Gradient vector flow using an implicit method

Wei-gang Chen

School of Computer and Information Engineering, Zhejiang Gongshang University, Hangzhou 310035

gary302a2003@yahoo.com.cn

Abstract

Gradient vector flow is an efficient external force for active contours, which can largely solve the problems such as poor convergence to boundary concavities, leaking from discontinuous boundary and miserable anti-noise ability of the active contour models. A detailed analysis of the original method of Xu et al. is proposed. Due to the fact that GVF gives rise to a stiff and very large system of initial-value problem for ordinary differential equations, an implicit method which has a larger stability region is proposed. The efficacy of the scheme is demonstrated with numerical experiments on synthesized and real medical images.
Keywords: Active contour models, gradient vector flow, stiff equations, Gear method

I. Introduction

Image segmentation serves as a key step in many applications for identifying homogeneous regions in a scene. As a boundary-based algorithm, active contour models have grown significantly since the seminal work of Kass et al. for segmentation and contour extraction.

Active contour models are broadly classified as either parametric models or geometric models according to their representation and implementation. An active contour is a deformable continuous curve whose shape is controlled by internal continuity forces and external image forces. Internal forces act as a smoothness constraint, and external forces guide the active contour toward an object boundary or other desired features within an image.

Xu and Prince developed an external force, called gradient vector flow(GVF)[1,2]. It is a bi-directional external force which is computed as the diffusion of the edge-driven information of a gray-level image. Recently, several models of active contours which were efficiently combined with GVF field were proposed for boundary extraction[3,4]. The reported results show such an external force can capture the object boundaries from either sides, can deal with the problems such as poor convergence to boundary concavities, leaking from discontinuous boundary and miserable anti-noise ability of the active contour models.

However, GVF gives rise to a stiff and very large system of initial-value problem for ordinary differential equations(ODEs). In Section 2, we give a detailed analysis of the original method of Xu et al., and point out that it is indeed an explicit Euler schema for initial-value problem for ODEs. It is first-order accuracy. Therefore, the time step must be kept small to guarantee stability and accuracy. In Section 3, we propose an implicit method which has a larger stability region for calculating the GVF fields. Experimental results are presented in Section 4, and make comparison with the results using the method proposed by Xu et,al. The conclusion is given in Section 5.

II. Gradient vector flow

As a type of external force, GVF fields are calculated by applying generalized diffusion equations to both components of the gradient of an image edge map. This computation causes diffuse force vectors to exist far from the object, and crisp force vectors near the edges. One can interpret GVF as the direction to be followed to reach the desired boundaries.

A. Definition and numerical implementation

Let $I:[0,m_1]\times[0,m_2]\to \mathbf{R}^+$ be a given image. Let the edge map derived from the image be:

$$f(x,y)=\left|\nabla\left[G_\sigma(x,y)*I(x,y)\right]\right|^2$$

where G_σ is a two-dimensional Gaussian kernel with standard deviation σ, and $|\nabla|$ denotes the norm of gradient.

Xu et al. defined the gradient vector flow as a two-dimensional vector field $\mathbf{u}(x,y)=\left[u(x,y),v(x,y)\right]$ that minimized the following objective function[1, 2]:

$$E(\mathbf{u})=\iint \mu\left(|\nabla f|\right)|\nabla \mathbf{u}|^2+h\left(|\nabla f|\right)|\mathbf{u}-\nabla f|^2 \qquad (1)$$

The first term on the right is a regularization term which dominates the functional in areas far from the edges. $\mu(\cdot)$ is a weighting function governing the tradeoff between the regularization term and data term(the second term). μ is large, the solution yields a smooth field, whereas the solution preserves close to ∇f. The second term on the right is a data-driven component which dominates the functional in the vicinity of the object boundaries. $h(\cdot)$ is a non-decreasing function of ∇f. Large value h encourages the vector field $\mathbf{u}$ to be close to ∇f.

As proposed in [1], one can optimize $\mathbf{u}$ by seeking the steady-state solution to the following time-dependent partial differential equations(PDEs).

$$\begin{cases} u_t=\mu\nabla^2 u-(u-f_x)|\nabla f|^2 \\ v_t=\mu\nabla^2 v-(v-f_y)|\nabla f|^2 \end{cases} \qquad (2)$$

where ∇^2 is the Laplacian operator.

One way to solve the time-dependent PDEs in Eq.(2) is to discretize in space but leave the time variable continuous[5]. This approach results in a system of ODEs, which can then be solved by a numerical method for ODEs.

Let the spacing between pixels be $\Delta x, \Delta y$. Then $\nabla^2 u$ and $\nabla^2 v$ at node (i, j) can be replaced with the finite difference approximation:

$$\nabla^2 u = \frac{1}{\Delta x \Delta y}\left(u_{i+1,j} + u_{i,j+1} + u_{i-1,j} + u_{i,j-1} - 4u_{i,j}\right)$$

$$\nabla^2 v = \frac{1}{\Delta x \Delta y}\left(v_{i+1,j} + v_{i,j+1} + v_{i-1,j} + v_{i,j-1} - 4v_{i,j}\right)$$

Let

$$b(x,y) = f_x(x,y)^2 + f_y(x,y)^2$$

$$c^1(x,y) = b(x,y) f_x(x,y)$$

$$c^2(x,y) = b(x,y) f_y(x,y)$$

Substituting those three and the approximations of $\nabla^2 u$ and $\nabla^2 v$ into Eq.(2), we obtain an $N_1 \times N_2$ system of ODEs:

$$\frac{du_{i,j}}{dt} = -\left(\frac{4\mu}{\Delta x \Delta y} + b_{i,j}\right)u_{i,j} + \frac{\mu}{\Delta x \Delta y}\left(u_{i+1,j} + u_{i,j+1} + u_{i-1,j} + u_{i,j-1}\right) + c^1_{i,j} \tag{3}$$

$$\frac{dv_{i,j}}{dt} = -\left(\frac{4\mu}{\Delta x \Delta y} + b_{i,j}\right)v_{i,j} + \frac{\mu}{\Delta x \Delta y}\left(v_{i+1,j} + v_{i,j+1} + v_{i-1,j} + v_{i,j-1}\right) + c^2_{i,j} \tag{4}$$

Where $N_1 = \frac{m_1}{\Delta x}$, $N_2 = \frac{m_2}{\Delta y}$, and $(i, j) \in (1,...,N_1) \times (1,...,N_2)$.

Thus the solution of the GVF field is constructed with respect to time by integrating a family of ODEs in which the k th unknown is the time course of $\mathbf{u}_{i,j}(t)$, where $k = (j-1) \times N_1 + i$. Let Δt be the discretization of time, and $t_n = n\Delta t$. With the initial conditions $\mathbf{u}(t_0) = \nabla f$, the system of ODEs can be solved using Euler's method.

$$u^{(n+1)}_{i,j} = \left(1 - 4r - b_{i,j}\Delta t\right)u^{(n)}_{i,j} + r\left(u^{(n)}_{i+1,j} + u^{(n)}_{i,j+1} + u^{(n)}_{i-1,j} + u^{(n)}_{i,j-1}\right) + c^1_{i,j}\Delta t \tag{5}$$

$$v^{(n+1)}_{i,j} = \left(1 - 4r - b_{i,j}\Delta t\right)v^{(n)}_{i,j} + r\left(v^{(n)}_{i+1,j} + v^{(n)}_{i,j+1} + v^{(n)}_{i-1,j} + v^{(n)}_{i,j-1}\right) + c^2_{i,j}\Delta t \tag{6}$$

Where $r = \mu\Delta t \big/ \Delta x \Delta y$

B. Accuracy and stability

Accuracy The iterative solution to GVF that given in Eqs.(5) and (6) is in fact an Euler's method for initial value problems. Therefore it is first-order accurate, i.e., the accumulated error is proportional to Δt. In order to guarantee convergence and accuracy, the time-step Δt must be kept small. Xu et al. suggested that the following condition must be maintained[1]:

$$\Delta t \le \frac{\Delta x \Delta y}{4\mu} \tag{7}$$

Stability The system of ODEs in Eqs.(3) and (4) which arise from the time-dependent partial differential equations are very large and often stiff[6]. For notational convenience, we rewrite the Eq.(3) in a vector function form (and similarly for Eq.(4)):

$$\frac{d\mathbf{x}_u}{dt} = \mathbf{A}\mathbf{x}_u \tag{8}$$

Where $\mathbf{x}_u = [u_1, u_2, \ldots, u_N, 1]^{\mathrm{T}}$. **A** is an $N \times (N+1)$ sparse matrix, and $N = N_1 \times N_2$. The *k*th (N+1)–dimensional row vector of matrix **A** is:

$$\mathbf{a}_k = \frac{1}{\Delta x \Delta y}\left[\cdots \quad \mu \quad \cdots \quad \mu \quad -(4\mu + b_k \Delta x \Delta y) \quad \mu \quad \cdots \quad \mu \quad \cdots \quad \Delta x \Delta y c_k^1\right]$$

where $b_k = b_{i,j}$ and $c_k^1 = c_{i,j}^1$.

The Jacobian matrix of the vector function in Eq.(8) is:

$$\mathbf{J} = \begin{bmatrix} \nabla \mathbf{a}_1 \mathbf{x}_u \\ \vdots \\ \nabla \mathbf{a}_N \mathbf{x}_u \end{bmatrix}$$

Where $\nabla \mathbf{a}_k \mathbf{x}_u = \left[\frac{\partial \mathbf{a}_k \mathbf{x}_u}{\partial u_1}, \ldots, \frac{\partial}{\partial u_N}\right]$.

The Gershgorin circle theorem identifies a region in the complex plane that contains all the eigenvalues of a square matrix. For the $N \times N$ Jacobian **A**, each eigenvalue is in at least one of the disks:

$$\left|\lambda - \left(-\frac{4\mu}{\Delta x \Delta y} - b_k\right)\right| \le \frac{4\mu}{\Delta x \Delta y} \tag{9}$$

Thus the real part of the eigenvalues range from $-\left(\frac{8\mu}{\Delta x \Delta y} + b_{\max}\right)$ to $-b_{\min}$, That is

$\mathrm{Re}(\lambda) \in \left[-\left(8\mu \big/ \Delta x \Delta y + b_{\max}\right), -b_{\min}\right]$.

Using the definition of b in the previous section, one can find that:

$$\mathrm{Re}(\lambda_i) < 0, \quad 1 \le i \le N$$
$$\max_{1 \le i \le N} \left|\mathrm{Re}(\lambda_i)\right| \; \square \; \min_{1 \le i \le N} \left|\mathrm{Re}(\lambda_i)\right|$$

Therefore, the linear system of ODEs in Eq.(8) is a stiff problem with large stiffness ratio. It is generally agreed that stability is more of a constraint than accuracy when numerically solving such a stiff problem. For Euler's method, the following condition must be satisfied to keep the errors bounded as the calculation advances:

$$\left|\lambda_i \Delta t + 1\right| < 1 \quad i = 1, 2, \ldots, N \tag{10}$$

Substituting the minimum value of $\mathrm{Re}(\lambda)$, i.e. $-\left(\frac{8\mu}{\Delta x\Delta y}+b_{\max}\right)$, into Eq.(10) and considering the accuracy restriction on the time step, we obtain:

$$\Delta t < \min\left\{\frac{2\Delta x\Delta y}{8\mu + b_{\max}\Delta x\Delta y}, \varepsilon_{\Delta}\right\} \tag{11}$$

where ε_{Δ} is a positive number smaller than one.

On the basis of the comparison between Eq.(7) and Eq.(11), one can conclude that Eq.(7) is an approximation to the solution of Eq.(11) by assuming $b_{\max} \approx 8\mu/\Delta x\Delta y$. We evaluate these two solutions at $\Delta x = \Delta y = 0.5$, $\mu = 0.3$. Taking $b_{\max} = 2$, we get $\Delta t < 0.208$ using Eq.(7), and $\Delta t < 0.172$ using Eq.(11). While taking $b_{\max} = 32$, the result remains $\Delta t < 0.208$ using Eq.(7), whereas the result is $\Delta t < 0.05$ by using our definition. Experimental results support that Eq.(11) gives a more accurate restriction on the time step for numerical stability.

It is worth making a few comments about the time step. In choosing a stepsize Δt for advancing the numerical solution of the systems of ODEs in Eqs.(3) and (4), we would like to take as large a step as possible to minimize computational cost. But we must also take into account both stability and accuracy. First, to yield a meaningful solution, the stepsize must obey the stability restrictions imposed by the numerical method, that is Eq.(11). Second, the chosen stepsize is needed to ensure that the desired accuracy is attained. With Euler's method which is first-order accurate, a much smaller stepsize than that calculated using Eq.(11) or Eq.(7) would be required.

III. GVF based on linear multi-step methods

The numerical solution to GVF that given in Eqs.(5) and (6) has the great virtue that it is simple to implement. However, a small stepsize would be required to obtain the desired accuracy.

A. Multi-step method for ODEs

Assume that the initial value problem to be solved is in the form:

$$\begin{cases} \dfrac{dy}{dt} = g(t, y) \\ y(t_0) = y_0 \end{cases}$$

Single-step methods, such as the methods of Euler, Heun, and Runge-Kutta, use only the information from one previous point to compute the successive point. That is, only y_n is needed to compute y_{n+1}. Multi-step methods use information at more than one previous point to estimate the solution at the next point. The linear multi-step methods have the form[7]:

$$y_{n+1} + \sum_{i=1}^{k} \alpha_i y_{n-i+1} = \Delta t \sum_{i=0}^{k} \beta_i g\left(t_{n-i+1}, y_{n-i+1}\right)$$

The parameters α_i and β_i are determined by polynomial interpolation. If $\beta_0 = 0$, the method is explicit; otherwise, it is implicit. It is generally agreed that the implicit methods, such as Gear's method[8], implicit Runge-Kutta method are more suitable for stiff problems.

We use Gear's method to solve the linear stiff system of ODEs,

$$\mathbf{x}_u^{(n)} + \sum_{i=1}^{k} \alpha_{k,i} \mathbf{x}_u^{(n-i)} = \Delta t \beta_k \mathbf{A} \mathbf{x}_u^{(n)}$$

By substituting entries of the matrix **A** into the right side of the equation, we get the iterative solution corresponding to the node (i, j) as:

$$u_{i,j}^{(n)} - s_{i,j}\left(u_{i+1,j}^{(n)} + u_{i,j+1}^{(n)} + u_{i-1,j}^{(n)} + u_{i,j-1}^{(n)}\right) = w_{i,j}^{1} \tag{12}$$

Where

$$s_{i,j} = \frac{\Delta t \beta_k \mu}{\left(1 + \Delta t \beta_k b_{i,j}\right)\Delta x \Delta y + 4\Delta t \beta_k \mu}$$

$$w_{i,j}^{1} = \frac{\Delta x \Delta y\left(\Delta t \beta_k c_{i,j}^{1} - \sum_{l=1}^{k} \alpha_{k,l} u_{i,j}^{(n-l)}\right)}{\left(1 + \Delta t \beta_k b_{i,j}\right)\Delta x \Delta y + 4\Delta t \beta_k \mu}$$

Similarly, we can obtain the iterative solution of $v_{i,j}$ corresponding to the node (i, j) as:

$$v_{i,j}^{(n)} - s_{i,j}\left(v_{i+1,j}^{(n)} + v_{i,j+1}^{(n)} + v_{i-1,j}^{(n)} + v_{i,j-1}^{(n)}\right) = w_{i,j}^{2} \tag{13}$$

Where

$$w_{i,j}^{2} = \frac{\Delta x \Delta y\left(\Delta t \beta_k c_{i,j}^{2} - \sum_{l=1}^{k} \alpha_{k,l} v_{i,j}^{(n-l)}\right)}{\left(1 + \Delta t \beta_k b_{i,j}\right)\Delta x \Delta y + 4\Delta t \beta_k \mu}$$

B. Iterative implementation

We rewrite the system of linear equations in (12) as:

$$\mathbf{S}\mathbf{x}_u = \mathbf{w}^1 \tag{14}$$

Where $\mathbf{w}^1 = \left[w_{1,1}^1, \ldots, w_{i,j}^1, \ldots, w_{N_1,N_2}^1\right]^{\mathrm{T}}$. **S** is an $N \times N$ sparse matrix with each row vector contains the coefficients in Eq.(12). That is, the *k*th row vector(corresponds to the node (i, j)) is:

$$\mathbf{s}_k = \left[\cdots \quad -s_{i,j} \quad \cdots \quad -s_{i,j} \quad 1 \quad -s_{i,j} \quad \cdots \quad -s_{i,j} \quad \cdots\right]$$

The diagonal entries of **S** is one. Each row of **S** has two nonzero entries(both take value $-s_{i,j}$) on the left and right of the diagonal entry, respectively. And there are zero entries between these left two and right two nonzero entries.

Similarly, we can rewrite the system of linear equations in (13) as

$$\mathbf{S}\mathbf{x}_v = \mathbf{w}^2 \tag{15}$$

Where $\mathbf{w}^2 = \left[w_{1,1}^2, \ldots, w_{i,j}^2, \ldots, w_{N_1,N_2}^2 \right]^{\mathrm{T}}$

In practice, iterative methods are usually more effective than sparse direct methods for solving the large sparse problem shown as Eqs.(14) and (15)[9]. Such schemes always save space and can often save computational work as well. Considering the matrix **S** is diagonally dominant but nonsymmetric, the BiConjugate Gradient Stabilized method(Bi-CGSTAB) is applicable.

We use three-step Gear's method. The parameters for calculating matrix **S** are $\alpha_{3,1} = -18/11$, $\alpha_{3,2} = 9/11$, $\alpha_{3,3} = -2/11$, $\beta_3 = 6/11$. An overview of the proposed algorithm for solving Eqs.(14) and (15) is given below:

1. Choose $\mathbf{u}^{(0)} = \nabla f$. Compute $\mathbf{u}^{(1)}$, $\mathbf{u}^{(2)}$ using fourth-order Runge-Kutta method. Let $k = 3$.
2. Let $\mathbf{z}^{(0)} = \mathbf{x}_u^{(k-1)}$ or $\mathbf{z}^{(0)} = \mathbf{x}_v^{(k-1)}$, $\mathbf{w} = \mathbf{w}^1$ or $\mathbf{w} = \mathbf{w}^2$. Solve the sparse linear problem $\mathbf{S}\mathbf{z} = \mathbf{w}$ using Bi-CGSTAB method(i.e., (a)--(b))
   (a) Compute $\mathbf{r}^{(0)} = \mathbf{w} - \mathbf{S}\mathbf{z}^{(0)}$, choose $\tilde{\mathbf{r}} = \mathbf{r}^{(0)}$, let ε be a small real number.
   (b) For *i* = 1, 2, …, do
   $\rho_{i-1} = \tilde{\mathbf{r}}^{\mathrm{T}} \mathbf{r}^{(i-1)}$
   if *i* = 1
   $\mathbf{p}^{(i)} = \mathbf{r}^{(i-1)}$
   else
   $\eta = \rho_{i-1}\gamma_{i-1} / \rho_{i-2}\omega_{i-1}$, $\mathbf{p}^{(i)} = \mathbf{r}^{(i-1)} + \eta\left(\mathbf{p}^{(i-1)} - \omega_{i-1}\mathbf{d}^{(i-1)}\right)$
   endif
   $\mathbf{d}^{(i)} = \mathbf{S}\mathbf{p}^{(i)}$, $\gamma_i = \rho_{i-1} / \tilde{\mathbf{r}}^{\mathrm{T}}\mathbf{d}^{(i)}$, $\mathbf{e} = \mathbf{r}^{(i-1)} - \gamma_i \mathbf{d}^{(i)}$
   if $\|\mathbf{e}\| \le \varepsilon$
   $\mathbf{z}^{(i)} = \mathbf{z}^{(i-1)} + \gamma_i \mathbf{p}^{(i)}$, goto 3
   endif
   $\mathbf{t} = \mathbf{S}\mathbf{e}$, $\omega_i = \mathbf{t}^{\mathrm{T}}\mathbf{e} / \mathbf{t}^{\mathrm{T}}\mathbf{t}$, $\mathbf{z}^{(i)} = \mathbf{z}^{(i-1)} + \gamma_i \mathbf{p}^{(i)} + \omega_i \mathbf{e}$, $\mathbf{r}^{(i)} = \mathbf{e} - \omega_i \mathbf{t}$
   if $\|\mathbf{r}^{(i)}\| / \|\mathbf{w}\| \le \varepsilon$
   goto 3
   endif
   next
3. $k = k + 1$, compute $\mathbf{w}^1$ and $\mathbf{w}^2$
   if $\|\mathbf{w} - \mathbf{S}\mathbf{z}\| \le \varepsilon$
   stop
   else
   goto 2
   endif

Through experiments, we find the nonstationary methods (e.g., the BiCGSTAB method, the Quasi-Minimal Residual method, etc.) are more effective than the stationary methods (e.g., the Successive Overrelaxation method) for solving the Eqs.(14) and (15). If other nonstationary method (e.g., the Quasi-Minimal Residual method) is chosen, the only thing to do is to replace the steps 2.(a) and 2.(b) with the selected algorithm.

IV. Experimental results

In this section, we report our experimental results with both synthetic and real medical images, and make comparison with the results using the method proposed by Xu et al[1, 2].

Fig. 1(a) shows a synthetic image corrupted by additive white Gaussian noise. We take the parameters as $\mu = 0.3$, $\Delta x = \Delta y = 0.5$. Fig. 1(c) shows the steady-state solution after 800 iterations by using Xu's method with the time step $\Delta t = 0.08$. Fig. 1(d) shows the result after 350 iterations by using our method with $\Delta t = 0.5$. Since we can not obtain the true solution of Eq.(2), it is reasonable and acceptable to record the iterations that converged to the steady-state solution. Fig.1(b) illustrates the iterations of the proposed method and the method proposed by Xu et al (using Eq.(5) and (6)).

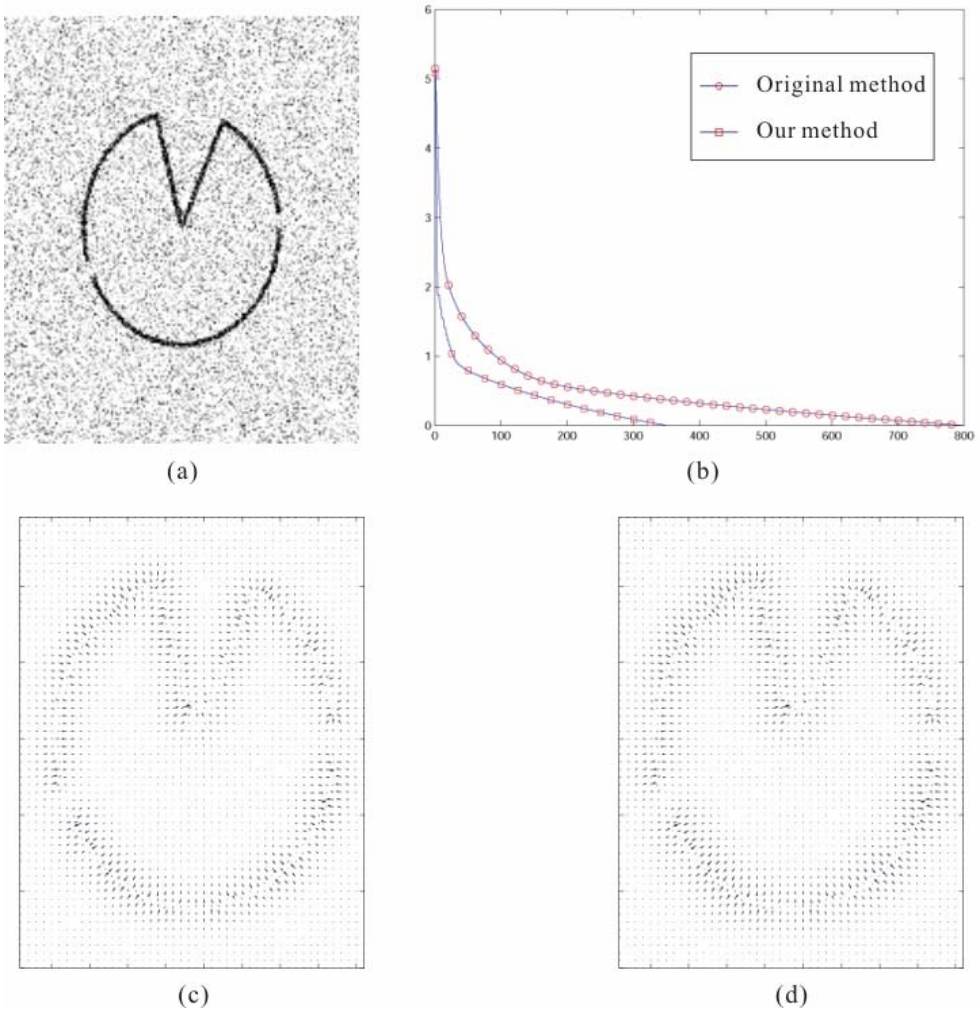


Fig.1 (a) Noisy 166×200 synthetic image, (b) Comparison of iterations, (c) GVF field by using Xu's method, (d) GVF field by using our method.

Then, the GVF field is expressed within the framework of a geometric active contour[10] that deforms an initial curve toward the object boundary. Fig.2 shows an example of shape modeling with GVF field serving as the external forces.

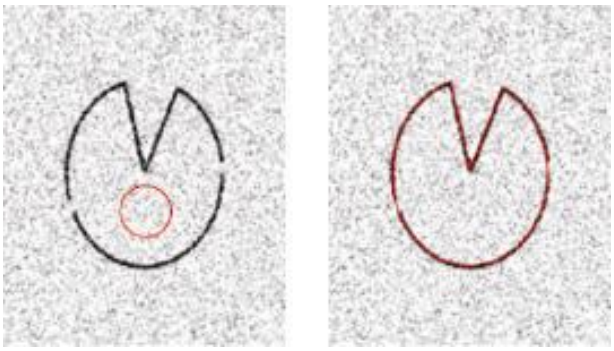


Fig.2 The initial curve(left), and the final result(right).

In Fig.3, the real medical image, the GVF fields computed using the proposed method, and the local detail corresponding to the region enclosed by the circle are shown from left to right. The parameters μ, Δx, Δy were set as same as these of the previous one.

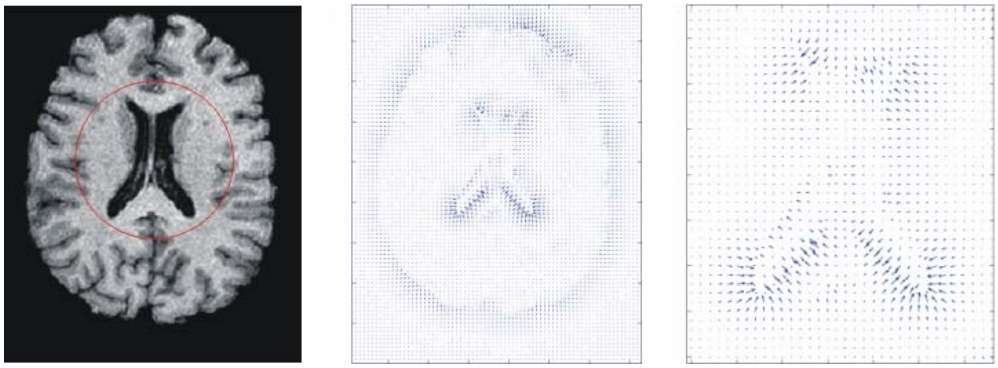


Fig.3 From left to right are: input MRI brain image, GVF field using the proposed method, and its local detail corresponding to the region enclosed by the circle.

V. Conclusion

We introduced a implicit method for computing gradient vector flow in this paper. Such a method has more large stability region and faster convergence than the explicit Euler schema proposed by Xu et al. Experimental results demonstrated that the proposed algorithm achieved near real-time performance. Utilizing the distribution of skin-tones in color space, it can be applied to the problems such as face tracking[11] and face feature tracking etc.

References

[1] C. Xu, J. L. Prince, “Snakes, shapes, and gradient vector flow”, IEEE Trans. Image Processing, vol. 7, pp.359-369, 1998.

[2] C. Xu, J. L. Prince, “Generalized gradient vector flow external forces for active contours”, Signal Processing, vol. 71, pp.131-139, 1998.

[3] N. Paragios, O. Mellina-Gottardo, V. Ramesh, “Gradient vector flow fast geometric active contours”, IEEE Trans. PAMI, vol.26, pp.402-407, 2004.

[4] F. Cen, F. Qi, “Tracking non-rigid objects in clutter background with geometric active contours”, Electronics Letters, vol. 38, pp.550-551, 2002.

[5] B. Gustaffson, H. -O. Kreiss and J. Oliger, TimeDependent Problems and Difference Methods, John Wiley & Sons, New York, 1995.

[6] E. Hairer, G. Wanner, Solving Ordinary Diferential Equations II: Stiff and Differential-Algebraic Problems, Springer-Verlag, New York, 1991.

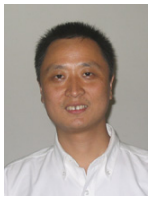
[7] M. T. Heath, Scientific Computing: An Introductory Survey, McGraw-Hill, 1997.

[8] C. W. Gear. Numerical Initial Value Problems in Ordinary Differential Equations. Prentice Hall, Englewood Cliffs, NJ, 1971.

[9] R. Barrett, M. Berry, et al. Templates for the solution of linear systems: Building blocks for iterative methods, Philadelphia: SIAM, 1994, 5-37.

[10] R. Malladi, J. Sethian, and B. C. Vemuri. “Shape Modeling with Front Propagation: A Level Set Approach”. IEEE Trans. PAMI, vol.17, pp.158-174, 1995.

[11] W. Chen, and F. Qi, “Tracking of the facial region in color video sequences using an improved narrow band algorithm”, Journal of Electronics & Information Technology, vol.27, pp.540-543, 2005.



Received his M.S. degree in mechanical design and theory from Zhejiang Institute of Silk , Hangzhou, in 1995, and Ph.D degree in computer science and engineering from Shanghai JiaoTong University, Shanghai, China, in 2004. Since 2004, he has been with the School of Computer and Information Engineering at Zhejiang Gongshang University, where he is currently an associate professor.