International Journal of Engineering & Technology, 7 (1.1) (2018) 101-105

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET

Research paper

GVF snake algorithm-a parallel approach

Priya P Sajan [1*], S.S. Kumar [2]

[1] *Research Scholar, Department of Computer Applications, Noorul Islam University, India*
[2] *Associate Professor, Electronics and Instrumentation, Noorul Islam University, India*
**Corresponding author E-mail: priyasajans@gmail.com*

Abstract

Multicore architecture is an emerging computer technology where multiple processing element will be acting as independent processing cores by sharing a common memory. Digital image segmentation is a widely used medical imaging application to extract regions of interest. GVF Active Contour is a region based segmentation technique which extracts curved and irregular shaped regions by diffusing gradient vectors and by the influence of internal and external forces. This requires prior knowledge on the geometric position and anatomical structures to locate the specific region defined within an image domain. This process requires complex mathematical calculations which in turn results in the immense consumption of CPU processing time. This may adversely affect the overall performance efficiency of the process. With the advancements in multicore technology, this processing time delay can be reduced by adapting parallelization in the computation of GVF field to the specific region of interest which is to be segmented. OpenMP is a shared memory parallel programming construct, which could implement multicore parallelism with extensive and powerful APIs thereby supporting the functionalities required to attain parallelism. This article provides a high level overview of OpenMP, its effectiveness and ease of implementation in adapting parallelism to existing traditional sequential methods using instruction, data and loop level parallelism. Performance comparison could be done with sequential versions of the program written in Matlab, Java and C languages with the proposed parallelized version of OpenMP. The result is also comparable with different operating systems like Windows and Linux.

Keywords: *GVF, active contour snake, region based segmentation, OpenMP, API.*

1. Introduction

Digital image segmentation is the process of extracting desired and meaningful information from irregular shaped objects. To recognise this region, distinct features like shape, location, size of the object etc should be exploited. Snakes or active contours are curved surfaces that account a prior knowledge on the geometric position and anatomical structures to locate a region within a defined image domain. These curve movements are influenced by both internal and external forces which emerge within the curve itself or by the computed image data. The internal and external force definition will make the snake conform to a specific object boundary and within the image it will constitute some specific desired features. Snakes have profound applications in various innovative areas of science and technology.

Active contour models can be generally classified into two types– Parametric Active Contours and Geometric Active Contours. Parametric active contours are based on some parameters which synthesize the image domain to be configured towards desired boundaries. This is termed to be the potential force which demarks the curves around a specified edge. Irrespective of this potential force, there are additional forces which make the curve to hold together with some elasticity or to keep away from the bending area with some internal pressure. Object boundary could be located and calculated by initializing the parametric curves within the object domain under the influence of both internal and external forces. Mathematically, a deformable models is a curve

X(s) = (X(s), Y(s)), s ∈[0,1]

which moves through the spatial domain of an image to minimize the following energy functional.

$\varepsilon(X) = S(X) + P(X)^2$

The first term is the internal energy functional and is defined to

be $S(x) = \frac{1}{2}\int_0^1 \alpha(s)\left|\frac{\partial x}{\partial s}\right|^2 + \beta(s)\left|\frac{\partial^2 x}{\partial y^2}\right|^2 \quad ds$

The first order derivative discourages stretching and makes the model behave like an elastic string. The second order derivative discourages bending and makes the model behaves like a rigid rod. The weighting parameters *α(s)*and *β(s)* can be used to control the strength of the model's tension and rigidity respectively. Practically, values of *α(s)*and *β(s)* are often chosen to be a constant value 1.

The second term is the potential energy functional and is computed by integrating a potential energy function *P(x,y)* along the contour

$X(s): P(x) = \int_0^1 P(X(s)) \quad ds$

The potential energy function *P(x,y)* is derived from the image data and takes smaller values at object boundaries as well as other features of interest. Given a gray level image *I(x,y)* viewed as a function of continuous position variables *(x,y)*, a typical potential energy function is designed to lead a deformable contour toward step edges. Inorder to handle the problems of less capture of region of interest and convergency overlapping in boundary concavities, *Gradient Vector Flow* method is proposed.

2. GVF snake algorithm

Gradient vector field is obtained from the input image by computing the pixel values in contour map *f(x,y).* The contour map is a potential function which is defined over the image plane whose value will be slightly larger nearby the edges of interested area which is proposed to be segmented. The internal and external forces are derived from energy functions which gradually converges towards the edges. The internal energy function is defined as

$$E_{internal=\int_s \left(\frac{1}{2}\right)(\alpha(s)|x_s|^2+\beta|x_{ss}|^2)\,ds} \quad (1)$$

The first integral term represents the external elastic energy which makes the curve to shrink forcibly. The second integral term determines the bending energy which smoothens the curve. $\alpha(s)$and$\beta(s)$ are weighting functions whose values are constants. The external image energy along the curve could be defined as

$$E_{external} = \int_s E_{image}\,(x(s))\,ds \quad (2)$$

The image energy is defined as either of the following two terms

$$E_{image}\,(x,y) = -|\nabla I(x,y)|^2 \quad (3)$$
$$E_{image}(x,y) = -|\nabla[G_\sigma(x,y) * I(x,y)]|^2 \quad (4)$$

The total energy is the sum of internal bending energy and external elastic energy which can be defined as

$$E_{snake} = E_{internal} + E_{external}$$
$$= \int_s[\left(\frac{1}{2}\right)(\alpha(s)|x_s|^2 + \beta|x_{ss}|^2) + E_{image} \quad (5)$$

The Gradient Vector field is defined as *v(x,y) = [u(x,y), v(x,y)]* that minimizes the energy functional

$$\varepsilon = \iint \mu(u_x^2 + u_y^2 + v_x^2 + v_y^2) + |\nabla f|^2 v - \nabla f^2 dxdy \quad (6)$$

where *f(x,y)* is the contour map of the image; ∇ is the gradient operator; μ is a positive parameter of regularisation; *u(x,y)* and *v(x,y)* are the functions which denote the components of GVF field and u_x, u_y and v_x, $v_y$represent the partial derivatives of *u(x,y)* and *v(x,y)* in correspondence to the respective values of *x* and *y*. The energy functional keeps $w \approx \nabla f$ when ∇f is larger. Otherwise, it will be producing a field which is only slightly varying in homogenous regions.

Hence this is a variational formulation which is following a standard systematic principle that produces a smooth result for the homogenous edges when there is a gradient contour map field. It means that as the value of ∇f decreases, the energy functional will be dominated by the sum of the squares of the partial derivatives *u(x,y)*and*v(x,y)* in the ctor field generating a slowly varying field. Instead if ∇f increases, the difference term dominates the integral value by regularising the value $v = \nabla f$. μwill be acting as the regularisation parameter which controls the compromisations between the x and y in the integral. The value of this parameter will be normally set in accordance with the amount of noise present in μ. As noise level increases, value of μwil also increase.

The first term of the integral function, which is the smoothing term, is the correspondance value that depends upon the curliness and concavity of the vector field. This vector field will inturn results in pixel allocation status over the convergent edges. By using variational calculus, GVF can be computed by solving the values of *u(x,y)*and *v(x,y).*

$$\mu\nabla^2 v - (v - f_y)(f_x^2 + f_y^2) = 0 \quad (7)$$
$$\mu\nabla^2 u - (u - f_x)(f_x^2 + f_y^2) = 0 \quad (8)$$

where ∇^2 is the Laplacian operator.

Solutions of these equations will generate computational field *w*, which provide advance instinctive knowing about the Gradient Vector field. If the homogenous region that is I_M *(x,y)* becomes readily constant, the value of second term in each function becomes zero since *f(x,y)* becomes zero. In such homogenous regions, Lapalce equation will determine the values of *u* and *v*which results in the competition among pixels in the edge boundary, inside, outside or within the region of interest. This explains the reason behind how the vector points converge towards the boundary.

The component values of *u(x,y)*and*v(x,y)* in the GVF computational field can be obtained by considering *u* and *v* interms of function of time as follows

$$u_t(x,y,t) = \mu\nabla^2 u(x,y,t) - [u(x,y,t) - f_x(x,y)].[f_x(x,y)^2 + f_y(x,y)^2] \quad (9)$$
$$v_t(x,y,t) = \mu\nabla^2 v(x,y,t) - [v(x,y,t) - f_y(x,y)].[f_x(x,y)^2 + f_y(x,y)^2] \quad (10)$$

These Laplacian linear parabolic equations uses centered finite difference equations. Once decouple these equations, they can be used for solving the partial differential equations which gives the values of of *u* and *v.*

$$\nabla^2 u = u_{i-1,j} + u_{i+1,j} + u_{i,j-1} + u_{i,j+1} - 4u_{i,j} \quad (11)$$
$$\nabla^2 v = v_{i-1,j} + v_{i+1,j} + v_{i,j-1} + v_{i,j+1} - 4v_{i,j} \quad (12)$$

The range of the produced image field that is captured will be directly proportional to the total number of iterations which inturn determines the GVF field. This mode of controlling and regularising the image capture range provides the GVF method to evacuate the location dependancy for the initial active contour. If a certain shape of interest is not included or is partially included in the initial snake, the GVF field will allow deformations of the initial active contour to include the shape. If the region of interst is located completely out of the initial active contour, it is even possible that the complete snake move towards the shape to capture it, provided there are no same additional shapes in the neighbourhood.

GVF active contour snake involves huge and complex arithmeticcalculations which drastically increases the processing time. However the computers arefaster, technical implications have to be improved to utilize this immense processing power of the internal hardware architecture. Multicore processor architecture means computers with multiple independent functional units that may be able to process simultaneously. To accomplish this, compiler directives are deployed to determine the inter-dependencies between processes and their processing order which may further utilize instruction,data and loop level parallelism in the concerned area there by exploiting threaded parallel shared memory architecture.

With technological advancements, it is made possible that the rate at which instructions are processed in extremespeed and accuracy. Since shared multicore technology is going mainstream, it is crucial that application programs must take account the effective use of parallelism that is available in currentprocessor's architecture.

OpenMP

OpenMP enables the creation of shared memory parallel programs. It describes how the work is to be shared among threads that will execute on different processing cores and the order to access the shared data as and when needed. OpenMP's directives tell the compiler which instructions to execute in parallel and how to distribute them among the threads that will run the code. The code portions has to be re-organized to obtain independent instruction sequences. Practical benefit of OpenMP is that it can be applied to incrementally create a parallel program from an existing sequential one. In writing multithreaded parallel programs, the most important task is to understand which data is

shared or private, not only to performance, but also for program correctness. OpenMP makes this distinction apparent to the programmer through a set of classes such as shared, private and default which could be set according to the nature of the program. With OpenMP, it is the developer's responsibility to indicate to the compiler which pieces of memory should be shared among the threads and which should be kept private. When memory is identified as shared, all threads access the exact same memory location. When memory is identified as private, a separate copy of variable is made for each thread to access in private.

To achieve optimal performance, a multithreaded OpenMP program must have effective loop scheduling and partitioning. Ultimate aim is the better utilization of processing cores with minimum overhead of scheduling, context switching and synchronization. By default, loops in OpenMP parallel program uses static-even scheduling, which means the iterations of the loop are distributed among threads in an approximate equal number of iterations.

Parallel GVF snake implementation

The biggest time consuming part of GVF Snake is curve deformation.The first step is to compute the gray level scale luminance of the actual image. Inorder to compute this gray level scale luminance, a weighted sum must be calculated in a linear RGB space thereafter by removing Gama compression function through Gama expansion.

$$C_{linear}=\begin{cases}\frac{C_{srgb}}{12.92}, & C_{srgb}\leq 0.04045\\ \left(\frac{C_{srgb}+0.055}{1.055}\right)^{2.4}, & C_{srgb}>0.04045\end{cases}$$

Image gradient is calculated for each and every pixel co-ordinates in the image and these pixels will be identified based upon the intensity value. The starting seed point is calculated using binary gradient value. For this a pixel map is to be calculated which is then to be converted to binary gradient value which computes and displays the pixel values in the proposed segmented area. This sequential procedure involves complex mathematical computation which consumes enormous computation time. This could be solved by converting this sequential procedure to a parallelised version using OpenMP construct.

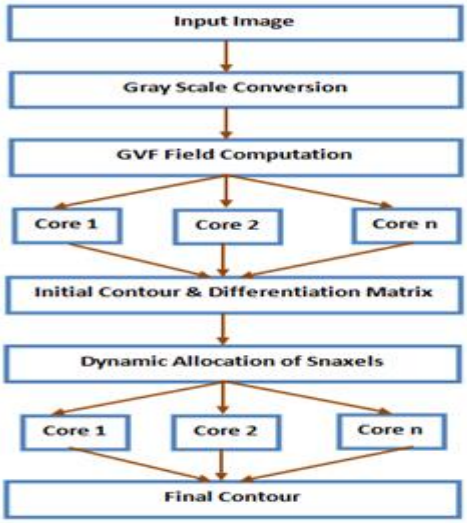

Fig. 1(a): Proposed Parallel Flow Chart

3. Experimental results

Segmentation as well as performance analysis is done with medical images and were conducted in Intel Core i7-3770 CPU at 3.40GHz and 8GB RAM. The program was able to run in both

a. Distribute computation work among processing threads for parallel execution
b. Specify the mode of work sharing among threads
c. Processing of threads must be mutually exclusive so that co-processing threads could execute without much interference or barriers

OpenMP region consists of all codes encountered during a specific instance of the execution of a given OpenMP construct or library routine which encompasses all the code that is in the dynamic extent of construct. Some of the runtime library routines have an effect on the thread that invokes them or return information pertinent to that thread only, whereas others are relevant to a team of threads or to all threads that execute the program. Parallel constructs plays a crucial role in OpenMP. Parts of the program that are not enclosed by parallel construct will be executed serially. When a thread encounters this construct, a team of threads is created to execute the associated parallel region, which dynamically contains the parallel construct. At the end of the parallel region, there is an implied barrier that forces all threads to wait until the work inside the region has been completed. Only the initial thread continues execution after the end of the parallel region. Threads that encounter the parallel construct become master of the new team. Each thread in the team is assigned a unique thread number which may range from zero to one less than the number of threads within the team. But the case is that each thread is allowed to follow a different path of execution.

To calculate pixmap of similar gradient level, data level and loop level parallelism can be applied with *omp critical* construct. This construct ensures mutual exclusion so that no two threads cannot access or update the same shared memory simultaneously. So the thread which enters the critical region checks and waits until no other co-processing thread is trying to access the same region with same name. For the parallel implementation of GVF Snake method, OpenMP adopts data, instruction and loop level parallelism which could simultaneously process hundreds of pixels so that processing speed and efficiency is catalysed.

RGB value of input image is converted into gray scale for computing the gradient value. Threshold value for each pixel position will be calculated to identify its gradient value. Using this result, the region of interest which is to be segmented will be encountered. This sequential process could be parallelised with the help of appropriate omp work sharing parallel constructs. It specifies the region of code where and how the work is distributed among executing threads.

```
#pragma omp parallel shared(n)
{   for(i=0;i<=n;i++)
    #pragma omp master
      i) compute length of contour region
      ii) compute the rgb values to gray scale

    #pragma omp for private
      i) calculate each point of contour region

    #pragma omp for nowait()
      i) boundary check with pixel allocation for GVF field computation

    #pragma omp for atomic nowait()
      i) initial contour identification
      ii) dynamic allocation of snaxels

    #pragma omp for ordered()
      i) final contour extraction
}
```

Fig. 1(b): Proposed Parallel Pseudocode

Windows and Linux platforms. The sequential program is written in Matlab, Java and C language. Further the parallelization of code was made by implementing OpenMP directives in the C version of the program. Again this parallelized code is made executed in Windows platforms to make a comparative study on processing

speed and efficiency in the kernel level, if any with OpenMP constructs in level, work sharing, synchronization and data property modes. The experiment showed almost similar execution time for the C version of the OpenMP when it made executed in Windows and Linux and has an immense processing speed and efficiency when compared with the traditional sequential program.

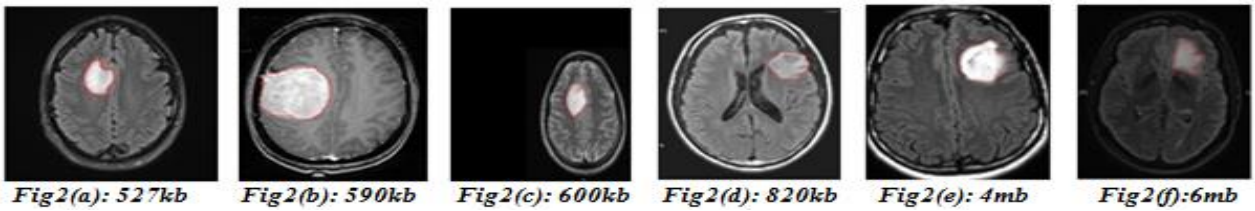

Fig2(a): 527kb Fig2(b): 590kb Fig2(c): 600kb Fig2(d): 820kb Fig2(e): 4mb Fig2(f):6mb

a. Sequential Vs parallel execution time comparison

The striking effect of the proposed parallel GVF Active Contour Snake method were analysed by comparing the execution time consumed for sequential and parallel mode by implementing in Matlab, Java, C and C with OpenMP. While implementing with OpenMP, more time oriented results were studied with multiple flavours of level, work sharing, synchronization and data property constructs in Windows octa core system.

Table 1: Sequential and Parallel Execution Time Analysis for Matlab, Java, C and OpenMP

Image Size	Sequential (in secs)			Parallel (in secs)
	Matlab	*Java*	*C*	*OpenMP*
527 KB	4.38	4.56	4.80	1.11
590 KB	4.65	4.79	4.90	1.19
600 KB	5.13	5.02	5.34	1.31
820 KB	5.93	6.02	6.08	1.85
4 MB	10.56	12.19	13.09	4.92
6 MB	25.32	21.39	24.08	10.80

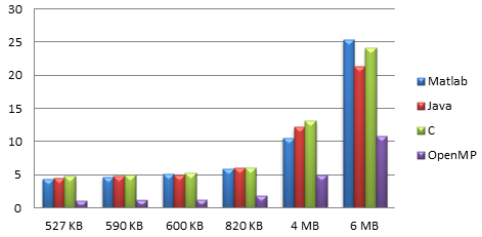

Fig. 3: Sequential and parallel execution time analysis

b. Performance analysis for OpenMP constructs

Outcome of examinations with the proposed parallel version of Gradient Vector Flow Active Contour Snake method in level, work sharing, synchronization and data property modes contemplates as it being most proficient candidate for implementing parallelism in multicore architecture to perform segmentation on medical images.

b.1. Level Constructs

Table 2: Level Constructs based Parallelization with OpenMP

Image Size	Level based parallelizing with OpenMP (in sec)			
	Data	*Iterations*	*Instruction*	*Task*
527 KB	1.05	1.22	1.38	1.25
590 KB	1.08	1.58	1.63	1.57
600 KB	1.20	1.37	1.49	1.63
820 KB	1.31	1.89	2.36	2.57
4 MB	4.71	6.65	5.98	6.46
6 MB	10.39	10.78	11.24	11.05

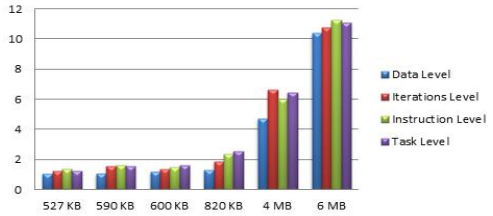

Fig. 4: Time comparison using level constructs

b.2. Work-sharing constructs

Table 3: Work Sharing Constructs based Parallelization with OpenMP

Image Size	Work-sharing based parallelizing with OpenMP(in sec)		
	Loop	*Sections*	*Single*
527 KB	1.83	1.43	1.12
590 KB	1.97	1.97	1.19
600 KB	2.51	2.39	1.56
820 KB	2.93	3.93	1.80
4 MB	6.91	6.74	5.58
6 MB	10.82	10.61	9.77

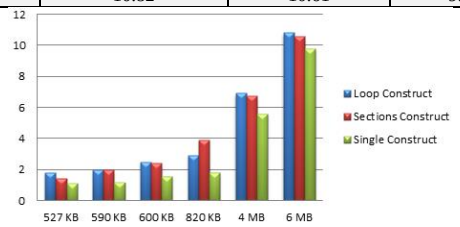

Fig. 5 : Time comparison using work-sharing constructs

b.3. Synchronization constructs

Table 4: Synchronization Constructs based Parallelization with OpenMP

Image Size	Synchronization based parallelizing with OpenMP (in sec)			
	Barrier	*Ordered*	*Critical*	*Atomic*
527 KB	1.43	1.29	1.04	1.04
590 KB	1.58	1.45	1.14	1.18
600 KB	2.13	2.13	1.47	1.54
820 KB	3.33	3.40	1.68	1.94
4 MB	6.13	6.04	5.50	5.48
6 MB	10.16	10.18	9.21	9.07

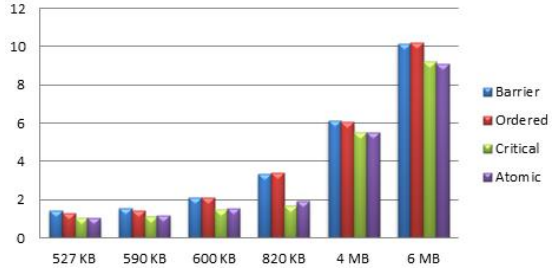

Fig. 6: Time comparison using Synchronization constructs

b.4. Data property constructs

Table 5: Data Property Constructs based Parallelization with OpenMP

Image Size	Data Property based parallelizing with OpenMP (in sec)			
	Private	***Shared***	***Default Shared***	***Nowait***
527 KB	1.76	1.60	1.48	1.29
590 KB	1.97	1.79	1.53	1.27
600 KB	2.41	2.31	1.79	1.55
820 KB	2.93	2.76	1.84	1.45
4 MB	6.51	6.24	5.83	5.11
6 MB	10.35	10.19	9.38	9.06

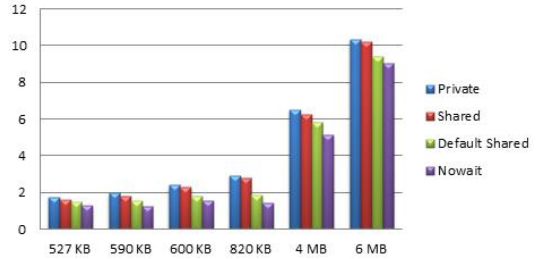

Fig. 7: Time comparison using data property constructs

4. Conclusion and future scope

Image segmentation with GVF snake method will gives accurate and precise result. But it consumes enormous amount of time to get processed. This processing delay could be reduced by implementing shared memory parallel approach using OpenMP. This experimental approach with OpenMP will provide an insight into the deeper understanding of the inter process communication between kernel threads and process threads. OpenMP is still an active research area. With its particular level of simplicity and its high level support for creating threads and exploiting shared memory multicore architecture, OpenMP will be having a foreseeable future in diverse areas of science and technology.

References

[1] Alvarado R, Tapia JJ & Rolón JC, "Medical image segmentation with deformable models on graphics processing units", *The Journal of Supercomputing*, Vol.68, No.1, pp.339-364, (2014).

[2] Phillips RD, Watson LT & Wynne RH, "Hybrid image classification and parameter selection using a shared memory parallel algorithm", *Computers & Geosciences*, Vol.33, No.7, pp.875-897, (2007).

[3] Mahmoud MKA & Al-Jumaily A, "Segmentation of skin cancer images based on GVF snake", *IEEE International Conference on mechatronics and automation*, pp. 216-220, (2011).

[4] Schellmann M, Gorlatch S, Meilander D, Kosters T, Schafers K, Wubbeling F & Burger M, "Parallel medical image reconstruction from graphics processing units to grids", *Journal of Supercomputing*, Vol.57, pp.151-160, (2011).

[5] Pallippuram VK, Bhuiyan M & Smith M C, "A comparative study on GPU programming models and architectures using neural networks", *Journal of Supercomputing*, Vol.61, pp.673-718, (2012).

[6] Zheng ZY & Zhang RX, "A fast GVF snake algorithm on the GPU", *Research Journal of Applied Sciences, Engineering and Technology*, Vol.4, No.24, pp.5565-5571, (2012)

[7] Lee S & Eigenmann R, "OpenMPC: Extended OpenMP for efficient programming and tuning on GPUs", *International Journal of Computational Science and Engineering*, Vol.7, No.1, (2012).

[8] Chapman B, Jost G & Van Der Pas R, *Using OpenMP: portable shared memory parallel programming*, MIT press, Vol.10, (2010).

[9] Kim W & Kim C, "Active contours driven by the salient edge energy model", *IEEE Transactions on Image Processing*, Vol.22, No.4, pp.1667–1673, (2013).

[10] Zhao J, Liang G, Yuan Z & Zhang D, "A new method of breakpoint connection using curve features for contour vectorization", *Electronics and Electrical Engineering*, Vol.18, No.9, pp.79–82, (2012).

[11] Zhao J, Chen B, Sun M, Jia W & Yuan Z, "Improved algorithm for gradient vector flow based active contour model using global and local information", *The Scientific World Journal*, (2013).

[12] Wang L, Li C, Sun Q, Xia D & Kao CY, "Active contours driven by local and global intensity fitting energy with application to brain MR image segmentation", *Computerized Medical Imaging and Graphics*, Vol.33, No.7, pp.520-531, (2009).

[13] Shen J, Fang J, Sips H & Varbanescu AL, "An application-centric evaluation of OpenCL on multi-core CPUs", *Parallel Computing*, Vol.39, No.12, pp.834-850, (2013).

[14] Karantasis KI, Polychronopoulos ED, Panourgias KT & Ekaterinaris JA, "Accelerating the simulation of brain tumor proliferation with many-core GPUs", *Journal of Computational Science*, Vol.3, No.5, pp.306-313, (2012).

[15] Wan J & Liu Y, "Hybrid MPI-OpenMP Parallelization of image reconstruction", *Journal of Software*, Vol.8, No.3, pp.687-693, (2013).