Vector bundle constraint for particle swarm optimization and its application to active contour modeling*

Zeng Delu, Zhou Zhiheng and Xie Shengli**

(College of Electronic and Information Engineering, South China University of Technology, Guangzhou 510641, China)

Accepted on April 27, 2007

Abstract Active contour modeling (ACM) has been shown to be a powerful method in object boundary extraction. In this paper, a new ACM based on vector bundle constraint for particle swarm optimization (VBCPSO-ACM) is proposed. Different from the traditional particle swarm optimization (PSO), in the process of velocity update, a vector bundle is predefined for each particle and velocity update of the particle is restricted to its bundle. Applying this idea to ACM, control points on the contour are treated as particles in PSO and the evolution of the contour is driven by the particles. Meanwhile, global searching is shifted to local searching in ACM by decreasing the number of neighbors and inertia. In addition, the addition and deletion of particles on the active contour make this new model possible for representing the real boundaries more precisely. The proposed VBCPSO-ACM can avoid self-intersection during contour evolving and also extract inhomogeneous boundaries. The simulation results proved its great performance in performing contour extraction.

Keywords: boundary extraction, active contour modeling, particle swarm optimization, velocity update, vector bundle.

Since the introduction of snakes by Kass et al.[1], active contour modeling (ACM) has a wide variety of applications in image processing and computer vision, such as segmentation, reconstruction, shape analysis and visual tracking. In the past two decades, various algorithms based on ACM have been put forward[2—7]. Among these contributions, the invention of the level set methods[6] is a breakthrough for changing the topology of evolving curve, and then makes the automatism of multiple objects segmentation possible.

ACM was originally achieved from minimizing some energy functional. But the minimization often sticks into local minima, particularly in the case of long concavities and weak boundaries. Xu et al.[7] proposed a gradient vector flow (GVF) snake, in which the GVF is to take the place of external force to attract the snake to the real boundaries. To some extent, this scheme can solve the concavities case, while fails in dealing with the long ones.

To overcome the weak boundaries leakage, Chan et al.[4] derived a region and edge-based energy functional from the mumford-shah functional[8], utilizing not only the image information near the evolving contour, but also the image statistics inside and outside the contour to yield more robust performance. However, when there are more than two averages, this kind of methods, although there had been much improvement in Refs. [9—11], still bring on high computational costs for computing the level set functions.

Self organizing map (SOM)-based ACM[12—14] with competitive learning can solve the problem in a certain degree. However, SOM-based ACM is feasible only after a set of feature points from the edge map has been obtained. So it is necessary to do edge detection before ACM. Moreover, neurons cannot share their information with each other to search the real objects boundaries.

Particle swarm optimization (PSO)[15] is a new kind of optimization method. In a PSO paradigm, particles transmit information among themselves in order to search the optimal positions.

However, since the traditional PSO only permits all the particles in the image space to search for the best positions if measured by gradient and intensities, the particles will not approach the real contour but tend to flock together and then the contour driven by

* Supported by National Fund for Distinguished Young Scholars (Grant No. 60325310), Guangdong Province Science Foundation (Grant Nos. 04205783, 07006490), Specialized Prophasic Basic Research Projects of Ministry of Science and Technology, China (Grant No. 2005CCA04100), China Postdoctoral Science Foundation (Grant No. 20060390728) and the Key Project of National Natural Science Foundation of China (Grant No. U0635001)

** To whom correspondence should be addressed. E-mail: adshlxie@scut.edu.cn

them will possibly self-intersect. On the other hand, as a result of illumination, quality of image and etc., the features of object boundaries, i.e. gradient and intensities, may vary in different parts. This phenomenon is called inhomogeneous boundaries here. In this case, the particles on different parts of the boundaries cannot be optimized simultaneously. Consequently, we cannot simply apply the traditional PSO to ACM.

In this paper, we present a vector bundle constraint for PSO, called VBCPSO, and its application to ACM. The VBCPSO-ACM can solve the self-intersection problem well and make it possible for the inhomogeneous boundaries extraction, where velocity update for each of the particles is restricted in the corresponding predefined vector bundle.

1 Traditional particle swarm optimization

Particle swarm optimization (PSO)[15] is a population-based evolutionary computation method inspired from the social behavior of birds flocking, fish schooling, bees swarming and etc. The main principle of PSO is that each of the particles (members of the population), which are treated as candidate solutions of an optimization problem, flies through the search space to get to its optimal position, with the velocity dynamically adjusted according to its own flying experience and that of its neighbors. In the PSO mechanism, each particle owns the following four attributes: current velocity, current position, the best position it discovered so far, the best position discovered by its neighbors so far.

Suppose that there are N particles in the population, then the updates of velocities and positions are described as follows:

$$\begin{aligned} v_i(t+1) = & w(t) \circ v_i(t) \\ & + c_1 \circ \varphi_1 \circ (u_i(t) - x_i(t)) \\ & + c_2 \circ \varphi_2 \circ (u_{n(i)}(t) - x_i(t))) \end{aligned} \quad (1)$$

$$x_i(t+1) = x_i(t) + v_i(t+1) \quad (2)$$

where $w(t)$, $v_i(t)$ and $x_i(t)$, $(i=1, \cdots, N, d=1, \cdots, M)$ are inertia, velocity, current position, respectively; $n(i)$ denotes the neighborhood for the ith particle; $u_i(t)$ and $u_{n(i)}(t)$, $(i=1, \cdots, N, d=1, \cdots, M)$ are the best discovered position and the best discovered position by the neighbors of the ith particle at epoch t under some fitness function, respectively; and c_1 and c_2 are acceleration constants; φ_1 and φ_2 are random numbers.

Typically, the inertia $w(t)$ is chosen to be a linearly decreasing function. The decreasing inertia over time will cause the particle to shift from global searching to local searching.

2 VBCPSO-ACM algorithm

In this section, the algorithm of vector bundle constraint for PSO and its application to ACM (VBCPSO-ACM) are proposed in detail.

Give some control points with a chain topology, and suppose that they are connected by line segments in the image space. Then we use the idea of PSO to drive the control points, which are treated as particles, along with the line segments to search for the real contour of the desired object. The control points with chain topology play the role of an active contour in ACM. In the following, the control points and the particles are not treated discriminatively unless specified.

In an $H \times W$ image, given a fitness function f, the active contour is modeled by a population of $N(t)$ control points $\{p_i(t)\}_{i=1}^{N(t)}$, where $N(t)$ is gross of the population at epoch t, and $p_i(t)$ owns the following four attributes: $x_i(t)=(x_i(t), y_i(t))$ is its 2-D position, $v_i(t)=(v_{i,1}(t), v_{i,2}(t))$ is its 2-D velocity, $u_i(t)=(u_{i,1}(t), u_{i,2}(t))$ is its best position discovered so far, $u_{n(i)}(t)=(u_{n(i),1}(t), u_{n(i),2}(t))$ is the best position discovered so far by its neighbors.

2.1 Vector bundle

Suppose that in a 2-D space, $\boldsymbol{V}_i$ and $\boldsymbol{V}_j$ are two vectors, which have the common starting points O. Rotating $\boldsymbol{V}_i$ to $\boldsymbol{V}_j$ clockwise with respect to O, we obtain a sector. So vector bundle $\boldsymbol{S}_{i\to j}$ as shown in Fig. 1 (case I and case II) is defined as the set of vectors sampled from this sector and sharing the common starting point O.

In the following, we show in detail how to judge whether a vector $\boldsymbol{V}$ belongs to the vector bundle $\boldsymbol{S}_{i\to j}$ constructed by vectors $\boldsymbol{V}_i$ and $\boldsymbol{V}_j$.

In order to handle this problem conveniently, we solve it in the corresponding complex plane $Z(O)$ with the origin O. That is, if the three corresponding complex numbers are still denoted as $\boldsymbol{V}$, $\boldsymbol{V}_i$ and $\boldsymbol{V}_j$, then we have: If $0 \leqslant \theta \leqslant \theta_{i\to j}$, where

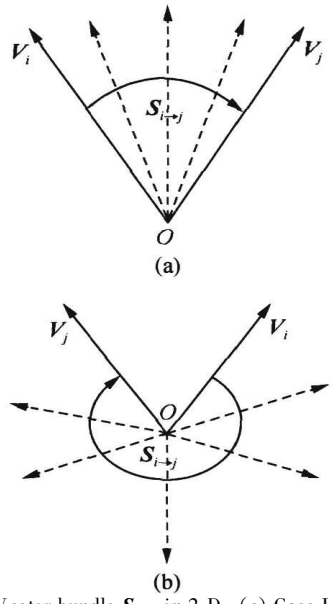

Fig. 1. Vector bundle $S_{i\to j}$ in 2-D. (a) Case I; (b) case II.

$$\theta = \arg\left(\frac{V_i}{V}\right), \quad \theta_{i\to j} = \arg\left(\frac{V_i}{V_j}\right), \quad V_i = \frac{V_i}{\| V_i \|},$$
$$V_j = \frac{V_j}{\| V_j \|}, \quad V = \frac{V}{\| V \|},$$

$\arg(\cdot)$ is the principal value of arguments of complex numbers, then V belongs to the vector bundle $S_{i\to j}$. Otherwise, V does not belong to the vector bundle $S_{i\to j}$.

2.2 Velocity selection for particles

We intend to drive the active contour to approach the real boundary by moving the particles on it. So it is critical to design the velocities for those particles. However, since the particles always search for the global optimal positions, if we directly use the traditional PSO, it may drive the line segments connected those particles to self-intersect (Fig. 2), then the subsequent particles movement will become more and more chaotic and end up with active contours of disorder. This phenomenon is unfavorable in ACM. Thereby, we should make an adaptation in the velocity update procedure.

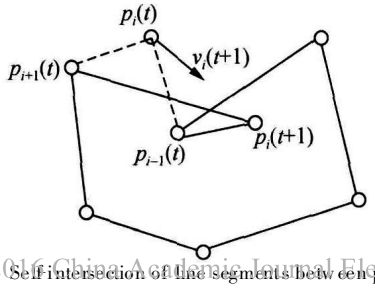

Fig. 2. Self-intersection of line segments between particles.

In order to eliminate this chaos, we present the following scheme for velocity selection of the particles. For each particle $p_i(t)$ at epoch t, $i = 1, \cdots, N(t)$, we restrict its next velocity $v_i(t+1)$ in the vector bundle $S_{i-1\to i+1}$. In Fig. 3, the vector bundle $S_{i-1\to i+1}$ is sampled from the sector formed by rotating the vector V_{i-1} clockwise to V_{i+1}, e.g., the velocity $v_i'(t+1) \in S_{i-1\to i+1}$ is one of the feasible choice of velocity for $p_i(t)$.

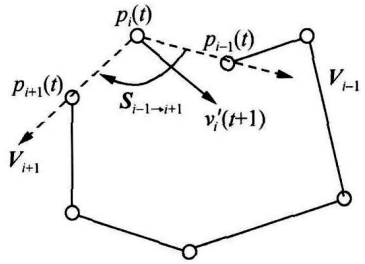

Fig. 3. The vector bundle $S_{i-1\to i+1}$ for particle $p_i(t)$.

For particle $p_i(t)$, if the velocity computed by Eq. (1) is not in the vector bundle $S_{i-1\to i+1}$, we replace $u_{n(i)}(t)$ in Eq. (1) with the best position of other particles until this replacement causes the velocity $v_i(t+1)$ to belong to $S_{i-1\to i+1}$. Note that, for $p_1(t)$ we have $S_{N\to 2}$, while for $p_N(t)$ we have $S_{N-1\to 1}$.

Consequently, give an inertia

$$w(t) = \left(1 - \frac{t}{T_{\max}}\right) \cdot w_{\text{satrt}} + \frac{t}{T_{\max}} \cdot w_{\text{end}} \quad (3)$$

where $T_{\max}$, w_{start}, and w_{end} are the iteration time, starting inertia, and ending inertia, respectively. And supposing that $\{x_i(t), v_i(t), u_i(t), u_{n(i)}(t)\}_{i=1}^{N(t)}$ are current information for the population $\{p_i(t)\}_{i=1}^{N(t)}$, we summarize the selection procedure of $v_i(t+1)$ as follows:

(i) Initialize the acceleration constants c_1 and c_2, produce random numbers φ_1 and φ_2 from a uniform distribution between 0.0 and 1.0.

(ii) Rank $\{u_i(t)\}_{i=1}^{N(t)}$ according to their fitness on the fitness function f. Thus we have

$$f(u_{n(i)}(t) = u_{i_1}(t)) \geqslant f(u_{i_2}(t)) \geqslant \cdots \geqslant f(u_{i_{N(t)}}(t))$$

(iii) For particle $p_i(t)$, find the range of vector bundle $S_{i-1\to i+1}$. And set a counter $k = 0$.

(iv) Let $k=k+1$, If $k>N(t)$, turn to (vi); otherwise, compute $\bar{v}_i(t+1)$ by the following equation:

$$\bar{v}_i(t+1) = w(t) \circ v_i(t) + c_1 \circ \varphi_1 \circ (u_i(t) - x_i(t)) + c_2 \circ \varphi_2 \circ (u_{i_k}(t) - x_i(t))) \quad (4)$$

(v) If $\bar{v}_i(t+1) \in \boldsymbol{S}_{i-1 \to i+1}$ set $v_i(t+1) = \frac{\bar{v}_i(t+1)}{\|\bar{v}_i(t+1)\|}$, turn to (vii); else turn to (iv).

(vi) There is no $\bar{v}_i(t+1)$ such that $\bar{v}_i(t+1) \in \boldsymbol{S}_{i-1 \to i+1}$, set $v_i(t+1)=0$, turn to (vii).

(vii) Velocity selection is finished.

Note that, in the step (v), $\|\bar{v}_i(t+1)\|$ is the Euclid norm of $\bar{v}_i(t+1)$. The normalization of $\bar{v}_i(t+1)$ before the assignment to $v_i(t+1)$ is to avoid the explosion of velocity.

2.3 Outline of the proposed algorithm

An outline of the proposed VBCPSO-ACM algorithm is summarized in the following steps:

(i) Parameters initialization: initialization of acceleration constant c_1 and c_2, iteration time $T_{\max}$, starting inertia w_{start} and ending inertia w_{end}.

(ii) Position initialization: initialize the active contour as a chain topology, then number the nodes (particles) counterclockwise, and let $N(0)$ be the gross of the particles. For each $i=1, \cdots, N(0)$, let $x_i(0)$ be the initial position of the particles at epoch $t=0$.

(iii) Velocity initialization: for each $i=1, \cdots, N(0)$, let $v_i(0)$ be random unit vectors, the components of which are non-negative, thus each particle is assigned by a random speed at epoch $t=0$.

(iv) Compute the best position for each particle and the one among the population: for all i, set $u_i(0)=x_i(0)$ and $u_{n(i)}(0)=\arg\max_{x\ i=1,\cdots,N(0)} f(x_i(0))$.

(v) Compute the velocity and position at epoch $t=1$ according to

$$v_i(1) = w_{\text{start}} \circ v_i(0) + c_2 \circ \varphi_2 \circ (u_{n(i)}(0) - x_i(0)) \quad (5)$$

and

$$x_i(1) = x_i(0) + v_i(1) \quad (6)$$

(vi) Update the best position $u_i(t)$ for each particle and the best position $u_{n(i)}(t)$ among its neighbors at epoch t.

(vii) Update velocity for each particle at the next epoch $t+1$ according to the scheme described in section 2.2.

(viii) Update the position for each particle at epoch $t+1$ as follows:

$$x_i(t+1) = x_i(t) + v_i(t+1) \quad (7)$$

(ix) Particle insertion and deletion.

(x) If $t=T_{\max}$ the VBCPSO-ACM finishes; otherwise turns to step (vi).

In step (i), we initialize the parameters $T_{\max}$, c_1 and c_2 as usual as the traditional PSO paradigm.

In step (ii), we initialize a chain of particles large enough to encircle the desired object in the observed image. If no *a priori* information of the observed objects is available, set the initial chain to be a rectangle a bit smaller than the image (Fig.4).

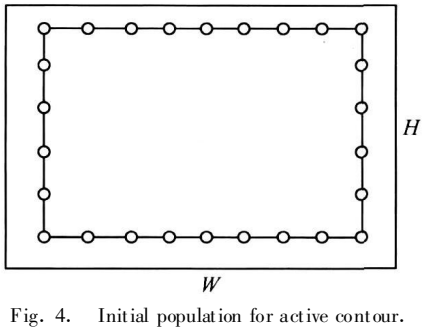

Fig. 4. Initial population for active contour.

In step (iii) and (vii), the velocity should not cause the particles to move out of the image.

In step (vi) the number of neighbors for the particle is usually chosen to be decreasing. Since the velocity of each particle is computed from the best position discovered by its neighbors, the decreasing number of neighbors will naturally cause the fact that the velocity gradually shifts to dependence on a shrinking subset of the population. Meanwhile the population shifts from global searching to local searching.

Usually, we choose a circle topology for the neighborhood while the number of neighborhood at epoch t is:

$$N(n(i), t) = \left(1 - \frac{t}{T_{\max}}\right) \circ (N(t) - 1) + \frac{t}{T_{\max}} \circ 2 \quad (8)$$

where $N(t)-1$ and 2 are the number of members in the neighborhood $n(i)$ for the ith particle at the beginning and in the end, respectively.

In step (ix), we can set two constants d_{max} and d_{min}, where $d_{max} > d_{min} > 0$.

If the distance between any two adjacent particles is larger than d_{max}, then a new particle is inserted between them. For example, for the particles $p_i(t)$ and $p_{i+1}(t)$ such that $|p_i(t)-p_{i+1}(t)| > d_{max}$, both the position and the best discovered position of the newly inserted particle are $\frac{p_i(t)+p_{i+1}(t)}{2}$, and the best discovered position by its neighbors is $\max(u_{n(i)}(t), u_{n(i+1)}(t))$.

If the distance such that $|p_i(t)-p_{i+1}(t)| \leqslant d_{min}$, the suboptimal one of the two particles is deleted. In both the above insertion and deletion, the topology of the chain should be maintained. This paradigm aims to build a contour with uniformly distributed particles and control the accuracy of the approximation.

3 Implementation

In this section, we give some implementations of the proposed VBCPSO-ACM. During the active contour evolution, the line segments between particles are helpful to represent the real boundaries while the evolving contour has approached the real boundaries. For this reason, it is not necessary to move the segments, and they can be computed by the positions of the particles. Consequently, in the implementation below, we do not draw the line segments before the convergence of the active contour. Moreover, we only use the gradient information as the fitness function in the proposed VBCPSO-ACM.

The first implementation shown in Fig. 5 is the case of a hand image, in which long concavities appear on the boundary. In this implementation, $N(0)=48$ particles are set to be the initial population. The values of other parameters are: $c_1=c_2=2$, $w_{start}=0.9$, $w_{end}=0.4$, $T_{max}=40$, $d_{max}=4$, $d_{min}=2$. With particle insertion and deletion, finally $N(T_{max})=123$ particles and the segments between them are used to represent the real boundary.

The second implementation shown in Fig. 6 is the case of a contour modeling for a plane under the inhomogeneous background. Note that the boundary of the plane has different features at different parts. The parameters are as follows: $c_1=c_2=2$, $w_{start}=0.9$, $w_{end}=0.4$, $T_{max}=72$, $d_{max}=4$, $d_{min}=2$. With $N(0)=52$ particles used on the initial contour, after the implementation of the proposed algorithm, $N(T_{max})=58$ particles are used to represent the real boundary.

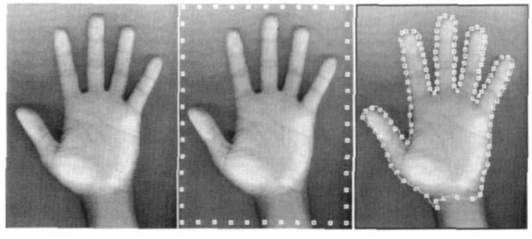

Fig. 5. Implementation of a 128× 164 hand image. (a) Original image; (b) 48 particles on the initial contour; (c) results.

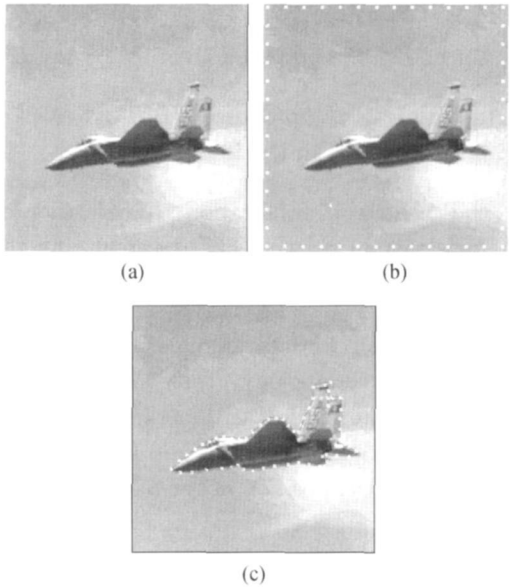

Fig. 6. Implementation of a 256× 256 plane image. (a) Original image; (b) 52 particles on the initial contour; (c) results.

In the end, we apply the proposed VBCPSO to the case of multiple contours modeling in Fig. 7. The parameters are as follows: $c_1=c_2=2$, $w_{start}=0.9$,

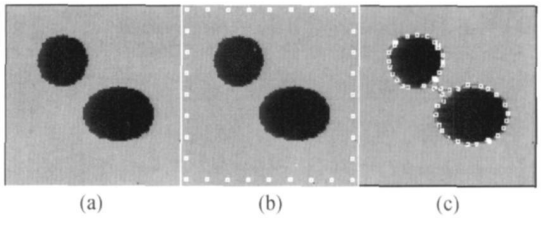

Fig. 7. Implementation on two objects. (a) 128× 128 original image; (b) 32 particles on the initial contour; (c) results.

$w_{end}=0.4$, $T_{max}=37$, $d_{max}=3$, $d_{min}=1$, $N(0)=32$, $N(T_{max})=46$. The results are good except that there are some parts of the contour between the desired objects.

4 Conclusions

In this paper, a modified particle swarm optimization for active contour modeling (VBCPSO-ACM) is proposed.

In comparison with the SOM-ACM, we do not use any feature points to attract the contour in VBCPSO-ACM. Therefore, we can omit the step of preprocessing edge detection for the observed image.

According to the characteristics of contour modeling, a vector bundle for velocity update of each particle is defined first. Then in the velocity update scheme, the velocity is restricted in the corresponding vector bundle and decided on the best discovered positions of other particles as well.

As a consequence, the self-intersection problem is eliminated to maintain the contour topology. Moreover, the VBCPSO-SOM is capable for inhomogeneous boundary extraction.

In the process of iteration, we also insert and delete particles according to the adjacent distance of the particles to keep the precision of approaching the real contour. What is more, in the neighboring function, the decreasing number of neighbors causes the population to shift from global searching to local searching.

The implementations show that the proposed VBCPSO can perform contour extraction greatly in the case of long concavities, inhomogeneous background, inhomogeneous boundary and multiple contour extractions.

5 Discussions and future work

In this paper, the neighboring function for the number of neighboring particles is determined on a monotonously decreasing function (Eq. (8)). In fact, it is more favorable to consider the continuity of the information (including gradient and intensities) of the neighboring particles to design the number of its neighbors since each peace of boundaries is continuous. What is more, we only use the gradient of image to be the fitness function. In the future work, we are going to explore some more appropriate neighboring function and better fitness function to seek for more robust performance.

References

1 Kass M, Witkin A and Terzopoulos D. Snakes: Active contour models. Int'l J Computer Vision, 1988, 1: 321—332
2 Caselles V, Kimmel R and Sapiro G. Geodesic active contours. In: Proc 5th IEEE Int'l Conf. Computer Vision, 1995, 694—699
3 Paragios N and Deriche R. Geodesic active contours and level sets for the detection and tracking of moving objects. IEEE Trans on Pattern Analysis and Machine Intelligence, 2000, 22(3): 266—280
4 Chan TF and Vese LA. Active contours without edges. IEEE Trans on Image Processing, 2001, 10(2): 266—277
5 Malladi R, Sethian JA and Vemuri BC. Shape modeling with front propagation: a Level set approach. IEEE Trans on Pattern Analysis Machine Intelligence 1995, 17(2): 158—175
6 Xu C and Prince JL. Snakes, shapes and gradient vector flow. IEEE Trans Image Processing, 1998, 7(3): 359—369
7 Osher SJ and Sethian JA. Fronts propagating with curvature-dependent speed: Algorithms based on Hamilton-Jacobi formulations. J Comp Phys, 1988, 79(1): 12—49
8 Mumford D and Shah J. Optimal approximation by piecewise smooth functions and associate variatioal problems. Comm Pure Appl Math, 1989, 42: 577—685
9 Vese LA and Chan TF. A multiphase level set framework for image segmentation using the Mumford and Shah model. UCLA Department of Mathematics, CAM Report 01-25. Int'l J Computer Vision, 2001
10 Tsai A, Yezzi A and Willsky AS. Curve evolution implementation of the Mumford-Shah functional for image segmentation, denoising, interpolation and magnification. IEEE Trans on Image Processing, Aug 2001, 10(8): 1169—1186
11 Gao S and Tien DB. Image segmentation and selective smoothing by using Mumford-Shah model. IEEE Trans On Image Processing, 2005, 14(10): 1537—1549
12 Venkatesh YV and Rishikesh N. Self organizing neural networks based on spatial isomorphism for active contour modeling. Pattern Recognition, 2000, 33: 1239—1250
13 Hamed SH and Reza S. A TASOM-based algorithm for active contour modeling. Pattern Recognition Letters. 2003, 24: 1361—1373
14 Venkatesh YV, Kumar Raja S and Ramya N. Multiple contour extraction from gray level images using an artificial neural network. IEEE Trans on Image Processing, 2006, 15(4): 892—899
15 Kennedy J and Eberhart R. Particle swarm optimization. In: Proceedings of IEEE International Conference on Neural Networks, Perth, Australia, 1995, 4: 1942—1948