Study of Parallel Image Processing with the Implementation of vHGW Algorithm using CUDA on NVIDIA'S GPU Framework

Sanjay Saxena, Shiru Sharma, Neeraj Sharma

Abstract - This paper provides an effective study of the implementation of parallel image processing techniques using CUDA on NVIDIA GPU framework. It also discusses about the major requirements of parallelism in medical image processing techniques. Additional important aspect of this paper is to develop vHGW(van Herk/Gill-Werman morphology) algorithm intended for erosion and dilation proposed for diverse types of structuring elements of random length and along with random angle parallely on NVidia's GPU GeForce GTX 860M. The main motive behind implementing image morphological operations is its importance for extracting components of an image. That can be beneficial in the demonstration and explanation of shape of the region. These experiments have been implemented on CUDA 5.0 architecture with NVidia's GPU GeForce GTX 860M and got significant results in terms of time.

***Index Terms*: Image Processing, CUDA (Computed Unified Device Architecture), GPU, Parallel Computing, vHGW (van Herk/Gill-Werman morphology)**

I. INTRODUCTION

Real time image processing techniques/issues requisite a massive extent of processing supremacy, expertise in computing and massive resources to perform the computation and operations. These constraints generally appear on the system due to the mammoth dimension, nature of the images to be processed and severe augmentation of data from kilo to terabyte in last few years. Parallel processing of the images is found to be the most effective approach to handle this issue. For the efficient and quick implementation of image processing parallel, There are several tools and techniques are available such as CUDA, GPU, PCT of MATLAB™, Open-CV, and Open-CL. In the present day situation GPU(Graphics Processing Unit) has long been utilized to quicken computer aided design, computer fluid dynamics, computational structural mechanics, electronic design automation, 3D gaming, applications, high performance image processing and some other applications [1][2]. NVIDIA introduced its enormously parallel architecture named as CUDA (Computed Unified Device Architecture) in 2006- 2007 and transformed the whole outlook of GPU programming[3], As it provides a perfect environment for developing parallel programs on GPU [2]. In this paper, we have implemented and developed CUDA codes for vHGW algorithm of image morphology. Assessment has been measured between its parallel execution and its sequential execution on the same configuration of physical unit. The algorithms developed in CUDA environment for GPU have been applied for processing abdomen CT images. This paper is divided into five sections. Section II describes about the brief introduction of parallel image processing, Section III presents about the morphological operations and its implementation in NVidia's GPU 860M. Conclusion and future scope is presented in section IV.

This work is financially and technically supported by the Indian Institute of Technology(BHU), Varanasi and International Institute of Information Technology, Bhubaneshwar, India.
Dr. Sanjay Saxena is the Assistant Professor in Computer Science & Engineering at IIIT, Bhubaneshwar, India. Email – sanjay@iiit-bh.ac.in, Phone No - +91 9839677691.
Dr. Shiru Sharma is the Assistant Professor in School of Biomedical Engineering at IIT(BHU), Varanasi, India. Email – shiru.bme@itbhu.ac.in
Dr. Neeraj Sharma is the Associate Professor in School of Biomedical Engineering at IIT(BHU), Varanasi, India. Email – neeraj.bme@itbhu.ac.in

II. IMAGE PROCESSING USING PARALLEL COMPUTING

High performance computing/Parallel Computing has become an indispensable part of today's mainstream computing systems. GPU plays a very vital role for handling time consuming image processing techniques/ issues / algorithms.

A. GPU

GPU is a processor on a graphics card fanatical for compute- rigorous, highly parallel calculation. It is mainly intended for transforming, interpreting and quickening graphics. It consists millions of transistors rather than the central processing unit (CPU), specializing in arithmetic of floating point. The GPU has evolved into a highly parallel, multithreaded processor with excellent computational power. The GPU, since its beginning in 1999, has been a prevailing technology in the field of accelerated gaming and 3D graphics application. It empowers us to run HD (high definitions) graphics, which are the demand of current computing. GPU computation has delivered an enormous edge overhead the CPU with reverence to speed of computation. The major difference among a CPU and a GPU is that, CPU is a serial processor while the GPU is a stream processor. The CPU is optimized for high performance on sequential operations. It makes usage of classy control logic to accomplish the execution of many threads while maintaining resources of a sequential execution. The large cache memory is used to reduce access latency and slow memory bandwidth also contributes to the performance gap.

ISBN: 978-988-14047-4-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

B. CUDA

It is a parallel computing environment deliberated by NVidia for enormously parallel high-performance computing. It is the compute mechanism in the GPU and is reachable by developers through paradigm programming languages. CUDA is proprietary to NVidia video cards. NVidia offers APIs in their CUDA SDK to give a stage of hardware extraction that hides the GPU hardware from developers. Developers no longer have to realize the complexities at the back of GPU. All the particulars of instruction optimizations, memory and thread management are handled by the API. One benefit of the hardware abstraction is that it allows NVidia to change the GPU architecture in the prospect deprived of requiring the developers to absorb a new set of instructions.

C. Image Processing, CUDA and GPU

As we know that Image processing is a form of signals processing in which the input is an image, and the output suggest being an image or whatsoever else that experiences some meaningful processing. Several methods for image processing that exploit CUDA have been anticipated in the last few years. Till now, there are a number of researches have been embedded parallelism in image processing techniques. Following Table 1 illustrates the investigation of some prominent researches of implanting parallel computing in image processing techniques/issues/algorithms.

Table I: Analysis of Parallel Image Processing Techniques on CUDA and GPU

Image Processing Techniques/Issues/Algorithms	Observations
In [4] authors presented comprehensive review on medical image segmentation techniques such as thresholding, region growing , active contour, atlas based and many more.	Authors have pointed out that mainly segmentation and image processing issues each pixel using the similar instructions, and data from a miniature neighbourhood around the pixel so the thread count is too high but some segmentation methods do not process each pixel for e.g. active contours, statistical shape models.
In [5] authors have implemented several image enhancement algorithms such as brightening filter, darkening filter, negative filter and RGB to gray scale filter. Sobel filter for edge detection, low pass and high pass filter are also implemented.	They have used recursive ray tracing (RRT) technique is used. RRT also considers the light coming through the surrounding for e.g. light from reflection, refraction and shadows. They concluded that issues that involves high inter-thread communication increase in value of number of threads per block gives faster result. Issues that do not require inter-thread communication produces better result with lower no of threads.
Region growing algorithm of image segmentation is implemented in [6].	Initially all pixels are considered as seeds segments and fine grained parallel thread are assigned to individual pixels. That merges adjacent segments iteratively using minimization of the average heterogeneity (Spectral and Morphological Features) of image segmentation criteria. Gained speed up around 14.6 to 19.
In [7] region growing image segmentation using fine grained parallel thread is implemented and it also provides alteration to the heterogeneity computation that enhances the segmentation performance	The proposed parallelization scheme exploits the enormous computational competence of the GPU and also gives a good load balancing, as every thread deals with the similar amount of computation. One more benefit of this method is the capability to process each image segment directly, without partitioning the image into tiles.
In [8] median filtering is implemented by the authors on GPU.	They have used different CUDA fundamentals and methods for implementing median filtering and found that it is possible to get gain in response time with an access level GPU, allowing real-time image and audio filtering. Though, The bottleneck of these systems is the PCI Express bus, for devoted and straight bus throughout GPU/RAM and CPU/ GPU the response time is condensed.
Retinal fundus image enhancement is implemented in [9].	Two custom design masks are used for image enhancement.
Cardiac MRI data segmentation has been proposed in [10]. Authors have implemented two algorithms Runge – Kutta – Merson and GMRES method.	They implemented the Runge-Kutta-Merson and GMRES method using CUDA. They compared the CUDA implementations with corresponding multithreaded CPU implementations and found the CUDA implementations were about 3–9 times quicker than the 12-threaded CPU implementations.
In [11] authors have proposed CUDA based techniques for Image Enhancement using Wallis transformation.	In this paper, Wallis filter, is implemented and compared with the sequential implementations on CPU. Authors found that significant speedup achieved, and it increases with image resolution and size of grid

ISBN: 978-988-14047-4-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

	window growing.
Medical image segmentation of hepatic vascular is implemented using fuzzy connectedness method [12].	In this paper, an improved algorithm for (CUDA-kFOE) is proposed by adding a correction step on the edge points. The improved algorithm can greatly enhance the calculation accuracy. It is having two iteration, The affinity computation strategy is altered in the first and a look up table is employed for memory reduction. In the second one, The error voxels because of asynchronism are updated again.
In [13] this work authors have implemented image segmentation using fuzzy connectedness.	In this 240 cores of GPU are grouped into 30 multiprocessors. Each multiprocessors have 8 processing cores, organized in SIMD. CUDA implementation achieved of fuzzy connectedness segmentation a speedup from 7.2x to 14.4x over an optimized CPU implementation. Interactive speed of fuzzy object segmentation is reached.
In this paper [14], A scaling method for image segmentation using level sets is implemented using CUDA. Basically it is developed to find tumour from MRI bran images.	The proposed method does not require the solution of partial differential equation. Scaling approach, that uses fundamental geometric transformations, is used. Thus, the required computational cost reduces. The use of the CUDA programming on the GPU has taken benefits of classic programming as spending time and performance. Therefore results are obtained faster. The use of the GPU has provided to enable real-time processing.
Morphological operations have been implemented in [15].	In this paper, vHGW algorithm has been implemented for different directions not only for horizontal, vertical and diagonal structuring elements, but as well elements along the random line and of the random length. Significant speedup is achieved with CUDA implementation.
In [16] this paper Author's have shown a GPU implementation of quick shift approach which gives a 10 to 50 times speedup, ensuing in a super-pixelization algorithm that can run at 10Hz on 256x256 images.	This method is an exact replica of rapid shift, and could be more speeded up by approximating the density, via subsampling or other methods. It is likely that the implementation would also present similar speedups for exact mean shift.
In [17] author implemented sobel edge detection operator and gaussion blurring on GPU using CUDA.	This work gives an introduction of the CUDA and its benefits. Experiments are implemented on different grid size and blocks and got significant speedup.
In [18] authors computed Haralick's Texture features on GPU for microscopic biological cells.	Implementation condensed the computational time from half a year to around 9 hours of an un-optimized software and to 4 days of an optimized software version. The speedup of the GPU versions scales with the memory bandwidth.
In [19] researchers presented an effective study of Medical Image Processing on GPU.	That paper tells about the implementation of filtering, interpolation, histogram estimation and distance transforms, medical image processing algorithms (registration, segmentation and denoising) and algorithms that are specific to individual modalities (PET, CT, MRI, fMRI, SPECT, DTI, optical imaging, ultrasound, and microscopy) on GPU. The review concludes by providing a few future promise and challenges
Fuzzy C – Means Clustering is implemented in [20]. An extensive study is conducted to examine the dependency amongst the image pixels in the algorithm for parallelization.	In this, the pixels, memberships, and cluster midpoints arrays are defined in a 1 D pattern. The motive is to make sure coalesced memory transactions in GPU. Further, defining those input arrays in 1 D pattern will ease the numeral of CUDA blocks and grid sizes computations. The CUDA block and grid sizes are accordingly defined in 1 D patterns analogous to the input arrays.
In this paper [21] authors presented a parallel implementation based on CUDA of bias field algorithm PBCFCM that is an enhanced version of fuzzy C-means that accurate the in-homogeneity intensity and partitioned the image concurrently.	Firstly, initializing the centroids, vector and the variables, then assign and transport data from CPU to GPU before the loop iteration. Two main kernels are used, one to calculate the membership function and the second one to calculate the estimated bias field. Speed up obtained is significantly increasing as per the GPU's no of cores are increasing.
[22] In this work authors have implemented	In this paper authors perfectly utilized CUDA's huge amounts of threads

ISBN: 978-988-14047-4-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

first order edge detectors such as Roberts, Sobel and Prewitt.	on GPU Ge Force GTX 860M and got significant results. They have tested 60 images consisting three different size (512 X 512, 1024 X 1024, 3072 X 3072) and found speed up around hundred times in terms of percentage.
In [23] authors compared performance of canny edge detection. They introduced hadoop map reduce and CUDA based satellite image processing.	Significant speed up is obtained as the size of image is increasing for canny edge detection. The portion of the algorithm implemented on GPU gives significantly speed up.
Image filtering is implemented using CUDA in [24].	Authors founded that Median Filtering is suitable for implementation using CUDA on GPU. They concluded that we can't implement parallely. Instead we can implement those in which inherent scope of parallelization.

D. GPU and CUDA in Medical Imaging

It is one of the important applications to take benefits of GPU computing to get speeding up. The use of GPUs in the field of Medical Imaging has matured to the point that there are numerous medical modalities shipping with NVIDIA's Tesla GPUs now. In [25] authors have represented reconstruction time for CT(Computed Tomography) on 4 GPUs(Tesla 10 – Series) and 256 CPUs(AMD dual-core Opteron 250) and MRI(Magnetic Resonance Imaging) on 4 GPUs(Tesla 8 – Series) and Quad-core Intel Core 2 Extreme (2.66 GHz).

Till now, there have been done some efficient researches based on medical image processing based on CUDA. In [26] authors presented the past and present work based on GPU accelerated medical image processing, That research covers GPU acceleration of some basic image processing operations such as filtering, interpolation, histogram estimation , distance transforms and the most commonly used algorithms in medical imaging like image registration, image segmentation and image denoising and algorithms that are precise to individual modalities like CT, PET, MRI, SPECT, DTI, ultrasound, optical imaging , fMRI, and microscopy.

In [27] authors worked on the GPU for several medical imaging applications detailed algorithm in that paper represented key element of an automatic segmentation package that delivers investigation of thin segment CT angiographic images of the pelvis and abdomen. The software exploits a staged approach which accomplishes segmentation of the body wall and axial skeleton comprising sub-segmentation and identification of distinct bones inside the pelvis.

So we have seen there are so many cases where there is the extensive need of GPU. Now we have implemented an important issue of Image Processing Techniques on GPU and CUDA.

III. MORPHOLOGY IN IMAGE PROCESSING

The area of mathematical morphology contributes an extensive range of operators in the field of image processing, that is based on the mathematical concepts from set theory. Morphological operators are predominantly useful for the investigation of binary images and its usages include noise removal, edge detection, image enhancement and image segmentation. Some important morphological operations are dilation, erosion, opening, closing, hit or miss Transform, morphological image gradient, skeletonization and many more.

A. vHGW Algorithm

This algorithm is simple and graceful deals with the computation of dilation and erosion with complexity liberated of the dimension of the Structuring Element [15]. It executes for all structuring elements self-possessed of horizontal and/or vertical linear elements, and needs not additional than 3-pixel value assessments for each output pixel. Basic Step of the algorithm is given below

B. Basic Steps of the Algorithm:

STEP1: Image rows are segregated into divisions of length l with (l-1)/2 columns of overlap on each side to form a window of size 2l-1, centered at l-1, 2l-1, 3l-1, ……

STEP 2: For apiece pixel k=0 to (l-1) in a specified window w, a suffix max array M is formed for the pixels leftward of center

a. M[k] = max(w[j]) : j=k …(l-1) and a prefix max array N is formed for the pixels right of midpoint (l-1)…(2l-2),

b. N[k]=max(w[l-1+j]): j=0 …. K

c. (M[k] and N[k] are merged collected to calculate max filter)

STEP 3: For each and every pixel (l-1)/2 <= j < l+ (l-1)/2 in w (the segment of length l), the dilation result is

a. result[j]=max(M[j-m],N[j+m]), where m=(p-1)/2

C. Performance Analysis of Algorithm

There are two phases to the vHGW of above algorithm

I. In pre processing stage, calculating M[k] and N[k] and to find their max requires 2(p-1) comparisons
II. In merging step, merging of R[k] and S[k] requires p-2 comparisons

a. Since this procedure computes the maximum of l windows in total, we have that number of comparisons per window is (2(l-1)+(l-2))/l=3-4/l

ISBN: 978-988-14047-4-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

b. For large p, we have that the preprocessing step requires two comparison operations each element, while the merge step requires one more such comparison.

D. Implementation

vHGW algorithm based upon splitting the input pixel to overlapping segments of size 2l-1(the segment of length l). User is requested to input the angle for the structuring element that decides performed pre processing.
The matrix is reserved in the original form for the Horizontal structuring element.
An altered horizontal dilation that transcribes outcome out into a transposed consequence image so that a consequent horizontal dilation will produce the outcome for the vertical structuring element. It can be executed by captivating transpose of matrix dual stints, earlier processing and later processing. Structuring element at random angle: It uses predefined function to determine the next pixel in the row. User is requested to input the angle for the structuring element. After that the slope is then deliberated to determine $x_{Increment}$ and $y_{Increment}$. This technique sets pixels of line along given random angle, such as single directional array. Value of a particular point i converts the f(line(i)). The line is then transformed and the procedure is repeated till the whole image is processed. Here, the constraint is added, that angles used for processing are considered angle € [0,90] degree.

Then the implementation carries on with the steps of vHGW algorithm as discussed earlier for dilation by 1D structuring element of size p=2N+1. Erosion trails the similar process using least value arrays. The entire number of pixels to be padded is (p-1)/2 at individual side of rows, left and right, with the aim of computing prefix and suffix valueseasily. For the dilation, pad value is taken into account -128 and for the erosion pad value is taken 127[15]. This work presents vHGW algorithm with the shared memory arrays for the max arrays and 2 threads per window, one for each max array. Shared memory usage is anoteworthysubject in the operation. If manifold threads are used for single result, then it is indispensable to have inter-thread communication. Though Shared memory is limited in size making it only viable for smaller images and structuring elements. Manifold windows can be addressed in single CUDA block to accomplish utilization of good thread and scheduling. Result calculation is performed in straightforward way.

E. Setup for Simulation and Results Obtained

Hardware Environment:

Processor: Intel Core i7 -4710 CPU @2.50 GHz
RAM (Random Access Memory): 8 GB
Hard Disk Drive: 1 TB
GPU: Ge Force GTX 860M
Cuda Cores: 640

Software Environment:

System Type: 64 Bit operating System, x – 64- based processor
Image Data Set: 3 Image Data Set (512 X 512, 1024 X 1024, 3072 X 3072) (Obtained from freely available library)
Visual C++ 2010 Express
CUDA Version 5.0
Speed up plot with different structuring elements have been shown in the following figures

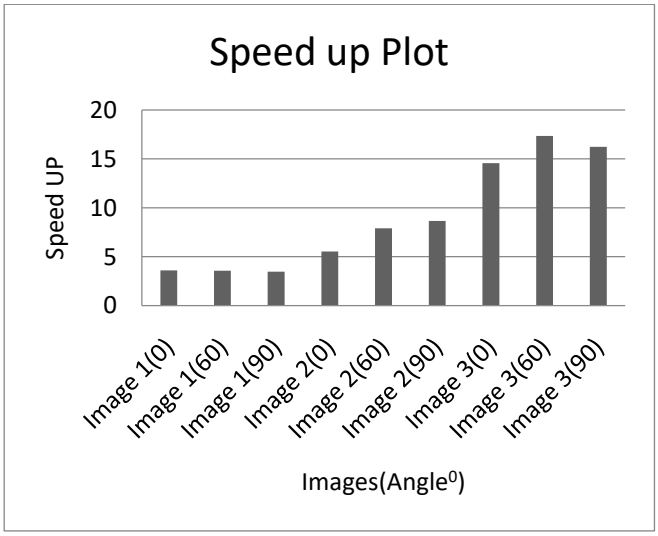

Fig 1: Speed up Plot at Structuring Element 11

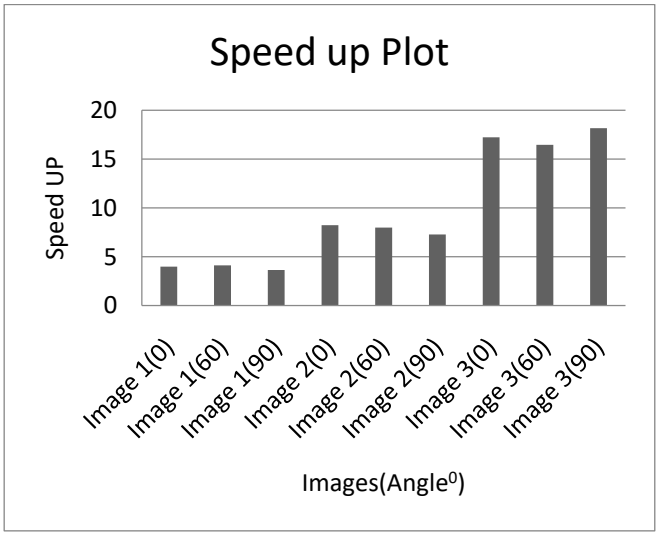

Fig 2: Speed up Plot at Structuring Element 21

F. Discussion

Significant speed up obtained for the vHGW algorithm shows that CUDA facilitates efficient penalization of image processing techniques executed on GPUs. This algorithm has been appreciably implemented in various directions with straight, perpendicular and oblique SE (structuring element). Further, It is also implemented on the elements along with the random line and of the uninformed length. The consequences designate highest performance gain of several times than the conventional sequential implementation of algorithm in terms of execution time with significant speed up with the CUDA code on GPU 860 M. However, It varies with the size of the images, size and shape of the used structuring element. Huge performance gain is accomplished with larger size of images by the effective use of CUDA on GPUs compared with the speed up gain obtained by [15]. It is shown that efficient utilization of CUDA on GPU gives the maximum benefits in terms of time on several application of image processing.

ISBN: 978-988-14047-4-9
ISSN: 2078-0958 (Print); ISSN: 2078-0966 (Online)

IV. CONCLUSION AND FUTURE RECOMMENDATION

In this paper we presented the state-of-art brief literature review of the image processing, its various techniques, major requirement of implanting parallelism using GPU and CUDA in image processing techniques specially in the case of medical imaging. In this paper, an effective comparative analysis of various parallel image processing techniques developed by different researchers were also been done in last few previous years. Later on, we extensively used CUDA programming environment in a highly parallel architecture i.e. GPU. We designed and developed CUDA codes on CUDA 5.0 framework for vHGW algorithm for morphological operators such as dilation and erosion and got significant results with effectively use of CUDA on GPU Geforce 860M. However, performance gain varies with the size of the images, size and shape of the used structuring element. Huge performance gain is accomplished with larger size of images with larger numbers of CUDA cores. For future perspective we can implement shared memory parallel computation through an interconnection network for fastest computing. Other parameters such as granularity, SMP (Symmetric Multi-Processor) can be added to this system to make it more well-organized.

REFERENCES

[1] P. Kaur, "Implementation of image processing algorithms on the parallel platform using MATLAB" in Int. J. Comp. Sci. Eng. Technol., vol. 4(6), 2013, pp. 696-706.
[2] J. Tse, "Image Processing with CUDA" in Master's Thesis, University of Nevada, Las Vegas, August, 2012.
[3] J. Ghorpade, J. Parande, M. Kulkarni, "GPGPU Processing in CUDA Architecture" in arXiv preprint arXiv:1202.4347, 2012
[4] E. Simstad, T.L. Falch, M. Bozorgi, A.C. Elster, F. Lindseth, "Medical Image Segmentation on GPUs – A comprehensive review" in Medical Image Analysis, Elsevier, Vol. 20, 2015, pp. 1-18.
[5] D. Saha, K. Darji, N. Patel, D. Thakore, "Implementation of Image Enhancement Algorithms and Recursive Ray Tracing using CUDA" in 7th International conference on communication, computing and virtualization, Procedia Computer Science, 79, 2016, pp. 516 – 524.
[6] P. N. Happ, R. Q. Feitosa, C. Bentes, R. Farias, "A Region Growing Segmentation Algorithms for GPUs" in Internal Research Report, Maxwell/ Lambda – Dee, Vol 30, 2013.
[7] P. N. Happ, R. Q. Feitosa, C. Bentes, R. Farias, "A Parallel Image Segmentation Algorithms on GPUs" in Proceedings of the 4th GEOBIA, May 7-9, 2012, pp. 580 – 585.
[8] Placido Salvatore Battiato, "High Performance Median Filtering Algorithm Based on NVIDIA GPU Computing", 2016.
[9] A. Arunkant, Y.P. Singh, S. Sharma, "Retinal Fundus Image Enhancement using CUDA Enabled GPU NVidia GT 720M" in International Journal for Technological Research in Engineering, 2016, pp. 194 – 196.
[10] T. Oberhuber, A. Suzuki, J. Vacata, V. E. Zabka, "Image Segmentaion using CUDA implementations of the Runge – Kutta – Merson and GMRES Methods" in Journal of Math – for – Industry, Vol. 3, 2011, pp. 73 – 79.
[11] H. Xio, Y- P. Song, Q – L. Zhou, "Multi – GPU Accelerated Parallel Algorithms of Wallis Transformation for Image Enhancement" in International Journal of Grid and Distributed Computing, vol. 7(1), 2014, pp. 99-114.
[12] L. Wang, D. Li, S. Huang, "An Improved Parallel Fuzzy Connected Image Segmentation method based on CUDA" in BioMed Eng OnLine, BioMed Central, 2016, 15:56.
[13] Y. Zhuge, Y. Cao, J.K. Udupa and R. W. Miller, "GPU Accelerated Fuzzy Connected Segmentation by using CUDA" in Proceedings of IEEE Engineering Medical and Biology Society, 2009, pp. 6341 – 6344.
[14] Z. Guler, A. Cinar, "GPU – Based Image Segmentation using Level Set Method with Scaling Approach" in Computer Science and Information Technology, 2013, pp. 81 – 92.
[15] M. A. Rane, "Fast Morphological Image Processing on GPU using CUDA" in Master of Technology Thesis, Department of Computer Engineering and Information Technology, 2013.
[16] B. Fulkerson, S. Soatto, "Really quick shift: Image segmentation on a gpu," tech. rep., Department of Computer Science, University of California, Los Angeles, 2010.
[17] J. Tse, "Image Processing with CUDA," Master's Thesis, University Of Nevada, Las Vegas, August 2012.
[18] M. Gipp, G. Marcus, N. Harder, A. Suratanee, K. Rohr, R. Konig, R. Manner, "Haralick's Texture Features Computations Accelerated by GPUs in Biological Applications" http://www.nvidia.com/content/gtc/posters/28_gipp__haralick_texture_features.pdf.
[19] A. Eklund, P. Dufort, D. Forsberg, S. M. Laconte, "Medical Image Processing on the GPU – Past, Present and Future" in Medical Image Analysis, Elsevier Vol. 17, 2013, pp. 1073 – 1093.
[20] M. Almazrooie, M. Vadiveloo, R. Abdullah, "GPU – Based Fuzzy C – Means Clustering Algorithm for Image Segmentation" arXiv:1601.00072v3, 28 Mar 2016.
[21] N. Aitali, A. E. Abbassi, B. Cherraddi, O Bouattane, M. Youssfi, " Parallel Implementation of Bias Field Correction Fuzzy C – Means Algorithm for Image Segmentation " in International Journal of Advanced Computer Science and Applications, vol. 7(3), 2016, pp. 375 – 383.
[22] S. Saxena, N. Sharma, S. Sharma, "GPU constructed image segmentation using first order edge detection operators in CUDA environment" in Journal of Chemical and Pharmaceutical Research, vol. 8(2), 2016, pp. 379 – 387.
[23] H. M. Patel, K. Panchal, P. Chauhan, M.B. Potdar, "Satellite Image Processing using CUDA and Haddop Architecture" in International Journal of Scientific & Engineering Research , vol. 7(5), 2016, pp. 329 – 336.
[24] M. Wadpalliwar, M. Bhutani, M. M. Deshpande, "Implementation of an image filtering technique for image processing using CUDA" in Proceedings of 20th IRF International Conference, 1st March 2015, Chennai, India, ISBN: 978-93-84209-01-8.
[25] S.S. Stone et al., "Accelerating Advanced MRI Reconstructions on GPUs" in Journal of Parallel and Distributed Computing, 2008.
[26] A. Eklund, P. Dufort, D. Forsberg, S. M. LaConte, "Medical image processing on the GPU—past, present and future, Medical Image Analysis", vol. 17(8), 2013, pp. 1073–1094.
[27] S. Maulik, W. Boonn, "The Role of GPU computing in Medical Image Analysis and Visualization", SPIE 7967, Medical Imaging 2011: Advanced PACS-based Imaging Informatics and Therapeutic Applications, 79670L, 2011 doi: 10.1117/12.880093