Eye: Composition and Design for Better Digital Photos, Focal Press, 2007.
[32] D. duChemin, Photographically Speaking: A Deeper Look at Creating Stronger Images, Voices That Matter, Pearson Education, 2011.
[33] R. Courant, D. Hilbert, Methods of Mathematical Physics, vol. 1, Interscience Publishers, Inc., New York, NY, 1953.