Semi-implicit Method of Fast Active Contour Models Using Iterative Method for Edge Detection

Norma Alias
Department of Mathematics
Faculty of Science
Universiti Teknologi Malaysia
MALAYSIA
norma@ibnusina.utm.my

Rosdiana Shahril
Department of Mathematics
Faculty of Science
Universiti Teknologi Malaysia
MALAYSIA
rosdianashahril@gmail.com

Abstract: In computational vision research, low-level tasks such as edge detection, stereo matching, and motion tracking have been widely emphasis as autonomous bottom-up processes. Active contours have also been widely applied for various applications in medical image processing. Semi-implicit AOS is very stable constraint on the size of the time step associated with explicit numerical schemes and will be adopted in our implementation. The proposed algorithm using iterative methods such as Jacobi, Gauss-Seidel and SOR will be implementing in object edge detection experiments performed on MRI images. MATLAB has been chosen as computational platform for the experiment implementation. It is well suited and widely used in the medical image processing for the monitoring and detection. The experimental results of the edge detection on medical images are illustrated in analysis section. As the conclusion the iterative method is the alternative scheme instead using direct method for accurate contours tracking of the medical image processing.

Key–Words: active contour, iterative methods, semi-implicit, AOS scheme, numerical method, image processing

1 Introduction

In computational vision research, process of autonomous bottom-up has been widely used for low-level tasks such as line or edge detection, motion tracking and stereo matching. Image processing has one problem where the edge detection has difficulty in finding lines separating homogeneous regions. Active contour model is one of the most-popular PDE(Partial Differential Equation) based tools in computer vision and powerful in object tracking.

Active contour model, also called classical explicit snaked is first introduce by Kass, Witkin and Terzopoulos [1]. They have been used in a variety of image and computer vision tasks such as object boundary detection and tracking. The snake ACM has two significant weaknesses [2]. Firstly, it depends on the intrinsic characteristics of the contour and parameterization and the model is a non-geometric model. Secondly, it cannot naturally handle topology of the evolving contour changes because of situations where no prior knowledge of the number of objects to be detected is available.

To solve these problems, in [3] has proposed a different model for active contours based on geometric partial different equation. It is independence of parameterization, intrinsic and stable. As important development has been the introduction of geodesic active contours [4][5]. This model is approach to object segmentation to connect snake model based on energy minimization and geometric active contours based on the theory of curve evolution. Level sets method were introduced by Osher and Sethian [5] for capturing moving fronts, in which the active contour is given implicitly as the zero level set of a scalar embedding function defined on the whole image domains; this allows for changes in the curves topology much more naturally than in parametric snakes.

In this paper, we will describe the proposed scheme by Weickert et al [6]. Edge detection based on semi-implicit addictive operator splitting (AOS) technique will be implementing for our edge detection on medical images such as medical resonance image (MRI). Direct method such as Thomas has been implemented in the proposed algorithm [6]. As the result, this method can be used as a consistent, unconditionally stable and computationally efficient. However, the disadvantages of this is reduced accuracy and number of iterations is very high. When the time-step, τ get very big, splitting artifacts emerge due to reduced rotational invariance can emerge. This keep number of iterations get very large for the contour to converge and constrains of the time-step.

In our proposed algorithm, iterative method such as Jacobi, Gauss-seidel and SOR will implementing

ISBN: 978-1-61804-051-0

for our edge detection. In this paper, we will describe the analysis of our result in terms of numerical performance.

2 Numerical Solution of the Model

2.1 Geodesic ACM using AOS scheme

The geodesic ACM using AOS scheme were proposed in [7] will be applied in this research to detect the edge of object. The equation of geodesic ACM [8] is expressed by

$$\frac{\partial u(x,y,t)}{\partial t} = |\bigtriangledown u| div(\frac{b(x)\bigtriangledown u}{|\bigtriangledown u|} + |\bigtriangledown u| kg(x)). \quad (1)$$

Then from (1) for $k = 0$, every step of the evolution of geodesic ACM corresponding to a linear nonhomogeneous diffusion is defined by:

$$\frac{\partial u(x,y,t)}{\partial t} = div(\frac{b(x)\bigtriangledown u}{|\bigtriangledown u|}). \quad (2)$$

The conduction coefficient not evolving function u and $b(x,y) = b(|\bigtriangledown I|)$ depends on the image I. For time derivative in (2), using a forward time difference:

$$\frac{\partial u}{\partial t} \approx \frac{u^{(n+1)} - u^n}{\tau}, \quad (3)$$

where τ is the size of the time-step.

Setting $C := \frac{b}{\bigtriangledown u}$. Descritize $div(c \bigtriangledown u)$ by the standard five-point stencil. In two dimensions, if for example the size of the square in the grid is h by h, the five point stencil of a point (x,y) in the grid is

$$\begin{aligned}
&(x-h,y),(x,y),(x+h,y),(x,y-h),(x,y+h)\\
div(c\bigtriangledown u) &\cong \partial_x(c_{(i,j)}\frac{u_{(i+1/2,j)}-u_{(i-1/2),j)}}{h_x}) + \partial_y(c_{(i,j)}\frac{u_{(i,j+1/2)}-u_{(i,j-1/2)}}{h_y}),\\
&\cong c_{(i+1/2,j)}((\frac{u_{(i+1/2,+1/2j)}-u_{(i-1/2+1/2,j)}}{h^2})\\
&-c_{(i-1/2,j)}((\frac{u_{(i+1/2-1/2,j)}-u_{(i-1/2-1/2,j)}}{h^2})\\
&+c_{(i,j+1/2)}((\frac{u_{(i,j+1/2+1/2)}-u_{(i,j-1/2+1/2))}}{h^2})\\
&-c_{(i,j-1/2)}((\frac{u_{(i,j+1/2-1/2)}-u_{(i,j-1/2-1/2))}}{h^2}),\\
&\cong c_{(i+1/2,j)}((\frac{u_{(i+1,j)}-u_{(i,j)}}{h^2}) - c_{(i-1/2,j)}((\frac{u_{(i,j)}-u_{(i-1,j)}}{h^2})\\
&+c_{(i,j+1/2)}((\frac{u_{(i,j+1)}-u_{(i,j)}}{h^2}) - c_{(i,j-1/2)}((\frac{u_{(i,j)}-u_{(i,j-1)}}{h^2}). \quad (4)
\end{aligned}$$

where h_x, h_y are the spatial finite difference discretization mesh grid lengths. In the following, it is assumed for convenience that $h_x = h_y = h$.

Setting $c_{(i+(1/2),j)} = \frac{c_{(i+1,j)}+c_{(i,j)}}{2}$, quotation in matrix-vector notation is written as

$$div(c\bigtriangledown u) \approx Au. \quad (5)$$

$A = [a_{(i,j)}]$ is the $N \times N (N = N_\times N_y$ is the total number of pixels of the $N_x \times N_y$ image) time-independent matrix with elements.

$\aleph(i)$ denotes the 4-neighborhood of pixel P_i. Hence, following its semi-implicit formulation as

$$a_{i,j} = \begin{cases} \frac{g_i+g_j}{2}, & j \in N(i) \\ -\Sigma_{k\Sigma N(i)}\frac{g_i+g_j}{2}, & j = i \\ 0, & \text{otherwise.} \end{cases} \quad (6)$$

$N(i)$ marked the 4-neighborhood of pixel P_i. Hence, following its semi-implicit formulation as

$$u_i^{n+1} = u_i^n + \tau(a_i|\bigtriangledown u|_i^n \Sigma_{j\in N(i)}\frac{(\frac{b}{|\bigtriangledown u|})_i^n + (\frac{b}{|\bigtriangledown u|})_j^n}{2}\frac{u_j^{n+1}-u_i^{n+1}}{h^2}). \quad (7)$$

$N(i)$ denotes the 4-neighborhood of pixel x_i. Here,the implementation is straight-forward finite difference which will give the cause of the problems when $|\bigtriangledown u|$ vanishes in a 4-neighborhood. If one uses a finite difference scheme with harmonic averaging then these problems will not appear. Thus, substituting $\frac{1}{2}((\frac{b}{|\bigtriangledown u|})_i^n + (\frac{b}{|\bigtriangledown u|})_j^n)$ in (7) by its harmonic counterpart:

$$u_i^{n+1} = u_i^n + \tau a_i|\bigtriangledown u|_i^n \Sigma_{j\in N(i)}\frac{2}{(\frac{|\bigtriangledown u|}{b})_i^n + (\frac{|\bigtriangledown u|}{b})_j^n}\frac{u_j^{n+1}-u_i^{n+1}}{h^2}). \quad (8)$$

In matrix-vector, notation is:

$$u^{n+1} = u^n + \tau\Sigma_{j\in\{x,y\}}A_l(u^n)u^{n+1}. \quad (9)$$

However, this scheme cannot determine the solution for u^{n+1} directly. Instead, it is needed in solving the systems of linear equations. The solution is given by:

$$u^{n+1} = (I - \tau\Sigma_{l\in\{x,y\}}A_l(u^n))^{-1}u^n, \quad (10)$$

where I is unit matrix. Reformulating (10) using AOS approximation yielded

$$u^{n+1} = \frac{1}{2}\Sigma_{l\in\{x,y\}}(I - 2\tau A_l(u^n))^{-1}u^n. \quad (11)$$

The operators $B_l(u^k) := I - 2\tau A_l(u^n)$ is to solve for the strict diagonally dominant tridiagonal linear systems efficiently. The constant force term $|\bigtriangledown u|kg$ had been neglected so far. This term stems from the hyperbolic dilation/erosion $\partial_t u = |\bigtriangledown u|$. Consequently, the gradient has to be approximated by an upwind scheme in a numerical implementation:

$$|\bigtriangledown u|_n^i = \begin{cases} |\bigtriangledown^- u|_n^i = max(D^{-x}u_i^n,0)^2 + max(D^{+x}u_i^n,0)^2 \\ +max(D^{-y}u_i^n,0)^2 + max(D^{+y}u_i^n,0)^{\frac{1}{2}}, \text{if } k \le 0 \\ |\bigtriangledown^+ u|_n^i = max(D^{-x}u_i^n,0)^2 + max(D^{+x}u_i^n,0)^2 \\ +max(D^{-y}u_i^n,0)^2 + max(D^{+y}u_i^n,0)^{\frac{1}{2}}, \text{if } k \le 0 \end{cases}$$

where D^{+x}, D^{+y}, D^{-x}, and D^{+y} denoted forward and backward for spatial derivatives approximation. The constant force term integrate into (11) and results for $k < 0$:

$$u^{n+1} = \frac{1}{2}\Sigma_{l\in\{x,y\}}(I - 2\tau A_l(u^n))^{-1}(u^n + \tau|\bigtriangledown^- u|^n kg). \tag{12}$$

The algorithm of the AOS step is summarized as follows [9]:

Algorithm 1 Pseudocode for AOS Step in m dimensions.

Input: $u = u^n$
(read image) Regularization: $v := K\sigma * u$
(stop function) Compute of diffusivity $g(|\bigtriangledown v|^2)$
(approximate $\bigtriangledown v$ using central differences)
(use look up table for evaluating g)
Create copy: $f := u$
Initialize sum : $u := 0$
(initial contour)
For $l = 1, \ldots, m$:
Compute $v := (mIm2\tau Al)^{-1} f$:
(solve $\frac{N}{N_l}$ tridiagonal systems of size N_l with Numerical method)

Update : $u := u + v$
Output: $u = u^{n+1}$ (final contour)

2.2 Numerical method

The semi-implicit scheme requires to solve a linear system, where the system is tridiagonal and diagonally dominant. Iterative method such as Jacobi, Gauss-seidel and SOR will be used to solve this tridiagonal linear system.

A system of algebraic equations has the form of

$$\begin{aligned}
A_{11}u_1 + A_{12}u_2 + \ldots + A_{1n}u_n &= f_1 \\
A_{21}u_1 + A_{22}u_2 + \ldots + A_{2n}u_n &= f_2 \\
A_{31}u_1 + A_{32}u_2 + \ldots + A_{3n}u_n &= f_3 \\
&\vdots \\
A_{n1}u_1 + A_{n2}u_2 + \ldots + A_{nn}u_n &= f_n
\end{aligned} \tag{13}$$

where the coefficients A_{ij} and the constants f_j are known, and u_i represents the unknowns. In matrix notation the equations are written as

$$\begin{pmatrix}
a & b & 0 & 0 & 0 & 0 & 0 \\
c & a & b & 0 & 0 & 0 & 0 \\
0 & \ddots & \ddots & \ddots & 0 & 0 & 0 \\
0 & 0 & \ddots & \ddots & \ddots & 0 & 0 \\
0 & 0 & 0 & \ddots & \ddots & \ddots & 0 \\
0 & 0 & 0 & 0 & c & a & b \\
0 & 0 & 0 & 0 & 0 & c & a
\end{pmatrix}
\begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ \vdots \\ \vdots \\ u_{n-1} \\ u_n \end{pmatrix}
=
\begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ \vdots \\ \vdots \\ f_{n-1} \\ f_n \end{pmatrix}$$

or, simply

$$Au = f \tag{14}$$

The following tridiagonal system of equations from (14) can be expressed as follows:

$$\begin{aligned}
\frac{2}{h^2}u_1^h - \frac{1}{h^2}u_2^h &= f_1^h, \\
-\frac{1}{h^2}u_{j-1}^h + \frac{2}{h^2}u_j^h - \frac{1}{h^2}u_{j+1}^h &= f_j^h \quad j = 2, 3, ..., n, \\
-\frac{1}{h^2}u_{n-1}^h + \frac{2}{h^2}u_n^h &= f_n^h.
\end{aligned} \tag{15}$$

The residual vector $b - Ax$ is a criteria to modify a component of approximate vector in order to improve an iteration. The algorithm is iterated until some specified convergence is achieved. Convergence is achieved when some measure of the relative or absolute change in the solution vector is less than a specified convergence criterion. The number of iterations required to achieve convergence depends on:

1. The dominance of the diagonal coefficients. As the diagonal dominance increases, the number of iterations required to satisfy the convergence criterion decreases.
2. The method of iteration used.
3. The initial solution vector.
4. The convergence criterion specified.

A large sparse linear system of equations is $Au = f$, where A can be decomposed into a diagonal D, a lower triangular component L, and a strictly upper triangular component U, then $A = D - L - U$ as illustrated in Figure 3.10. It is always assumed that the diagonal entries of A are all nonzero.

2.3 Jacobi

The Jacobi iteration determines the i-th component of the next approximation so as to eliminate the i-th component of the residual vector. In the following, $u_i^(k)$ denotes the i-th component of the iterate u_k and f_i the i-th component of the right-hand side f. Decomposition of $A = D - L - U$ will lead immediately to the

ISBN: 978-1-61804-051-0

vector form of the Jacobi iteration,

$$u_i^{k+1} = D^{-1}(L+U)u_k + (D)^{-1}b \tag{16}$$

This algorithm can be shown in discrete equation as

$$u_i^{k+1} = \frac{f_i - cu_{i-1}^{k+1} - bu_{i+1}^{k}}{a_{ii}} \tag{17}$$

The Jacobi method can be written as follows:

$$u_i^{k+1} = \frac{1}{a_{ii}}(f_i - \sum_{j=1}^{i-1} a_{ij}u_j - \sum_{j=i+1}^{n} a_{ij}u_j) \quad i = 1,2,...,n \tag{18}$$

This leads to the following program:

Algorithm 2 Pseudocode for Jacobi Iteration

For $k = 0, 1, \ldots,$ until convergence Do:
For $i = 1, 2, ..., p$ Do:
Solve $A_{ii}\delta_i = W_i^T(b - Ax_k)$
Set $x_{k+1} := x_k + V_i\delta_i$
EndDo
EndDo

2.4 Gauss Seidel

Gauss Seidel method is to solve linear systems of equations, $Au = f$. The Gauss-Seidel method requires diagonal dominance to ensure convergence and converges faster than the Jacobi iteration. The Gauss Seidel method can be written as follows:

$$u_i^{k+1} = \frac{1}{a_{ii}}(f_i - \sum_{j=1}^{i-1} a_{ij}u_j^{k+1} - \sum_{j=i+1}^{n} a_{ij}u_j^{k}) \quad i = 1,2,...,n \tag{19}$$

This algorithm can be shown in discrete equation as

$$u_i^{k+1} = \frac{f_i - cu_{i-1}^{k+1} - bu_{i+1}^{k}}{a_{ii}} \tag{20}$$

This equation will leads immediately to the vector form of the Gauss-Seidel iteration

$$u_i^{k+1} = (D-L)^{-1}Uu_k + (D-L)^{-1}b \tag{21}$$

This leads to the following program:

Algorithm 3 Pseudocode for Gauss-seidel Iteration

For $k = 0, 1, \ldots,$ until convergence Do:
For $i = 1, 2, ..., p$ Do:
Solve $A_{ii}\delta_i = W_i^T(b - Ax_k)$
Set $x := x + V_i\delta_i$
EndDo
EndDo

2.5 Successive-over-relaxation

Successive-over-relaxation is given by the following recursion:

$$(D - \omega E)u_{k+1} = (\omega F + (1-\omega)D)u_k + \omega b, \tag{22}$$

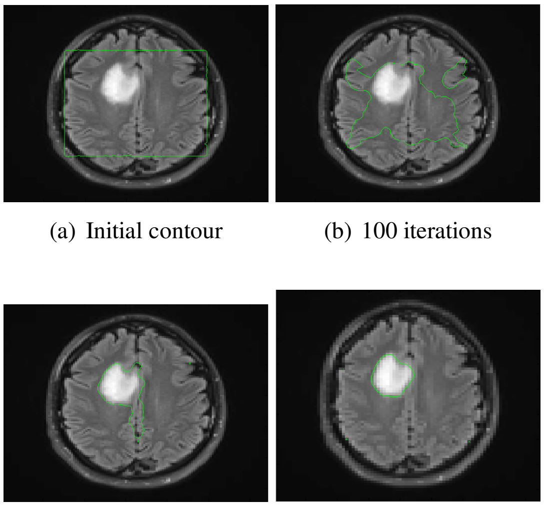
$$u_i^{k+1} = \frac{\omega}{a_{ii}}(f_i - \sum_{j=1}^{i-1} a_{ij}u_j^{k+1} - \sum_{j=i+1}^{n} a_{ij}u_j^{k}) + (1-\omega)u^k \quad i = 1,2,...,n \tag{23}$$

When $\omega = 1.0$, the SOR method is reduced to the Gauss-Seidel. The maximum rate of convergence is achieved for some optimum value of ω, denoted by $\omega_o pt$, which lies between 1.0 and 2.0.

3 Experimental Result and Discussion

The proposed algorithm using iterative methods such as Jacobi, Gauss-Seidel and SOR have been implemented in object edge detection experiments performed on MRI images. The MRI images used are brain tumor and breast tumor images. The proposed model was initialized randomly by closed curve. Figure 4.3-4.14 show the demonstration of the Geodesic ACM based AOS scheme using iterative method applied on 4 MRI images. In this case, time-step used was very small, τ is less than or equal to 5.

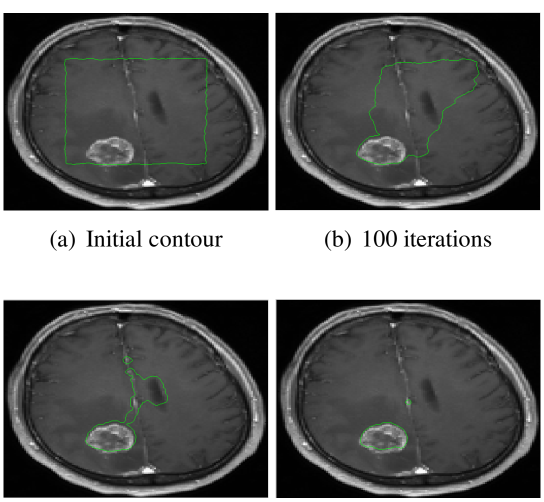
As shown on Figure 1-12, the edge boundaries of tumor regions are much visible on the final contour, where iteratively the tumor edge boundaries are located. From the demonstration, the speed and utility of iterative methods is proven for treating topological changes and finding boundaries for segmentation even in noisy environments.



(a) Initial contour (b) 100 iterations

(c) 250 iterations (d) Final contour(500 iterations)

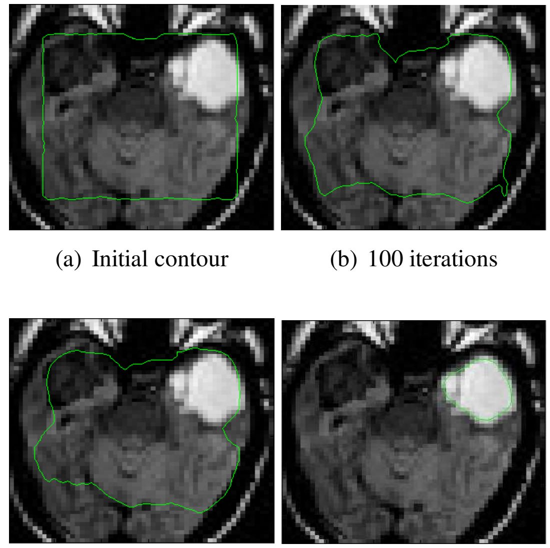
Figure 1: Final contour of the MRI image 1 based on Jacobi method



(a) Initial contour (b) 100 iterations

(c) 250 iterations (d) Final contour(500 iterations)

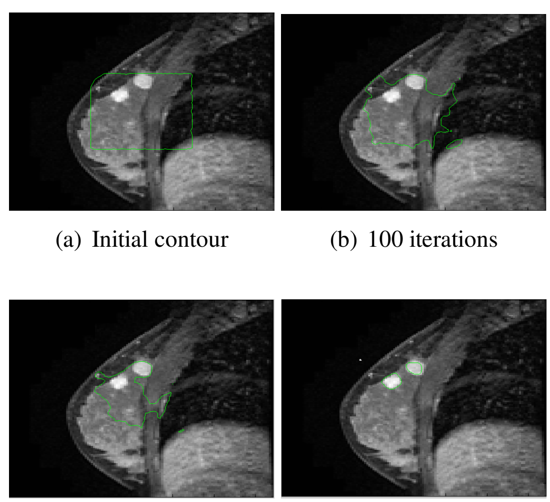
Figure 2: Final contour of the MRI image 2 based on Jacobi method



(a) Initial contour (b) 100 iterations

(c) 250 iterations (d) Final contour(2200 iterations)

Figure 3: Final contour of the MRI image 3 based on Jacobi method

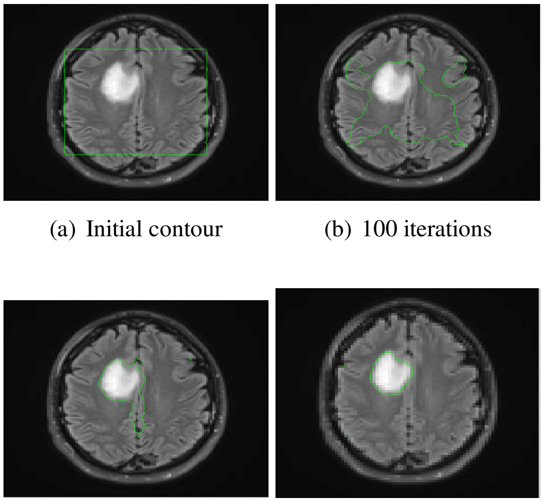


(a) Initial contour (b) 100 iterations

(c) 250 iterations (d) Final contour(700 iterations)

Figure 4: Final contour of the MRI image 4 based on Jacobi method

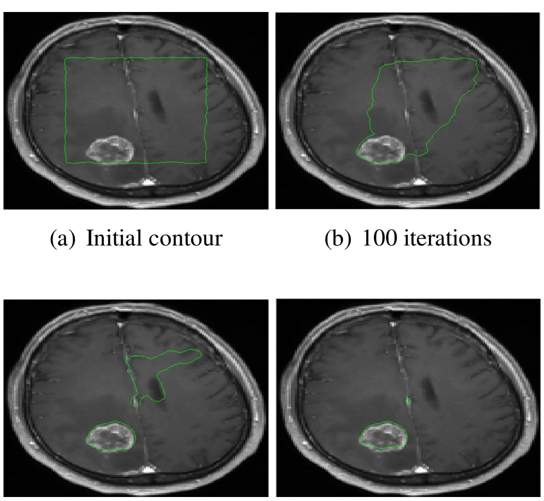
ISBN: 978-1-61804-051-0



(a) Initial contour (b) 100 iterations

(c) 250 iteration (d) Final contour((450 iterations))

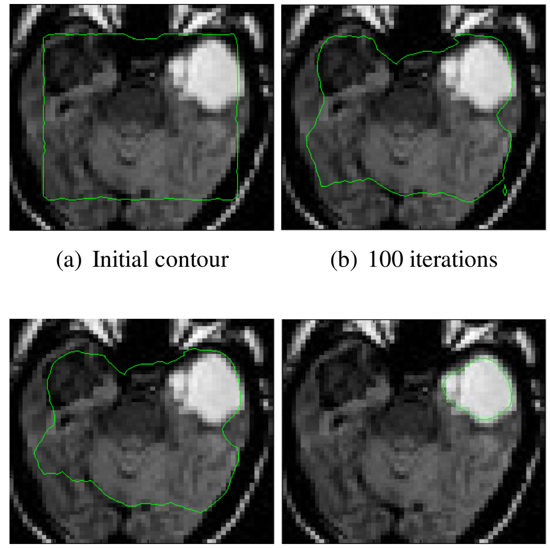
Figure 5: Final contour of the MRI image 1 based on GS method



(a) Initial contour (b) 100 iterations

(c) 250 iterations (d) Final contour(300 iterations)

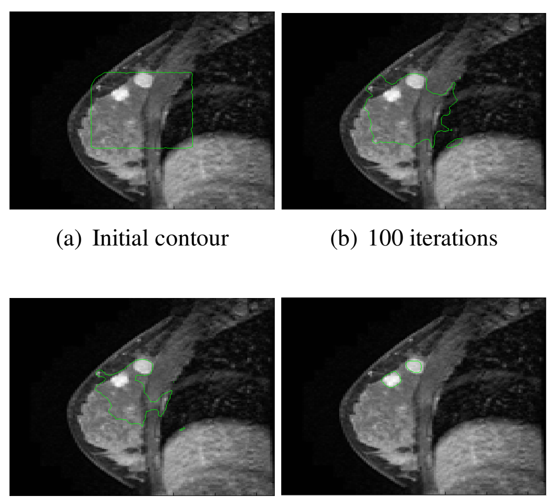
Figure 6: Final contour of the MRI image 2 based on GS method



(a) Initial contour (b) 100 iterations

(c) 250 iterations (d) Final contour(2000 iterations)

Figure 7: Final contour of the MRI image 3 based on GS method

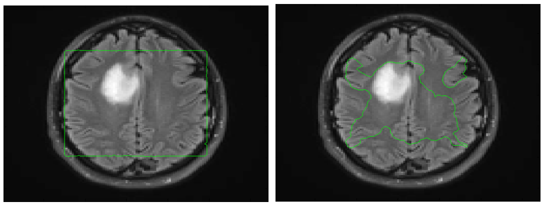


(a) Initial contour (b) 100 iterations

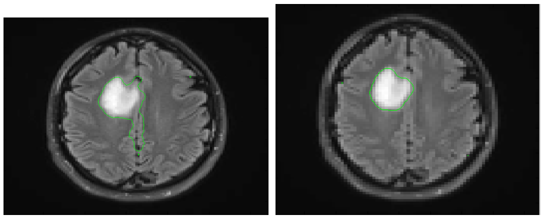
(c) 250 iterations (d) Final contour (600 iterations)

Figure 8: Final contour of the MRI image 4 based on GS method

ISBN: 978-1-61804-051-0

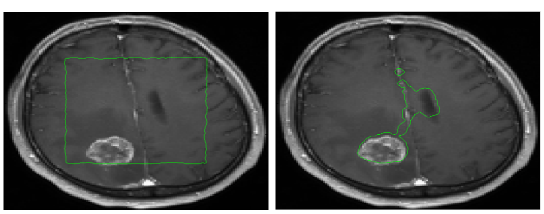


(a) Initial contour (b) 100 iterations

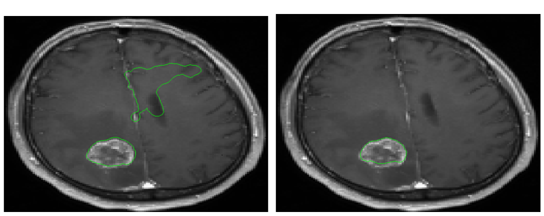


(c) 250 iterations (d) Final contour(430 iterations)

Figure 9: Final contour of the MRI image 1 based on SOR method

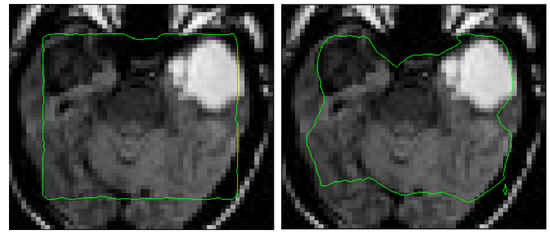


(a) Initial contour (b) 100 iterations

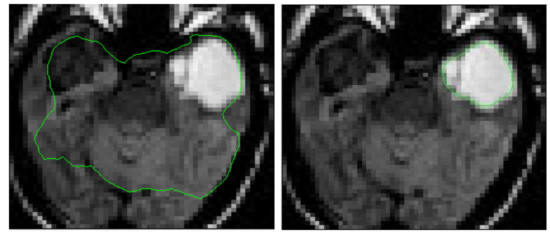


(c) 250 iterations (d) Final contour(250 iterations)

Figure 10: Final contour of the MRI image 2 based on SOR method

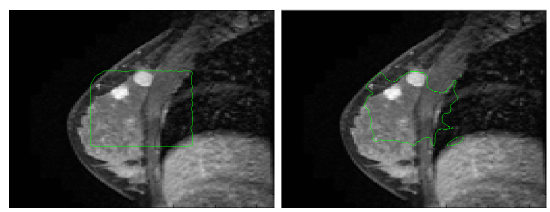


(a) Initial contour (b) 100 iterations

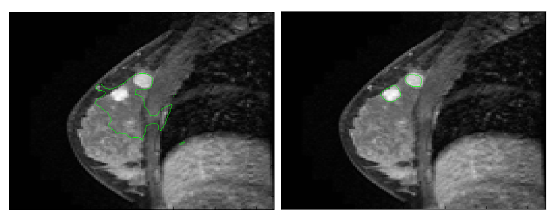


(c) 250 iterations (d) Final contour(1900 iterations)

Figure 11: Final contour of the MRI image 3 based on SOR method



(a) Initial contour (b) 100 iterations



(c) 250 iterations (d) Final contour (550 iterations)

Figure 12: Final contour of the MRI image 4 based on SOR method

The numerical analysis results and the parameters was used for sequential direct and iterative methods are shown in Table 1 and Table 2. From the table SOR shows the lowest iteration. To detect the edge for image 1, SOR needed 430 iterations to solve the 128x128 image pixels, while Gauss-Seidel needed 450 iterations and Jacobi with 500 iterations. Root mean squared error,*rmse* of the SOR method shows the lowest value than other methods, thus showing that SOR has the best convergence rate compared to Gauss-seidel and Jacobi method.

When compare between direct and iterative method, direct method is more accurate than iterative method. Direct method is much simpler and reduce the total number of operations, thus the truncation and round-off error that accumulates for large systems. While iterative method computes the error, r = f -Au and continue the same step to reduce the error until it converges. Although aims both of efficiency and accuracy, if accuracy takes as precedence will caused the accurate program is slightly slower while if is preferred to a faster one, the result will come out with unreliable accuracy. The unreliable accuracy arise from errors in the input data, computation errors due to finite precision arithmetic and the approximation error. This make the result is not accurate, while the direct method solve the problem directly to the exact solution.

Table 1: Numerical analysis of Gauss-elimination(GE) and Thomas Method(TH) for MRI image 1, image 2 and image 3

Parameter	Image 1		Image 2		Image 3	
	GE	TH	GE	TH	GE	TH
τ	5	5	5	5	5	5
k	-0.3	-0.3	-0.3	-0.3	-0.1	-0.1
Size of pixel	128×128	128×128	128×128	128×128	56×56	56×56
σ	-0.5	-0.5	-1.5	-1.5	-0.5	-0.5
Iteration	500,	500	600	600	2400	2400
$\Sigma\Delta$	0.8,0.1	0.3,0	0,3.3	0,2.9	1.8,3.3	1.5,2.7

4 Conclusion

In this section the proposed algorithm with various geodesic ACM based on numerical implementation is presented. The target application of the proposed ACMis tumor detection on MR images.

We have experimented with two algorithms, 1) Direct method such as Thomas algorithm and Gauss-elimination, 2) conventional iterative method such as Jacobi, Gauss-seidel and SOR, to solve our implementations based on Geodesic ACM using AOS scheme. Our future work concentrated how to improve the computational efficiency and accuracy with multigrid method.

References:

[1] M. Kass, A. Witkin, D. Terzopoulos, Snakes: Active contour models, Int. J. Comput. Vision, Vol. 1, 321331, 1988.

[2] Caselles, V., and Coll, B.: Snakes in movement. SIAM J. Numer. Anal., 33: 2445—2456 (1996).

[3] Caselles, V., Catt , F., Coll, T. and Dibois F.:A geometric model for active contours. Numerische Mathematik, 66: 1—31 (1993).

[4] S. Kichenassamy, A. Kumar, P. Olver, A. Tannenbaum, and A. Yezzi. A geometric snake model for segmentation of medical imagery. IEEE Med. Im., 16(2):199-209, 1997.

[5] S. Osher and J. A. Sethian. Fronts propagating with curvature dependent speed: Algorithms based on Hamilton-Jacobi formulations. J. C. Ph., 79:1249, 1988.

[6] J. Weickert and G. Kuhne. Fast methods for implicit active contour models. In S. Osher and N. Paragios, editors, Geometric Level Set Methods in Imaging, Vision and Graphics. Springer, 2002.

[7] J.Weickert, Anisotropic diffusion in image processing, Ph.D. thesis, Dept. of Mathematics, University of Kaiserslautern, P.O. Box 3049, 67653 Kaiserslautern, Germany, 1996.

[8] Caselles, V.,Kimmel, R. and Sapiro, G.: Geodesic active contours, Int. J. Comput. Vision. 22: 61—79 (1997).

[9] J.Weickert, B. ter Haar Romeny, and M. Viergever. Efficient and reliable schemes for nonlinear diffusion filtering. IEEE IP, 7(3):398410, Mar. 1998.

Table 2: Numerical analysis of Jacobi(JC), Gauss-seidel(GS) and SOR method for MRI image 1, image 2 and image 3

Parameter	Image 1			Image 2			Image 3		
	JC	GS	SOR	JC	GS	SOR	JC	GS	SOR
τ	5	5	5	5	5	5	4	5	5
k	-0.3	-0.3	-0.3	-0.1	-0.1	-0.1	-0.1	-0.1	-0.1
σ	0.5	0.5	0.5	1.5	1.5	1.5	0.5	0.5	0.5
Size of pixel	128×128	128×128	128×128	128×128	128×128	128×128	56×56	56×56	56×56
Iteration	500	450	430	500	300	250	2200	2000	1900
$\Sigma\Delta$	1.1,1.0	1.1,0.4	1.1,0.4	0.2,3.4	0.2,3.1	0.1,3.1	1.8,3.7	1.8,3.7	1.8,3.6
rmse	1.4493e-004	4.28226e-010	1.06581e-014	8.62281e-003	1.42624e-004	1.34345e-006	2.12180e-003	6.98604e-004	5.62541e-005
rmspe	6.59924e-004	1.94091e-009	4.83054e-014	5.70397e-002	8.87065e-004	8.34868e-006	4.64915e-002	1.51557e-002	1.21711e-003
max err	2.61442e-002	2.73352e-004	3.15326e-006	1.9881e-003	8.82359e-004	2.43006e-006	4.52338e-002	4.65890e-002	4.23943e-002
ε	0.5e-3	0.5e-3	0.5e-3	0.5e-3	0.5e-3	0.5e-3	0.5e-3	0.5e-3	0.5e-3