Journal of Advances in Computer Research
Quarterly pISSN: 2345-606x eISSN: 2345-6078
Sari Branch, Islamic Azad University, Sari, I.R.Iran
(Vol. 10, No. 1, February 2019), Pages: 11-24
www.jacr.iausari.ac.ir

High Performance Implementation of Fuzzy C-Means and Watershed Algorithms for MRI Segmentation

Farnaz Hoseini✉, Ghader Mortezaie Dekahi
Department of Computer Engineering, Faculty of Engineering, Shahriar Institute of Higher Education, Astara, Iran
farnazhoseini@shahriar.ac.ir; moertezaei@shahriar.ac.ir

Received: 2018/05/06; Accepted: 2018/07/08

Abstract

Image segmentation is one of the most common steps in digital image processing. The area many image segmentation algorithms (e.g., thresholding, edge detection, and region growing) employed for classifying a digital image into different segments. In this connection, finding a suitable algorithm for medical image segmentation is a challenging task due to mainly the noise, low contrast, and steep light variations of medical images. Due to the inherently parallel nature of image segmentation algorithms, they suit well for implementation on a Graphics Processing Unit (GPU). The main goal of this paper is to improve the performance of fuzzy c-means clustering through the parallel implementation of this algorithm. Although fuzzy c-means clustering is an important iterative clustering algorithm, it is computationally intensive and uses the same data between the iterations. The center of the clusters changes in each iteration, which requires a considerable amount of time for large data sets. The parallel fuzzy c-means clustering is implemented by applying pipeline parallelism on GPU. The experimental results show that the performance is improved up to 23.35x. Next, the watershed algorithm is applied to the final segmentation. In this paper using parallel fuzzy c-means clustering and computations we have attained competing results with other papers. The implementation results on the BRATS2015 show that the accuracy of diagnosis in Dice Similarity Coefficient metric 97/33% is obtained. This improvement is achieved using enhancing edges and reducing noises in images.

Keywords: *Parallel Fuzzy C-Means Clustering; Watershed Algorithm; FCM; CUDA; GPU*

1. Introduction

Image segmentation is a technique that partitions an image into multiple segments [1]. This technique is used in different applications such as medical image processing especially in Magnetic Resonance Imaging (MRI) [2]. Algorithms for segmentation of medical images are divided into three categories. The first category includes low-level techniques such as the use of intensity thresholds and region growing. The second category is determined by the application of uncertainty models and optimization methods. Lastly, the third category includes higher-level knowledge such as a priori information into the segmentation process [3]. Fuzzy C-Means (FCM) clustering has been widely used in the second category of image segmentation, which allows one piece of data to be assigned to two or more clusters. FCM is one of the multi-valued logics that allow medium values member of one fuzzy set also be members of other fuzzy sets

in the same image [4]; however, FCM clustering should probably be a time-consuming process and also requires a considerable amount of memory to cluster large data. For example, for an MR image of size 2000 × 950 pixels, FCM clustering takes 180.43705 seconds. The same data is used in FCM clustering between the iterations and it is a good candidate to be mapped to the parallel processors [5].

Most of the applications with a sequential algorithm can no longer rely on technology scaling to improve performance, while image-processing applications with high degree of parallelism are ideally excellent source for the multicore platforms. The major aim of parallel processing is not only improving the high performance of images but also giving a solution to reduce processing time and better utilization of resources [6]. Studies have shown that the bottleneck of limited processor speed affects the image processing algorithms in software implementation [7]. The recent opening of Graphics Processing Units (GPU) use NVIDIA CUDA application programming interface and offer a powerful platform with parallel calculation capabilities [8]. GPUs are Single Instruction Multiple Data (SIMD) devices that are considered inherently data-parallel [9]. Because of a high speed, programmability, low cost, and more inbuilt execution cores, GPUs are more popular than other image processing devices [10]. The Parallel Computing Toolbox (PCT) along with the Distributed Computing Server (DCS) is a commercial product offered by the Math Works Inc. The PCT provides functions for parallel for-loop execution, creation/manipulation of distributed arrays as well as message passing functions for implementing fine-grained parallel algorithms [11].

Since GPUs are mainly based on multicourse general-purpose processors and can provide a vast number of simple, data-parallel deeply multithreaded cores, and high memory bandwidths [12], we chose them as a suitable tool for parallelism. As an extended version of [13], the present work was conducted to improve the performance of FCM clustering by choosing parallel FCM as the objective function and combining it with the watershed algorithm (PFCM-WS). We have already used the combination of FCM clustering with a watershed algorithm for image segmentation especially for MR images in a cascade order form [14]. To have a more accurate segmentation in this paper, we used some useful pre-processing operation. FCM clustering is parallelized using Compute Unified Device Architecture (CUDA) technology on GPU platform [15-16]. The experimental results show that the performance is improved up to 23.35x. Next, the extracted boundaries of clusters are used as input for the watershed algorithm. This algorithm finds boundaries between regions based on discontinuities in intensity levels. The results show the segmentation algorithm improves edges and reduces noises by an accuracy of 97/33%.

The rest of this paper is organized as follows. Background and related works are discussed in section 2; Background information involves explaining about FCM clustering, watershed algorithm and introduces GPUs. In Section 3, the methodology of this paper is presented. In Section 4, experimental results are shown. Finally, theconclusion is presented in Section 5.

2. Background and Related Works

Background information about the construction of FCM clustering and watershed algorithm, also a brief description of GPU is presented in this section.

2.1 Fuzzy C-Means Clustering

FCM is a method of clustering which allows one piece of data to belong to two or more clusters that frequently used in pattern recognition. It's improved by Bezdek in 1981[17]. This algorithm specifies the following steps:

Step1) Initialize the membership matrix (U) with random values that has constraints in Equation (1).

$$\sum_{i=1}^{c} u_{ij} = 1, \forall j = 1,..., n \tag{1}$$

Step2) Fuzzy cluster centroids (ci) Calculates by using Equation (2).

$$c_i = \frac{\sum_{j=1}^{n} u_{ij}^{m} x_j}{\sum_{j=1}^{n} u_{ij}^{m}} \tag{2}$$

Step3) Compute dissimilarity between centroids and data points Calculate using Equation (3). Stop if either it is below a certain threshold value or its improvement over previous iteration.

$$J(U, c_1, c_2,..., c_c) = \sum_{i=1}^{c} J_i = \sum_{i=1}^{c} \sum_{j=1}^{n} u_{ij}^{m} d_{ij}^{2} \tag{3}$$

Step4) Compute a new U using Equation (4) then go to Step 2.

$$u_{ij} = \frac{1}{\sum_{k=1}^{c} \left(\frac{d_{ij}}{d_{kj}} \right)^{2/(m-1)}} \tag{4}$$

2.2 Watershed Algorithm

The concept of watershed algorithm was introduced in 1991 by Vincent and Soille [18]. In a landscape or topographic which is eroded by water, when a drop of water falling on it starts with their local minima. Dams are built at points where water coming from different basins. The immersion process is stopped, when water attains the highest altitude in the landscape. Finally, the landscape is divided into separated regions using dams, called watersheds [19]. This is a classical algorithm used for segmentation, which is for separating different objects in an image [20].

2.3 Graphics Processing Unit

The past few years, microprocessor design has followed two separate paths: on the one hand, there were the multi-core multiprocessors (e.g. Intel Core i7 is a

microprocessor with four processing cores, each of them implementing a complete x86 instructions set), On the other hand, the many-core microprocessors (e.g. graphics processors) developed mainly by the video-card manufacturers [21]. The recent openings of GPU use NVIDIA CUDA API and offer a powerful platform with parallel calculation capabilities [22]. In February 2007, NVIDIA released the CUDA programming model to be used with their GPUs to make them available for general purpose application programming which is based on an extended ANSI C language and a runtime environment and allows the programmer to specify explicitly data parallel computation [23]. Modern GPUs are very efficient at retouching computer graphics and image processing. GPUs highly parallel structure makes them more effective than general-purpose CPUs for algorithms where processing of large blocks of data is done in parallel.

2.4 Related Works

In this algorithm, each parameter has its own learning rate, and its scale is proportionally changed to the total squared history of the previous partial derivatives [10]. Therefore, the learning rate for parameters with a large partial derivative history is rapidly reduced, and the minimal reductions are experienced for parameters with a small minor derivative history. Totally, the algorithm finds better scrolling speeds when facing gentle slope directions. This algorithm possesses theoretical properties in the domain of convex cost functions. In practice, however, using this method in deep networks and dividing the learning rate on the aggregation of the entire partial derivative history, in some cases, the learning rate is reduced reaching the optimal point.

Pipeline parallelism and domain decomposition are two parallelization strategies employed by some researchers. A serial k-means algorithm (the hard c-means algorithm) consists of a group and objects in a data set into k cluster [24]. To obtain the acceptable computational speed of massive datasets, most researchers have tried to parallelize this algorithm [25- 28]. In order to solve the high computational cost, researchers have focused on parallelism of FCM clustering [29- 30]. FCM is very popular because it can be generalized much easier than the hard c-means clustering in many applications. In [31], an implementation of a parallel FCM cluster analysis was presented to optimize both aspects of cluster investigation: the number of clusters and the determination of clusters' centers. The clusters' centers were calculated by assigning the degrees of membership of records to clusters and the determination of clusters' centers optimized for a given dataset using the PBM index. The main contribution of this work was the integration of the cluster validation index in the optimization process, allowing the optimization of the overall parallel process. In [32], an efficient and scalable FCM clustering was introduced for processing input data based on graphics hardware. It gained speed in computational time and handled intermediate results within the GPU with reusability of shaded programs and minimizing the use of GPU resources. In [33], the parallel FCM algorithm for clustering large data sets was presented. The algorithm designed to run on parallel computers of the Single Program Multiple Data (SPMD) model type with the Message Passing Interface (MPI). A large data set from an insurance company was used for testing the algorithm, which demonstrated almost ideal speedups for larger data sets. In [34], a method using CUDA programming tools was proposed, which significantly speed up FCM computations with multiple cores built in a graphic card. In [35], an efficient method to cluster data points of all the images at once

is proposed using the gray level histogram in the FCM algorithm to minimize the time for segmentation and the space required. The results showed that the algorithm was almost twice as fast as the conventional FCM. In [36], to extract tumor region from MR brain image, cluster centroids were initialized through data analysis of tumor region, followed by applying reconstruction-based morphological for enhancing its performance for brain tumor extraction. The results show that simple FCM could not properly segments the region of interest, whereas enhanced algorithm effectively extracts the tumor region. In [37], a new algorithm transformation method was proposed, through which the original image was partitioned using FCM and then the controlled action of the edge indicator function was increased. The result of FCM segmentation was used to obtain the initial contour of the level set method. With the new edge indicator function, results of image segmentation show that the improved algorithm can exactly extract the corresponding region of interest. In [38] a parallel floating centroids method (PFCM) had been used to be a high performance neural network classifier. They proposed PFCM to speed up the FCM based, especially for a large data set.

In [39], three different techniques were implemented to extend FCM clustering to very large data. It compared these techniques based on 1) sampling followed by non-iterative extension; 2) incremental techniques that make one sequential pass through subsets of the data; and 3) kernelized versions of FCM that provide approximations based on sampling. Both loadable and very large datasets were used in this work to conduct the numerical experiments that facilitate comparisons based on time and space complexities as well as speed and quality of approximations. The results showed that random sampling plus extension FCM, bit-reduced FCM, and approximate kernel FCM are good choices to approximate FCM for very large data sets. In [40], a fuzzy level set algorithm was proposed to facilitate medical image segmentation. The algorithm was enhanced by locally regularized evolution since such improvement facilitates level set manipulation, which leads to a more robust segmentation. The performance of this algorithm was evaluated using MR brain Images. In [41], a novel FCM model was proposed based on the spatial similarity information. The proposed FCM was formulated by modifying the distance function to compensate the noise using both the non-local information and spatial structure similarity measurement. In [42], the improved version of FCM clustering and watershed algorithm was implemented. They proposed an effective method for the initial centroid selection based on histogram calculation in FCM clustering and proposed an atlas-based marker detection method for avoiding the over-segmentation problem in the watershed algorithm. The accuracies of these recent studies have had a lot of improvement over the conventional methods of the past, it is important to further improve the performance. The main limitation to these algorithms is time-consuming when they face with large amount of data. With regard to the considerable Cluster centroid updates of FCM clustering in most related works, reducing the time is crucial.

3. The Proposed PFCM-WS Algorithm

The Parallel Fuzzy C-Means & Watershed (PFCM-WS) algorithm for MR images segmentation of brain consists of five main steps which are shown in Fig. 1. In the first step, the input image is converted to grayscale image; the second step is related to the image processing operations and preparation of the input image; in the third step

derivation of image is calculated for detecting edges of images; in the fourth step parallel FCM clustering is used for classification; finally, in step sixth, the image segmentation is performed using the watershed function. These steps are explained as follows:

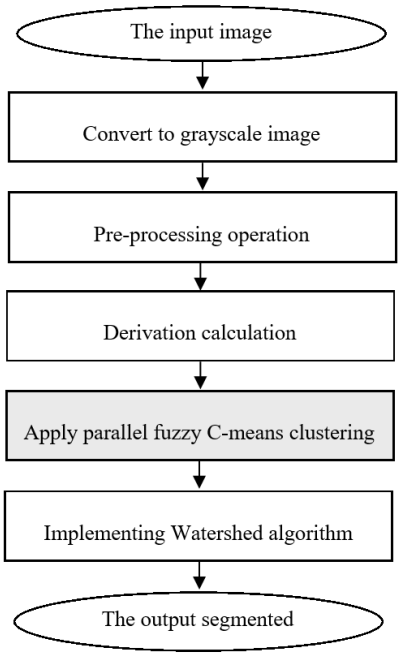


Fig. 1. Blocks diagram of the PFCM-WS algorithm

Step 1) Grayscale image: First, the color input images should be converted to grayscale. Color images have three layers: blue, red, and green but grayscale images have only one layer with different degrees of gray color.

Step 2) The input image pre-processing: In the second step, the N4ITK Bias correction was used for nonuniformity correction due to magnetic field changes. Also, Nyul's lighting normalization was used for intensity normalization throughout the image.

Step 3) Calculate the derivation: One of the conventional methods for detecting edges of grayscale images is the calculation of gradient, which is actually the detection of meaningful discontinuities of intensity values through the calculation of first and second derivation. At this step, derivation of the image is calculated and the square of elements is stored in a new variable, which is used in the next step.

Step 4) Apply parallel FCM clustering: In order to implement the algorithm on GPU, we need to decompose tasks in order to determine which parts of the program can be executed independently. Because of applying the centroids recalculation for many times, it can be a good candidate to isolate on the GPU. The pipeline is prepared for the computation before running the algorithm. The parallel FCM clustering is presented in the following six tasks:

Task 1) Define data array: Although data array is calculated just once in the algorithm, for the convenience, a kernel is defined on the GPU for this task and step 3 data's are assigned to the device-shared memory.

Task 2) Specify distance and membership kernels: In the previous task, a matrix (U) is built whose factors are numbers between 0 and 1. Now, the new cluster

membership matrix is calculated and the geometric center of its local data is computed. The components of the matrix U can be any value between 0 and 1, but the sum of the components of each column (i.e., the degree of membership) should equal 1.

Task 3) Cluster centroid updates: There are three main matrixes for the standard FCM clustering: the cluster centers, the memberships of each cluster, and the number of each member of iterations. In this task, the third matrix is overlooked, the cluster centers are specified, and then memberships of each cluster is calculated and stored in a thread matrix. The memberships are updated with the new values based on the distances of each observation to each of the cluster center. These results are copied into the host memory and then the cluster sizes are calculated and returned back to the device shared memory.

Task 4) Aggregate and reduce cluster centers: Before aggregation, centroid update calculation is completed to avoid loss of time in switching between kernels. The changed data are stored in a new matrix in the device-shared memory. The old data array is swapped with new data using their addresses.

Task 5) Check the Stopping Condition: The new cluster memberships are calculated and stored in a different matrix. This step is done to compute complete independence. Maximum difference for the current array and the previous array for all memberships is compared. If the termination criterion in Equation (5) is obtained, this iteration will stop, otherwise, would return back to Task 3 (ε is a termination criterion between 0 and 1 and k are the iteration steps).

$$\max\{| \quad U_{(K+1)} - U_{(K)} |\} \leq \varepsilon \qquad (5)$$

Task 6) Cluster centers (the results) are copied back to the CPU as output: In the last task, the iteration cluster distance and memberships are computed. Finally, the cluster centers are copied back to the host memory as output and all arrays in the device-shared memory are freed. Since this task happens just once, the time is insignificant compared to the implementation of the iterative portion of the algorithm. The parallel FCM clustering analysis is shown in Fig.2. This part of the output is used as input for the final segmentation.

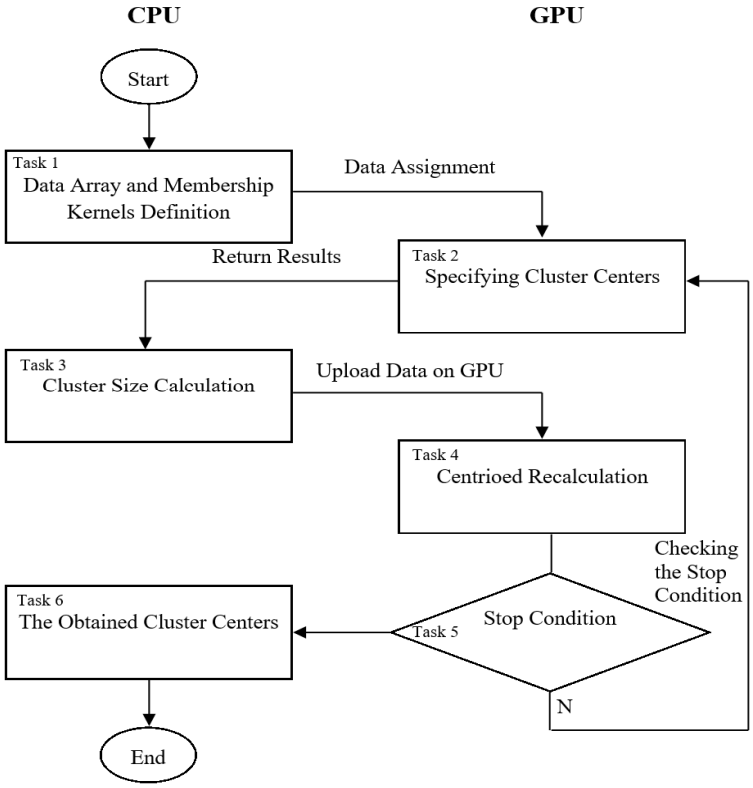


Fig. 2. Blocks diagram of the PFCM-WS algorithm

Step 5) Implementing watershed algorithm: Segmentation algorithms for grayscale images generally act based on either of two properties of intensity; i.e., discontinuity and similarity. The main goal of image segmentation is to divide it into several areas. We did it by finding the boundary between the areas based on discontinuities in the intensity levels. Linear space filtering is used to perform a neighboring process on classes determined in this step. Among the filters of linear spatial filtering, we applied Sobel filter and Replicate filter. To extract these characteristics in Sobel filter, first, the gradient magnitude is obtained followed by applying a 3×3 vertical and horizontal gradient kernel to the image. Finally, the square root for determining the boundary of clusters is calculated. Considering the boundary of clusters as input for the watershed algorithm, this algorithm is applied to the images for the final segmentation. In the two-dimensional images, the number of areas is eight (by default); thus, we selected the same number for the proposed algorithm. In addition, Sobel linear spatial filter with replicate method was chosen to expand an image size.

4. Experimental Results

In this section, evaluation of the PFCM-WS algorithm is presented.

4.1 Datasets and Evaluation Criteria

This study was conducted with the objective of improving the segmentation of medical images. To achieve the best program, testing of the program was performed on different images of 7-Tesla MR images of the brain. MR images were extracted from the BRATS2015 Challenge dataset [44], which includes 230 brain images. There are

two binary maps, one obtained by the model (P), and the other by the consensus of experts (T) available in the dataset. Therefore, the Dice similarity coefficient metric is calculated according to Equation (6) using the model output. In other words, this metric is the ratio of the overlapping region to the average region specified by the model and the expert. In the Dice Similarity Coefficient metric, P and T represent the model predictions and ground truth labels, respectively.

$$\mathrm{Dice}(\mathrm{P},\mathrm{T}) = \frac{|\mathrm{P} \wedge \mathrm{T}|}{(|\mathrm{P}|+|\mathrm{T}|)/2} \qquad (6)$$

4.2 Platform and Results

The experimental process was carried out using a NVIDIA GeForce series GeForce GTX 1080 Ti. In CUDA, programs are known as kernels, which they have a SIMD programming model [10]. The kernels were run on a grid, that is an array of blocks; and each block is an array of threads. Pipeline pattern which is shown in Fig.3 used in this paper (Task 3) that one of the standard parallel communication patterns in MATLAB [43]. In pipelining, each PID (the program identity) receives data, processes it, and then sends it to another PID.

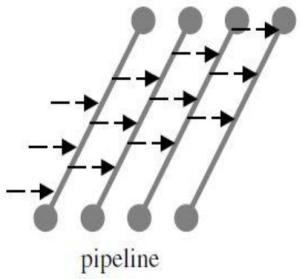


Fig. 3. Pipeline parallel communication patterns

First, we imported our inputs to GPU memory and then using CUDA kernel we executed our calculations (specify cluster centers and recalculate them) on the GPU. We executed the PFCM-WS algorithm on CPU and GPU many times and measured the smallest execution time. To compare the performance of the algorithm in both serial and parallel implementation, the number of threads is 512 in all implementation. Due to the size of images (numbers of pixels on the axis X of the input image); block numbers were calculated for all sizes of input images and all algorithm runs. For example, for images of size 1024×700 pixels, the number of pixels on the axis X of the image matrix is divided into threads size and the number of blocks is equal to 2 (1024/512). When a large size of images exists, it does not have any effect on our device-shared memory usage, because we just increase the number of blocks and keep the device shared memory allocations in a thread. The runtimes of our GPU and CPU implementations were compared for datasets with different size images; the speedup

results are shown in Fig.4. By applying a GPU-based implementation, we reduced the CPU implementation and obtained a speedup of the system up to 23.35x times.

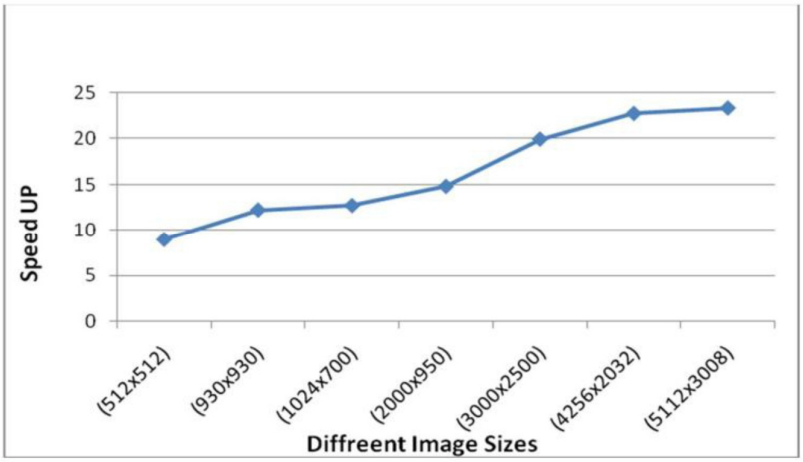
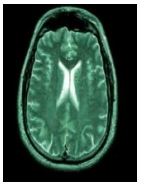
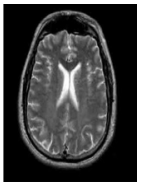
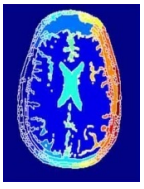


Fig. 4. Results of speedup obtained from executed the PFCM-WS algorithm on different MR images

Changes in the parameters (fuzzy clusters, the neighborhood in the watershed algorithm, filter types, and filter methods) can result in the subsequent changes in output results. In other word, changes in filters type and the ways that they are employed may affect the percentage of correct detection in our calculation. There are 612 evaluation modes for all states. In this paper, to improve the image segmentation, we used Sobel filter and applied replicate filter method. Table 1 presents one of tasted MR images outputs of the segmentation steps by definition.

***Table 1**: Results of the PFCM-WS algorithm steps on one tasted MR images*

The input image	The grayscale image result	The output image from parallel fuzzy c-means clustering	The gradient magnitude (gradma)	The output segmented image
	(Step 1)	(Step 4)	(Step 5)	
				

After testing the program on 148 different sizes of MR images of the brain and calculating the average results, the accuracy of correct diagnosis was obtained as 97/33%, which is acceptable compared to those proposed in other similar papers. In Table2, a few examples of the segmentation results and percentage of correct and incorrect diagnoses of classification are presented.

Table 2: Some examples for comparing the results of the PFCM-WS algorithm with those of the proposed segmentation algorithm

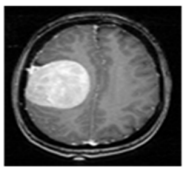
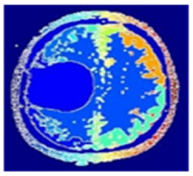
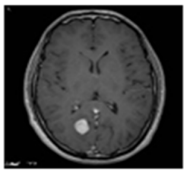
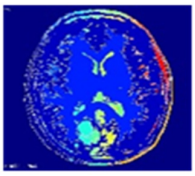
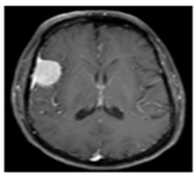
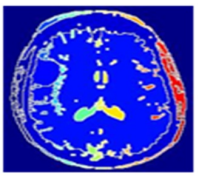
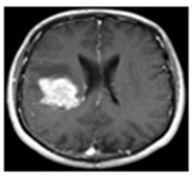
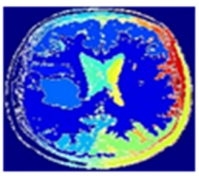
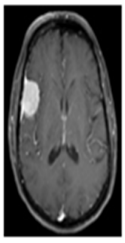
Image Segmentation Results			
Test Images [44]	Results of The FCM-WS Algorithm	The Results For Accuracy of Correct Diagnosis (%)	
		Percentage of pixels that are Classified correctly	Percentage of pixels that are Classified incorrect
		99.1616	0.8384
		95.8013	4.1987
		98.2852	1.7148
		97.7719	2.2281

Table 3 presents diagnosis rate using the proposed PFCM-WS algorithm compared to some related works. As the last row of the table shows, the accuracy of the PFCM-WS algorithm is more than other previous techniques.

Table 3: The comparison of the measured accuracy of the PFCM-WS algorithm with some related works for MRI segmentation

Test Image [44]	Reference	Detection methods	Segmentation Accuracy (%)
	[39]	Modified Version of Fuzzy C-Means Clustering	87.07
	[40]	Modified Adaptive Fuzzy C-Means Clustering	64.00
	[41]	A Novel Fuzzy C-Means Clustering	93.19
	[42]	Improved Version of Fuzzy C-Means Clustering	88.91
	#	The Proposed PFCM-WS Algorithm	98.28

5. Conclusion

In this paper, a segmentation algorithm using parallel FCM clustering and watershed algorithm is proposed (PFCM-WS). We observed that the CPU implementation increased by increasing the image size in traditional FCM clustering. It was deduced that the processing time required for implementing the parallel FCM clustering using CUDA programming tools is drastically reduced with the GPU. In addition, processing speedup grows with increasing the image size. Through a GPU-based implementation, we reduced the CPU implementation and obtained a speedup of the system up to 23.35x times. Although a watershed algorithm is able to make all parts of the medical images autonomously, it causes too much segmentation and is also sensitive to false edges. In this paper, by applying parallel FCM clustering before applying watershed markers, the problem of too much segmentation was solved and accuracy rate of 97/33% for the correct segmentation of the images was obtained. According to the obtained percentage, it is observed that the PFCM-WS algorithm is a proper method for segmentation of MR images of the brain and also has a high diagnostic accuracy. For user-friendly processing interfaces, it is desired to develop them with less effort. Because of the limited memory space and the number of parallel GPUs, the use of higher-volume data is difficult to handle. With the advancement of GPUs and the use of libraries that distribute processing across multiple GPUs and multiple machines, this problem can be addressed. In the near future, we look forward to seeing improvements in the programmability and generality of future GPU architectures. The decreased computation time can be immediately attributed to the parallel environment.

References

[1] Yambal M, Gupta H (2013) Image segmentation using fuzzy C means clustering: a survey. International Journal of Advanced Research in Computer and Communication Engineering 2(7): 2927-2929.

[2] Alanazi HO, Abdullah AH, Qureshi KN (2017) A Critical Review for Developing Accurate and Dynamic Predictive Models Using Machine Learning Methods in Medicine and Health Care. Journal of medical systems 41(4): 69.

[3] Huang X, Tsechpenakis G (2009) Medical image segmentation. Information Discovery on Electronic Health Records 10, pp 251-289.

[4] Gambhir S, Malik SK, Kumar Y (2016) Role of Soft Computing Approaches in HealthCare Domain: A Mini Review. Journal of medical systems 40(12): 287.

[5] Jose A, Ravi S, Sambath M (2014) Brain tumor segmentation using k-means clustering and fuzzy c-means algorithms and its area calculation. International Journal of Innovative Research in Computer and Communication Engineering, 2(3): 3496-3501.

[6] Harris C (2004) Using programmable graphics hardware to implement the fuzzy c-means algorithm. Honours Dissertation, The University of Western Australia.

[7] Wang L, Ma Y, Zomaya AY, Ranjan R, Chen D (2015) A parallel file system with application-aware data layout policies for massive remote sensing image processing in digital earth. IEEE Transactions on Parallel and Distributed Systems 26(6): 1497-1508.

[8] Kruliš M, Lokoč J, Skopal T (2016) Efficient extraction of clustering-based feature signatures using GPU architectures. Multimedia Tools and Applications, 75(13): 8071-8103.

[9] Garcia V, Debreuve E, Barlaud M (2008) Fast k nearest neighbor search using GPU. IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops, pp1-6.

[10] Park SI, Ponce SP, Huang J, Cao Y, Quek F (2008) Low-cost, high-speed computer vision using NVIDIA's CUDA architecture. 37th IEEE Applied Imagery Pattern Recognition Workshop, pp 1-7.

[11] Di Salvo R, Pino C (2011) Image and video processing on CUDA: state of the art and future directions. 13th WSEAS international conference on Mathematical and computational methods in science and engineering, pp 60-66.

[12] Cheikh TLB, Beltrame G, Nicolescu G, Cheriet F, Tahar S (2012) Parallelization strategies of the canny edge detector for multi-core CPUs and many-core GPUs. IEEE 10th International Conference on New Circuits and Systems , pp 49-52,

[13] Hoseini F, Shahbahrami A (2016) An efficient implementation of Fuzzy C-Means and watershed algorithms for MRI segmentation. In 8th International Symposium on Telecommunications (IST) IEEE, pp 178-184.

[14] Hoseini F, Haghipour S, DaeiSorkhabi A, (2014). Improvement of segmentation on MRI images using fuzzy clustering C-means and watershed marker control algorithm. Indian journal of Scientific Research 4(3): 447-451.

[15] Stratton JA, Stone SS, Wen-mei, WH. (2008) MCUDA: An Efficient Implementation of CUDA Kernels for Multi-core CPUs. In Languages and Compilers for Parallel Computing, Springer Berlin Heidelberg 208, pp16-30.

[16] Masoumi H, Behrad A, Pourmina MA, Roosta A (2012) Automatic liver segmentation in MRI images using an iterative watershed algorithm and artificial neural network. Biomedical Signal Processing and Control 7(5): 429-437.

[17] Bezdek JC, Ehrlich R, Full W (1984) FCM: The fuzzy c-means clustering algorithm. Computers & Geosciences, 10(2-3):191-203.

[18] Vincent L, Soille P (1991) Watersheds in digital spaces: an efficient algorithm based on immersion simulations. IEEE Transactions on Pattern Analysis & Machine Intelligence 6, pp 583-598.

[19] Jiang S. Parallelization of Image Segmentation Algorithms. University of Florida.

[20] Zhang X, Jia F, Luo S, Liu G, Hu, Q (2014) A marker-based watershed method for X-ray image segmentation. Computer methods and programs in biomedicine 113(3): 894-903.

[21] Vajda S, Santosh KC (2016) A Fast k-Nearest Neighbor Classifier Using Unsupervised Clustering. In International Conference on Recent Trends in Image Processing and Pattern Recognition Springer pp 185-193.

[22] Krishnamurthy A, Samsi S, Gadepally V (2014). Parallel MATLAB Techniques. Image Processing Book.

[23] Krishnamurthy A, Nehrbass J, Chaves JC, Samsi S (2007) Survey of parallel MATLAB techniques and applications to signal and image processing. IEEE International Conference on Acoustics, Speech and Signal Processing 4, pp IV-1181.

[24] MacQueen J. (1967) Some methods for classification and analysis of multivariate observations. In Proceedings of the fifth Berkeley symposium on mathematical statistics and probability 1(14): pp 281-297.

[25] Kerdprasop K, Kerdprasop N (2010) Parallelization of k-means clustering on multi-core processors. In Proceedings of the 10th WSEAS international conference on Applied computer science 10, pp 472-477.

[26] Shalom SA, Dash M, Tue M (2008) Efficient k-means clustering using accelerated graphics processors. In International Conference on Data Warehousing and Knowledge Discovery, pp 166-175.

[27] Ahmed MF (2014) Parallel Implementation of K-Means on Multi-Core Processors. Computer Science & Telecommunications, 41(1):53-61.

[28] Hong-Tao B, Li-li H, Dan-tong O, Zhan-shan L, He L (2009) K-means on commodity GPUs with CUDA. Congress on Computer Science and Information Engineering 3, pp 651-655.

[29] Modenesi MV, Costa MC, Evsukoff AG, Ebecken NF (2006) Parallel fuzzy c-means cluster analysis. In International Conference on High Performance Computing for Computational Science Springer, pp 52-65.

[30] Hung MC, Yang DL (2001) An efficient fuzzy c-means clustering algorithm. In International Conference on Data Mining IEEE, pp 225-232.

[31] Al-Ayyoub M, Abu-Dalo AM, Jararweh Y, Jarrah M, Al Sa'd M (2015) A gpu-based implementations of the fuzzy c-means algorithms for medical image segmentation. The Journal of Supercomputing 71(8): 3149-3162.

[32] Shalom SA, Dash M, Tue M (2008) Graphics hardware based efficient and scalable fuzzy c-means clustering. In the 7th Australasian Proceedings of Data Mining 87, pp 179-186.

[33] Kwok T, Smith K, Lozano S, Taniar D (2002) Parallel fuzzy c-means clustering for large data sets. Euro-Par parallel processing, pp 27-58.

[34] Rowińska Z, Gocławski J (2012) Cuda based fuzzy c-means acceleration for the segmentation of images with fungus grown in foam matrices. Image Processing & Communications 17(4): 191-200.

[35] Ravi A, Suvarna A, D'Souza A, Reddy GRM (2013) A parallel fuzzy c means algorithm for brain tumor segmentation on multiple mri images. In International Conference on Advances in Computing Advances in Intelligent Systems and Computing, pp 787-794.

[36] Sohi N, Kaur L, Gupta S (2012) Performance Improvement of Fuzzy C-mean Algorithm for Tumor Extraction in MR Brain Images. International Journal of Computer Applications, 59(5):40-45.

[37] Saikumar T, Yugander P, Murthy PS, Smitha B (2013) Image Segmentation algorithm Using Watershed transform and Fuzzy C-Means Clustering on Level set method. International Journal of Computer Theory and Engineering 5(2): 209-213.

[38] Havens TC, Bezdek JC, Leckie C, Hall LO, Palaniswami M (2012) Fuzzy c-means algorithms for very large data. IEEE Transactions on Fuzzy Systems, 20(6): 1130-1146.

[39] Wang L, Yang B, Chen Y, Chen Z, Sun, H (2014) Accelerating FCM neural network classifier using graphics processing units with CUDA. Applied intelligence 40(1): 143-153.

[40] Vadgure S, Thakre P (2014) Detection of Brain Tumor from MRI of Brain Using Fuzzy C-mean (FCM). International Journal of Science, Engineering and Technology Research (IJSETR) 3(8): 2222-2230.

[41] Wang S, Geng Z, Zhang J, Chen Y, Wang J (2014) A fuzzy C-means Model Based on the Spatial Structural Information for Brain MRI Segmentation. International Journal of Signal Processing, Image Processing and Pattern Recognition 7(1): 313-322.

[42] Benson CC, Deepa V, Lajish VL, Rajamani K (2016) Brain tumor segmentation from MR brain images using improved fuzzy c-means clustering and watershed algorithm. In International Conference on Advances in Computing, Communications and Informatics (ICACCI) IEEE, pp.187-192.

[43] Samsi S, Gadepally V, Krishnamurthy A (2010) MATLAB for signal processing on multiprocessors and multicores. IEEE Signal Processing Magazine, 27(2): 40-49.

[44] http://www.braintumorsegmentation.org/