ISSN 2277-3061
Volume 15 Number 14
INTERNATIONAL JOURNAL OF COMPUTERS & TECHNOLOGY

REGION GROWING IMAGE SEGMENTATION ON LARGE DATASETS USING GPU

Gurpreet Kaur [(1)], Sonika Jindal [(2)]
[(1)] Research Scholar, Department of Computer Science Engineering, SBSSTC, Ferozepur
Gurpreet878@gmail.com
[(2)] Assistant Professor, Department of Computer Science Engineering, SBSSTC, Ferozepur
sonikamanoj@gmail.com

ABSTRACT

Image segmentation is an important image processing, and it seems everywhere if we want to analyze what inside the image. There are varieties of applications of image segmentation such as the field of filtering noise from image, medical imaging, and locating objects in satellite images and in automatic traffic control systems, machine vision in problem of feature extraction and in recognition. This paper focuses on accelerating the image segmentation mechanism using region growing algorithm inside GPU (Graphical Processing Unit). In region growing algorithm, an initial set of small areas are iteratively merged according to similarity constraints. We have started by choosing an arbitrary seed pixel and compare it with neighboring pixels. Region is grown from the seed pixel by adding in neighboring pixels that are similar, increasing the size of the region. When the growth of one region stops we simply choose another seed pixel which does not yet belong to any region and start again. This whole process is continued until all pixels belong to some region. If any of the segment makers has the fusion cost lower than the maximum fusion cost (a given threshold), it is selected to grow. Avoid information overlapping like two threads attempting to merge its segment with the same adjacent segment. Experiments have demonstrated that the proposed shape features do not imply in a significant change of the segmentation results, as long as the algorithm’s parameters are properly adjusted. Moreover, experiments for performance evaluation indicated the potential of using GPUs to accelerate this kind of application. For a simple hardware (GeForce 630M GT), the parallel algorithm reached a maximum speed up of approximately 20-30% for different datasets. Considering that segmentation is responsible for a significant portion of the execution time in many image analysis applications, especially in object-oriented analysis of remote sensing images, the experimentally observed acceleration values are significant. Two variants of PBF (Parallel Best Fitting) and PLMBF (Parallel Local Mutual Best Fitting) have been used to analyze the best merging cost of the two segments. It has been found that the PLMBF has been performed better than PBF. It should also be noted that these performance gains can be obtained with low investment in hardware, as GPUs with increasing processing power are currently available on the market at declining prices. The parallel computational scheme is well suited for cluster computing, leading to a good solution for segmenting very large data sets.

Keywords

Segmentation, GPU, Image Processing, OpenCV, Region Growing algorithm, CUDA.

INTRODUCTION

Image segmentation is an important technology for image processing. There are many applications whether on synthesis of the objects or computer graphic images require precise segmentation. With the consideration of the characteristics of each object composing images in MPEG4, object-based segmentation cannot be ignored. Nowadays, sports programs are among the most popular programs, and there is no doubt that viewers’ interest is concentrated on the athletes. Therefore, demand for image segmentation of sport scenes is very high in terms of both visual compression and image handling using extracted athletes. In this project, we introduce a basic idea about color information and edge extraction to achieve the image segmentation. The color information helps obtain the texture information of the target image while the edge extraction detects the boundary of the target image. By combining these, the target image can be correctly segmented and represent. Besides, because color information and edge extraction can use basic image processing methods, they can not only demonstrate what textbook claims but also make us realize their function works. We expect that we can extract most part of the target. Segmentation is an important image processing technique and the main purpose is to separate target objects or regions from backgrounds. This can be achieved by clustering the input digital image into multiple salient areas [3]. Since segmentation can make an image easier to be processed further by higher level processing techniques, it is often used as pre-processing in many image analysis procedures [3]. Nowadays, image segmentation technique has been widely used in object detection and recognition, image editing, image compression, and image database search [4]. For example, segmentation is used as a pre-processing section of object recognition, i.e. face, iris, finger-print recognition etc., to locate or detect the target objects [3]. In addition, in traffic, meteorological, military and medical area, image segmentation is also becoming a vital technique [5]. For traffic image analysis, segmentation is normally used to locate and separate the target cars in the acquired images, so that some further work, such as car plate number recognition, can be continued [6]. In remote sensing area, segmentation technique plays a key part in satellite image processing, like city landform analysis and crop disease prediction. In the meteorological area, clouds location and analysis, and weather forecast are also need the help from segmentation technique [4]. In military field, it needs to segment targets to provide parameters for the automatic recognition systems. It can also help provide evidence for precise navigation and guidance of aircrafts [4]. In medical science fields, image segmentation techniques are widely used to segment organs, for instance, brain, heart, lungs etc., and also cells [5]. In particular, image segmentation can assist disease diagnosis by dividing different organs into separate regions. By extending the image segmentation techniques to three dimensions, it can be used for organ reconstruction [7]. We all know that every image is a set of pixels. And partitioning those pixels on the basis of the similar characteristics they have is called segmentation dividing an image into sub partitions on the basis of some similar characteristics like color, intensity and texture is called image segmentation. The goal of segmentation is to change the representation of an image into something more meaningful and easier to analyze. Image segmentation is normally used to locate objects and boundaries that is lines, curves, etc. in images. In image segmentation image is divided into some regions and in that region each pixel is similar with respect to some of the characteristic such as the color, intensity, or texture. Adjacent regions are significantly different with respect to the same characteristic(s). Image segmentation is to segment the image into multiple parts. It is useful everywhere whenever we want to analyze what is inside an image. For example, if we want to find if there is a chair or dog inside an indoor image, we need image segmentation technique to separate the objects in the image and analyze each object individually to check what it is…as we already know that because of image segmentation we can identify the diseases in medical imaging and also in many applications like face detection, iris detection, fingerprint recognition and also in brake light detection technique also we used this image segmentation technique.

GPU ARCHITECTURE

The GPUs are highly parallel, multithreaded programmable devices capable of rendering real-time graphics applications. It has tens, hundreds or sometimes even thousands of cores with tremendous power capable of high precision floating point arithmetic which is the requirement of real time video processing [3]. Graphics cards have a very high memory bandwidth of at least 64 bits in the modern processors which allows large amounts of data transfers in a single flow and a large number of on chip registers which allows it to hold several variable values while computation. Compared to the number of floating point operations per second (FLOPS) for a high-end CPU, FLOPS for GPU are exponentially higher. A standard GPU has a computation speed of few hundred gigaflops while for a high-end CPU its few tens of gigaflops. High end GPUs have computation speeds in few teraflops. The reasoning for such high computation speed is that GPUs are proficient in performing compute intensive highly parallel tasks such as graphics rendering. It is fashioned in a manner such that large number of transistors are dedicated to data processing instead of flow control and data caching.

GPU MEMORY ORGANIZATION

GPUs manufactured by NVidia have various memory spaces usable with its own advantages and constraints. Understanding the memory hierarchy and effectively utilizing it can result in better performance for GPU based applications. Device registers are the fastest memory available, which is accessible without latency on each clock cycle, just as in CPU. Each streaming multiprocessor (SM) has 16KB of registers. Each thread residing on GPU has its own register. These registers cannot be shared among threads. Also, all the registers are dispensed among all the threads that reside concurrently on the GPU. Therefore, if CUDA kernels utilize oodles of registers, the device will not be able to complete as many threads simultaneously. Each multiprocessor has shared memory space which is a 16KB region with very short access times. The purpose of shared memory is to act as a means for fast communication between threads. However, due to its speed, it can also be employed as a programmer controlled memory cache. After this, GPUs have DRAM (Dynamic Random Access Memory), or device memory, accessible at relatively 150x latency in comparison to registers or shared memory.

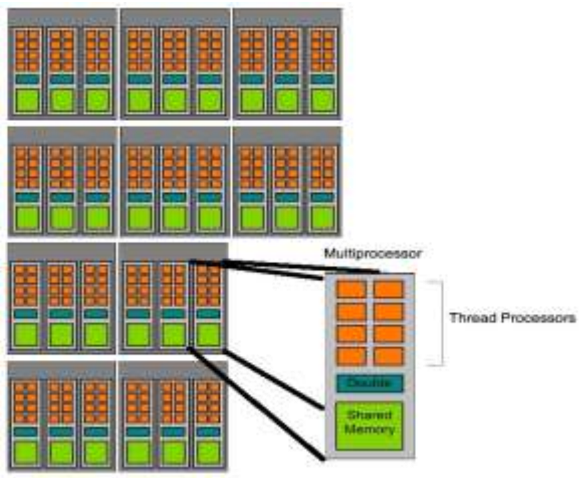

Figure 1. Diagram of multiprocessors in GPU.

CUDA

NVidia has released a complete software development toolkit for development of software for their GPUs, CUDA (Compute Unified Device Architecture). This took it includes a GPU compiler (nvcc) and a kernel profiler. CUDA is only adaptable with NVidia devices. It is a parallel computing platform and programming model which employs the Parallel Compute Engine to handle the threads running on the GPU. It is the job of the Parallel Compute Engine to lineup the threads to be executed on the GPU and map them to the vacant cores such that it complies with the CUDA architecture and specification. The CUDA platform provides a bedding for expansion to the C programming language alongside petty additions and limitation. To control hardware specific to the GPU, some additional language constructs are added [40]. Some constraints such as a lack of function pointers and recursion exist, because all functions are inclined by the compiler automatically. CUDA provides support for high level languages like C, C++, FORTRAN, Direct Compute and OpenACC, while third party wrappers are available for other languages like Python and Java [40]. CUDA C allows to define C functions called kernels, which are executed in parallel by N different CUDA threads. A CUDA kernel is defined as: global void kernel function name (parameter list) global declaration specifier indicates that code is executed on device and is invoked from host code. A thread is the basic unit of a CUDA program and it serves as the main component of a CUDA program. GPU can handle thousands on concurrent threads, CUDA allows a kernel launch to specify more threads than the GPU can execute concurrently which helps to amortize kernel launch times. Threads are grouped into blocks which in turn are grouped into a grid. It is in the hand of the programmer to specify the number of blocks to be created for execution of the parallel application. The programmer also defines the maximum number of threads that can be created per block.

$$Number\ of\ Blocks = \frac{Total\ number\ of\ threads}{Number\ of\ threads\ per\ block}$$

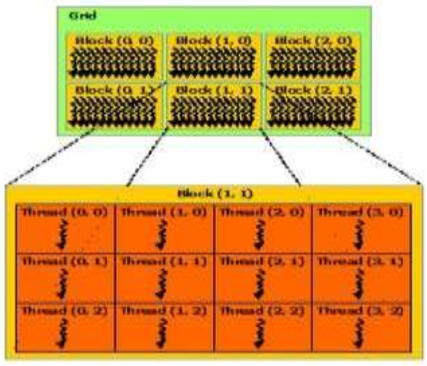

Figure 2. A thread of blocks

STRENGTHS OF GPU PLATFORM

Performance. The most apparent benefit of GPU computing is the usable performance at low cost. Some NVidia GPU cards provides 480 CUDA cores and can accomplish a peak GFLOP rate of 1788.48 for single precision. The fastest desktop CPUs have a theoretical peak rate of around 70 GFLOPs. Inexpensive and Convenient. Compared to customary supercomputing plat- forms, such as clusters, GPU computing has several noteworthy advantages. GPUs are available at low costs compared to desktop systems with CPU cores that would implement calculations at an equivalent rate, presuming that communication costs could be kept unknown. Further, a GPU system does not need a specialized server room with additional energy and maintenance costs.

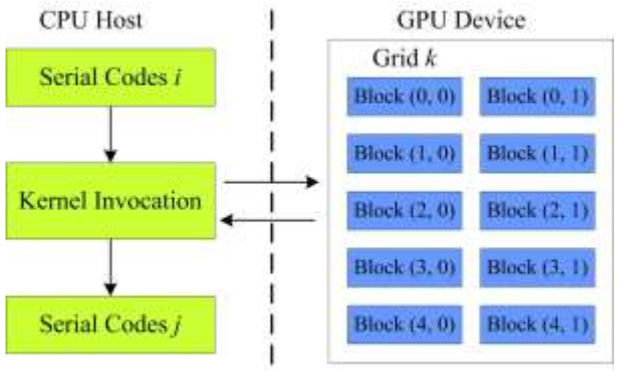

Figure 3. Kernel Execution

LITERATURE SURVEY

P. N. Happ et al. (2012) has stated that image segmentation is a computationally expensive task that continuously presents performance challenges due to the increasing volume of available high resolution remote sensing images.

Nowadays, Graphics Processing Units (GPUs) are emerging as an attractive computing platform for general purpose computations due to their extremely high floating-point processing performance and their comparatively low cost.

Anna Fabijanska et al. (2013) considers the problem of graph based image segmentation. In particular, attention is paid to minimal spanning tree based algorithm proposed by Felzenszwalb and Huttenlocher (FH). Although the method yields high quality results for various classes of images, its application is limited mainly to off-line processing.

Yong Li et al. (2013) has discussed that saliency detection for images and videos has become increasingly popular due to its wide applicability. In this research work, they have presented a new method that takes advantage of region-based visual dynamic contrast to generate temporally coherent video saliency maps. The concept of visual dynamics is formulated to represent both visual and motional variability of video content.

Patrick Nigri Happ et al. (2013) proposes a parallel version for graphics processing units (GPU) of a region-growing image segmentation algorithm widely used by the geographic object-based image analysis (GEOBIA) community. Initially, all image pixels are considered as seeds or primitive segments.

Abdenour amamra et al. (2014) presents a novel approach for background/foreground segmentation of RGBD data with the Gaussian Mixture Models (GMM). They first started by the background subtraction from the color and depth images separately.

Hyungsuk Choi et al. (2014) stated that as the size of image data from microscopes and telescopes increases, the need for high-throughput processing and visualization of large volumetric data has become more pressing. At the same time, many-core processors and GPU accelerators are common place, making high-performance distributed heterogeneous computing systems affordable.

Martin Rajchl et al. (2014) proposes a novel multi-region image segmentation approach to extract myocardial scar tissue from 3-D whole-heart cardiac late-enhancement magnetic resonance images in an interactive manner. For this purpose, they developed a graphical user interface to initialize a fast max-flow-based segmentation algorithm and segment scar accurately with progressive interaction.

Zhaohua Yi et al. (2015) focused on the hand detection and stated that it plays an important role in Human Computer Interaction (HCI). Most of the existing hand detection methods rely on the contour shape of hand after skin color segmentation. Such contour shape based approaches, however, are easily distorted by noises and other skin color segments.

Anju Soosan Baby et al. (2015) stated that image Segmentations play a heavy role in areas such as computer vision and image processing due to its broad usage and immense applications. Because of the large importance of image segmentation, a number of algorithms have been proposed and different approaches have been adopted. In this theme, I tried to parallelize the image segmentation using a region growing algorithm.

Choongsang Cho et al. (2015) has discussed that image smoothing has been used for image segmentation, image reconstruction, object classification, and three-dimensional content generation. Several smoothing approaches have been used at the pre-processing step to retain the critical edge, while removing noise and small details. However, they have limited performance, especially in removing small details and smoothing discrete regions.

Kwang Yeob Lee et al. (2015) has considered many uses of GPU architecture. Mobile devices provide a more realistic image processing and various high spec features to satisfy users. So, mobile devices have been developing in the direction of computing acceleration by using a strong parallelism of GP-GPU processing. In this research work, a GP-GPU architecture is proposed based on stream processing architecture which has the advantage of high parallelism to enhance the image processing capability.

Madison Gray McGaffin et al. (2015) Image de-noising is a fundamental operation in image processing, and its applications range from the direct (photographic enhancement) to the technical (as a sub problem in image reconstruction algorithms). In many applications, the number of pixels has continued to grow, while the serial execution speed of computational hardware has begun to stall.

RESEARCH GAP

The memory constrictions: The CUDA platform has two memory constrictions. Firstly, chokepoint of communication between the device GPU and the host CPU. Transfer of data to and from GPU poses great difficulties. Similarly, while functioning on accelerated device registers, GPU behaves optimum, compared to while working with device memory. GPU conducts at extreme speed when it executes many operations for each memory access to hide this latency. Applications requiring random access to very large datasets are usually memory bound and are unable to attain near the peak capabilities of a GPU.

Programming Difficulty: In spite of the fact that GPU programming has progressed substantially well and GPU chipset developers made considerable progress in designing tools for producing GPU accelerated software promptly accessible, GPU programming is yet not an effortless task that is undoubtedly suitable for a novice programmer. Competent GPU programming demands deep understanding of a modern memory hierarchy and programming model with new terminology and concepts.

Precision: The prevailing generations of GPUs are mainly used with single precision. Although the current GPU generation supports double precision, there is a considerable performance penalty.

PROBLEM FORMULATION

• Slow processing of images by GPU when the CPU invokes the GPU.

• Lot of swapping time is wasted when the CPU calls the GPU and GPU returns the result for the single image.

• GPU is activated only for a smaller duration of time whereas the time is wasted between the GPU call and returning to the host.

• For a single image, one thread is assigned by the GPU that will process all the pixels of an image.

OBJECTIVES

• To study the working of existing segmentation algorithms.

• To propose a robust region growing algorithm in GPU that segments the whole dataset in a single cycle.

• To create the multiple blocks and threads in GPU separately for each image in the dataset and for each pixel in the image.

• To implement the proposed algorithm in Open CV environment and compare the results of the proposed algorithm with the existing work.

REGION GROWING SEGMENTATION

Image segmentation is useful in many applications. It can identify the regions of interest in a scene or annotate the data. We categorize the existing segmentation algorithm into region-based segmentation, data clustering, and edge-base segmentation. Region-based segmentation includes the seeded and unseeded region growing algorithms, combined with two variants of the aforesaid algorithm: PBF (Parallel Best Fitting), based on the “best fitting” segmentation; and PLMBF (Parallel Local Mutual Best Fitting), derived from the “local mutual best fitting” heuristic. With few small differences, both versions are performed by the, and the fast scanning algorithm. All of them expand each region pixel by pixel based on their pixel value or quantized value so that each cluster has high positional relation. For data clustering, the concept of them is based on the whole image and considers the distance between each data. The characteristic of data clustering is that each pixel of a cluster does not certainly connective. The basis method of data clustering can be divided into hierarchical and partition clustering. Furthermore, we show the extension of data clustering called mean shift algorithm, although this algorithm much belonging to density estimation. The last classification of segmental ton is edge-based segmentation. This type of the segmentations generally applies edge detection or the concept of edge. The typical one is the watershed algorithm, but it always has the over-segmentation problem, so that the use of markers was proposed to improve the watershed algorithm by smoothing and selecting markers. Finally, we show some applications applying segmentation technique in the preprocessing.

The seeded region growing (SRG) algorithm is one of the simplest region-based segmentation methods. It performs a segmentation of an image with examine the neighboring pixels of a set of points, known as seed points, and determine whether the pixels could be classified to the cluster of seed point or not. The algorithm procedure is as follows.

Step 1. We start with a number of seed points which have been clustered into n clusters, called C1, C2… Cn. And the positions of initial seed points is set as p1, p2… p3.

Step 2. To compute the difference of pixel value of the initial seed point pi and its neighboring points, if the difference is smaller than the threshold we define, the neighboring point could be classified into Ci, where i = 1, 2, …,n.

Step 3. Recomputed the boundary of Ci and set those boundary points as new seed points pi (s). In addition, the mean pixel values of Ci have to be recomputed, respectively.

Step 4. Repeat Step2 and 3 until all pixels in image have been allocated to a suitable cluster.

The threshold is made by user and it usually based on intensity, gray level, or color values. The regions are chosen to be as uniform as possible. There is no doubt that each of the segmentation regions of SRG has high color similarity and no fragmentary problem. However, it still has two drawbacks, initial seed-points and time-consuming problems. The initial seed-points problem means the different sets of initial seed points cause different segmentation results. This problem reduces the stability of segmentation results from the same image. Furthermore, how many seed points should be initially decided is an important issue because various images have individually suitable segmentation number. The other problem is time-consuming because SRG requires lots of computation time, and it is the most serious problem of SRG.

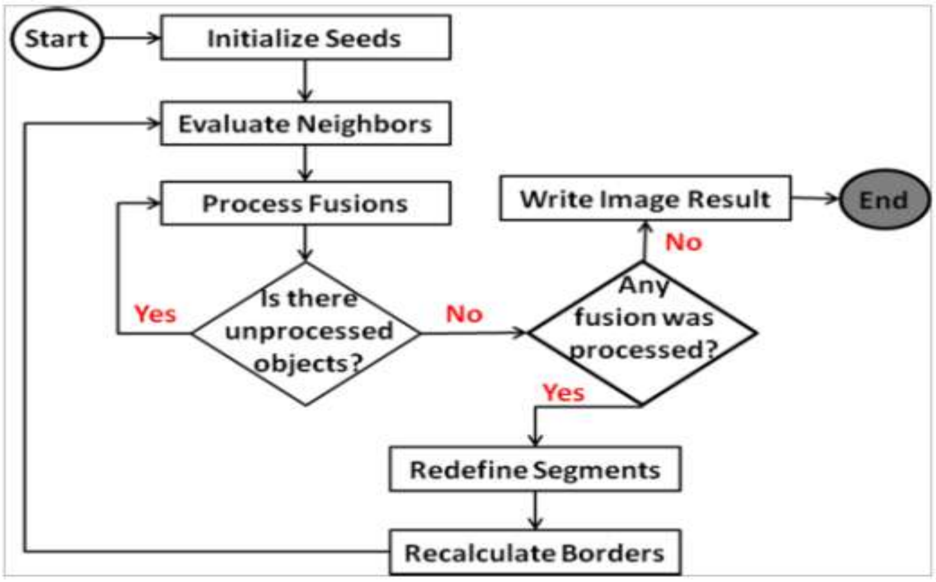

Figure 4. Region Growing Segmentation

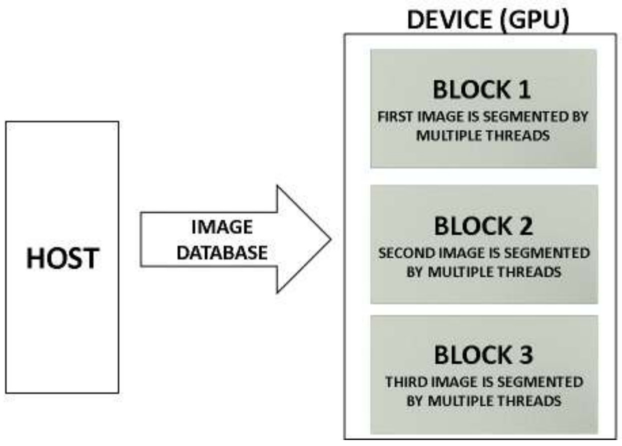
DEVICE (GPU)
BLOCK 1
FIRST IMAGE IS SEGMENTED BY MULTIPLE THREADS
HOST
IMAGE DATABASE
BLOCK 2
SECOND IMAGE IS SEGMENTED BY MULTIPLE THREADS
BLOCK 3
THIRD IMAGE IS SEGMENTED BY MULTIPLE THREADS

Figure 5. Image Segmentation using GPU Blocks

EXPERIMENTAL RESULTS

Different techniques are available to facilitate the performance of image segmentation such as pixel based, edge based, cluster based, region based, model based, color based and hybrid. This thesis focuses on accelerating the image segmentation mechanism using region growing algorithm inside GPU (Graphical Processing Unit). In region growing algorithm, an initial set of small areas are iteratively merged according to similarity constraints. We have started by choosing an arbitrary seed pixel and compare it with neighboring pixels. Region is grown from the seed pixel by adding in neighboring pixels that are similar, increasing the size of the region. When the growth of one region stops we simply choose another seed pixel which does not yet belong to any region and start again. This whole process is continued until all pixels belong to some region. If any of the segment makers has the fusion cost lower than the maximum fusion cost (a given threshold), it is selected to grow. Avoid information overlapping like two threads attempting to merge its segment with the same adjacent segment. Although this approach is flexible, and can be used with variety of datasets having different number of images of different dimensions. NVIDIA's CUDA framework for general purpose computation on GPUs is used in conjunction with NVIDIA GPUs to reduce processing time by creating multiple blocks and threads inside the kernel. Multiple experiments have been conducted on multiple number of sample datasets. Experiments have demonstrated that the proposed shape features do not imply in a significant change of the segmentation results, as long as the algorithm's parameters are properly adjusted. Moreover, experiments for performance evaluation indicated the potential of using GPUs to accelerate this kind of application. For a simple hardware (GeForce 630M GT), the parallel algorithm reached a maximum speed up of approximately 20-30% for different datasets. Considering that segmentation is responsible for a significant portion of the execution time in many image analysis applications, especially in object-oriented analysis of remote sensing images, the experimentally observed acceleration values are significant. The details of the host are specified in table1.

Table 1. Details of Host

HOST DETAILS	
PROCESSOR	i5 2450M 2.50 Ghz
RAM	8 GB DDR3
OS	WINDOWS 7 64 BIT
IDE	VISUAL STUDIO 2015
PROGRAMMING LANGUAGE	VISUAL C++ with OpenCV for Image Processing

In the Figure 6,7,8 we have showed the segmentation results for the different set of images like arms, legs etc. In the first row, we have showed the original image, in the second row the binary image is showcased and finally in the third row, the actual image after segmentation through region growing algorithm is shown.

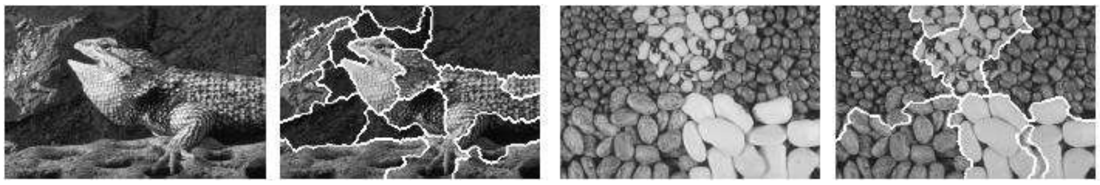

Figure 6. Segmentation of gray level images

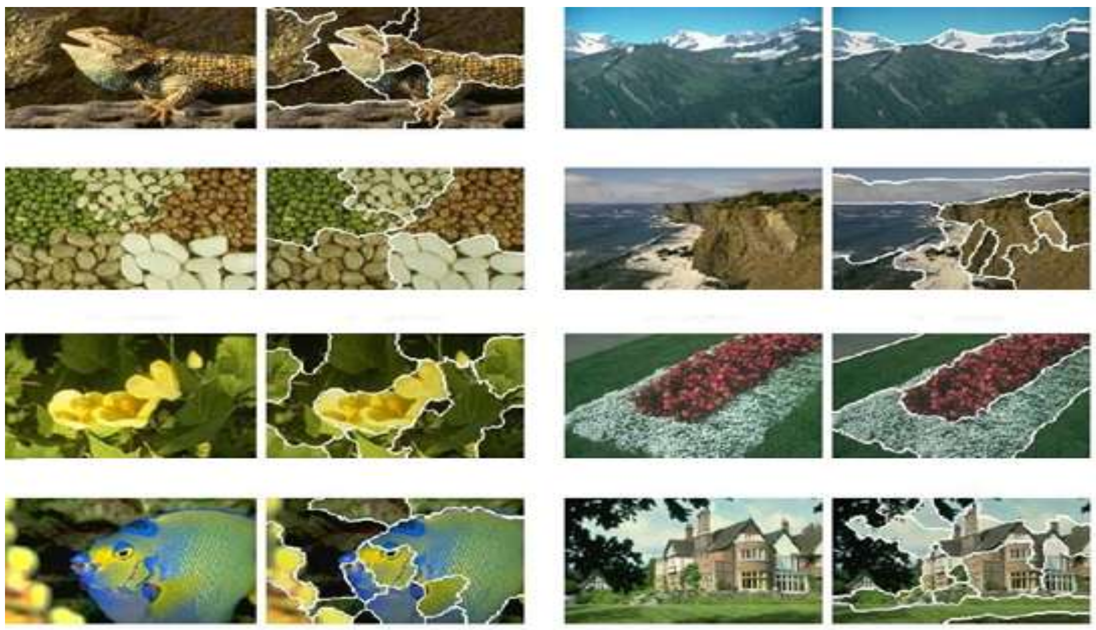

Figure 7. Segmentation results of some examples (one scale, size 192 *128 pixels).

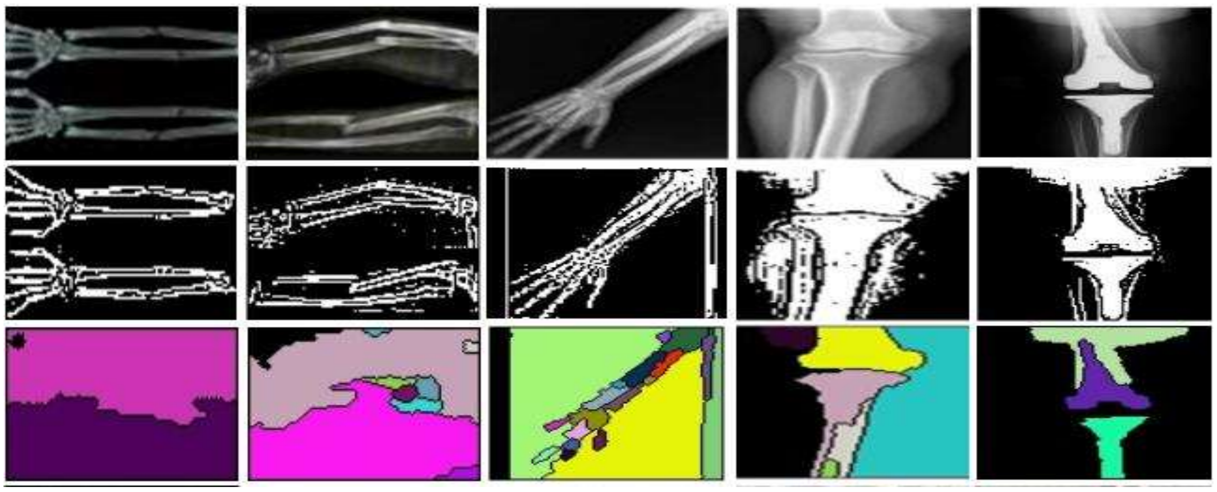

Figure 8. Segmentation on Medical Images

In the figure 9, different categories of images like boats, parking lot, butterflies, cars and residential plots are taken and are applied on the same algorithm for further analysis.

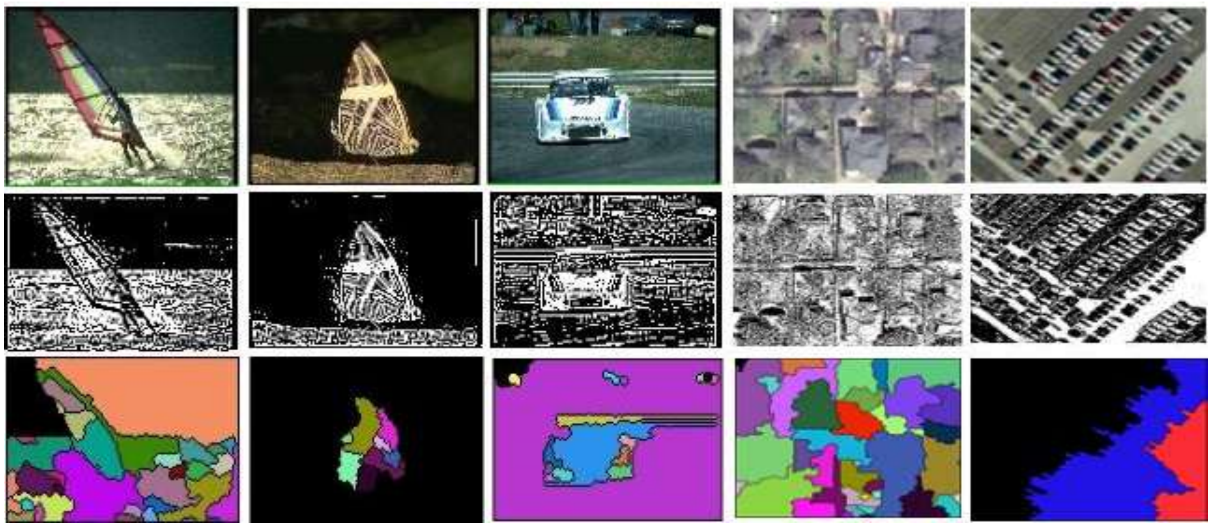

Figure 9. Segmentation on Different Classes of Images

In PLMBF, we remove small clusters. The assignment is according to the smallest differences between the pixel and its mean of adjacent clusters. We conclude the advantages and disadvantages of the PLMBF algorithm:

Advantages:

1. The pixels of each cluster are connected and have similar pixel value, i.e. it has good shape connectivity.
2. The computation time is faster than both region growing algorithm and region splitting and merging algorithm.
3. The segmentation results exactly match the shape of real objects, i.e. it has well shape matching.

Multiple number of experiments have been conducted on different categories of images of different sizes. These experiments are firstly conducted on the host machine and afterwards the experiments are conducted on the device. The results of the Host machine are specified in the table 2.

Table 2. Region Growing Algorithm on Host

S.No.	Category	No of Images	Image Dimensions	Image Size	Processing Time (in milliseconds)
1	Animals	68	128 x 96	274 KB	1652
2	Boats	68	128 x 85	285 KB	1354
3	Butterflies	76	129 x 85	293 KB	971
4	Cars	86	130 x 85	342 KB	973
5	Flowers	83	128 x 85	265 KB	809

6	Food	66	128 x 96	257 KB	1160
7	Greenery	54	128 x 85	224 KB	878
8	Mountains	71	128 x 96	270 KB	1045
9	People	65	128 x 96	270 KB	1239

In the table 3, we have specified the details of the GPU used for the experimental purposes.

Table 3. GPU (Device) Details

DEVICE DETAILS	
CARD NAME	**GeForce GT 630M 2GB**
Number of Cores	96
Processor Clock	800 MHz
Memory Interface	128-bit DDR3

The experiments are conducted on the above specified device (GPU) in Table 4. for the same set of images that are used in the host machine.

Table 4. Region Growing Algorithm on Device

S.No.	Category	No of Images	Image Dimensions in pixels	Image Size	Processing Time (in milliseconds)
1	Animals	68	128 x 96	274 KB	954
2	Boats	68	128 x 85	285 KB	856
3	Butterflies	76	129 x 85	293 KB	696
4	Cars	86	130 x 85	342 KB	745
5	Flowers	83	128 x 85	265 KB	506
6	Food	66	128 x 96	257 KB	763
7	Greenery	54	128 x 85	224 KB	633
8	Mountains	71	128 x 96	270 KB	566
9	People	65	128 x 96	270 KB	709

In the table 4, we have computed the parallel processing time in GPU for the same set of images that were used in the host machine. It is clear from the results that there is reduction of the processing time by approximately 40% and thereby increasing the overall efficiency of the system. Moreover multiple images of different resolutions like 64 * 64, 128 * 128, 256 * 256 etc. have been used for further analysis. The processing time of the experiments have been mentioned in the table 5.

Table 5. Images of Multiple Dimensions

Dim\Time	CUDA	CPU
64x64	401	748
128x128	572	976
256x256	1247	1778
512x512	2569	3640
768x768	3698	5260
1024x1024	5264	7656
2048x2048	12476	16985
4096x4096	18932	35421

Table 5 reveals that as the dimensions of the image are increasing, the gap between the CPU (host) processing time and the device processing time is also increasing.

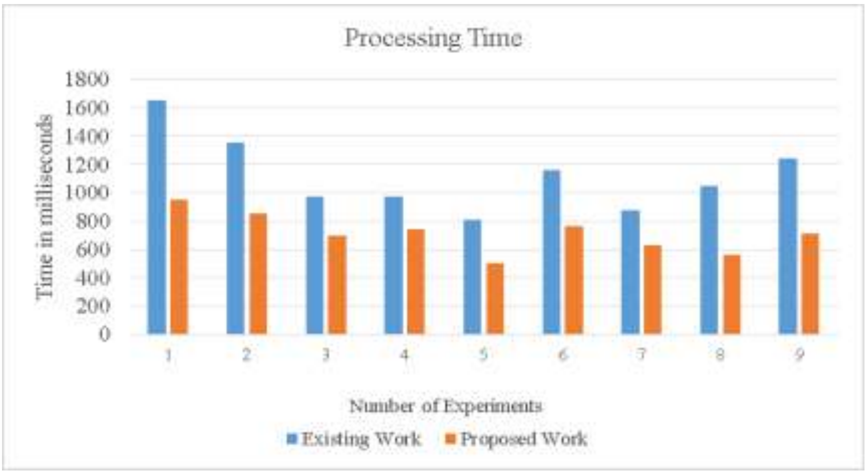

Figure 10. Processing Time

Figure 10. reveals the bar chart comparison of overall processing time for the existing work and the proposed work. In the existing work, one by one images are processed into the GPU, thereby taking lot of swapping time and overhead. But in the proposed work, full dataset is processed in a single cycle, thereby reducing the overall swapping time. Figure 11. shows the segmentation time for the different set of images of different resolutions.

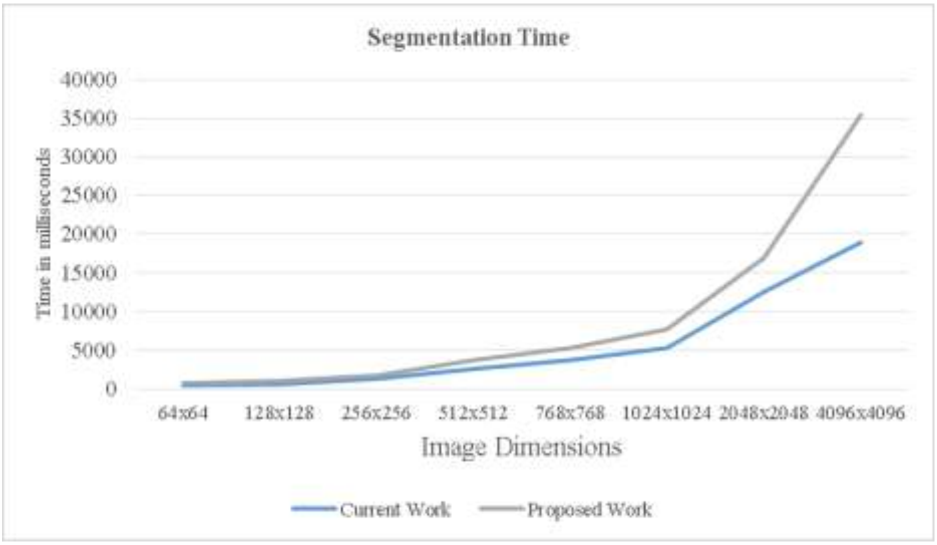

Figure 11. Overall Segmentation Time for Different Resolutions

In the figure 11, the segmentation times are compared for the different resolutions of the images. As the resolution increases, the gap between the existing work and the proposed work also increases, thereby increasing the overall efficiency of the system. The general region-growing procedure is to compare one pixel to its neighbors. If a similarity criterion is satisfied, the pixel is assumed to belong to the same region. The algorithm starts with a seed point selected manually inside the region of interest and begins to compare the intensity of this point to that of its neighbors.

CONCLUSION AND FUTURE SCOPE

Segmentation is an inherently subjective problem and quantitatively measuring performance of different segmentation algorithms is extremely tricky since there is no real "correct" answer to be compared with. Thus, the user should be able to parametrically control the segmentation that is achieved and this is provided for in the parameters of the weight function in all the graph theoretic formulations. The edge-based segmentation method, especially watershed, has the over-segmentation problem, so we normally combine the marker tool and watershed for overcoming the over-segmentation problem. The region growing can be seen to give quite good results for image segmentation. Additionally, by combining the PLMBF and PBF, the number of groups segmented can also be controlled. When comparing the other formulations like average association and average cut with PLMBF, it was seen, both theoretically and experimentally that PLMBF gives better results. Average association has a bias for finding tight clusters and it runs the risk of finding small, tight clusters in the data even though dissimilarities with other clumps should actually result in less number of clusters. On the other hand, average seed does not look at within-group similarity and this causes problems when the dissimilarity between groups is not very pronounced. Thus, region growing combined with GPU produces better results in practice because it takes into account both similarity within groups as well as dissimilarity across groups. Here we used the region growing technique to segment the image. Here, we overcome the correlation property of the image so, we got a better segmented image. It is very difficult to predetermine the right weighting function on each image region. In this paper we have proposed a parallel region growing segmentation in gpu for the whole dataset. We have used the multiple GPU blocks and threads for all the images and multiple sub threads for the corresponding pixels of the image of this algorithm. We have reached upto the conclusion that there is enhancement in the processing speed by 19 times and the segmentation results are almost similar in both the sequential and parallel cases. Future scope, we can go for the multiple experiments using different GPU's like GeForce 900, 800, 700 series. Therefore, it is important to design a grouping algorithm that is more tolerant to a wide range of weighting functions. Region-based methods are derived from the suggestion that neighboring pixels within the same region have similar intensity values and involve one of two different procedures: region growing, or split and merge. In the future scope, we can combine the region growing with other algorithms like N-cut method of pixel based technique to reduce the complexity and future reduce the segmentation time. Region based Ncut requires lower computational complexity compared to pixel based Normalized cut. And also, it gives a better segmented image when compared to the pixel based normalized cut method.

REFERENCES

[1] A. . K. Sahoo, G. Kumar, G. Mishra and R. Misra, "A New Approach for Parallel Region Growing Algorithm in Image Segmentation using MATLAB on GPU Architecture," *IEEE ,* pp. 279-283, 2015.

[2] J. X. Hua and M.-H. Jeong, "Real-Time Range Image Segmentation on GPU," *International Conference on Control, Automation and Systems (ICCAS),* pp. 150-153, 2014.

[3] S. Katsigiannis, E. Zacharia and D. Maroulis, "MIGS-GPU: Microarray Image Gridding and Segmentation on the GPU," *IEEE,* pp. 1-8, 2015.

[4] H. Cho, S.-J. Kang, S. I. Cho and Y. H. Kim, "Image Segmentation Using Linked Mean-Shift Vectors and Its Implementation on GPU," *IEEE,* pp. 719-727, 2014.

[5] Darian M. Onchis, D. Frunzaverde, M. Gaianu and R. Ciubotariu, "Multi-phase identification in microstructures images using a GPU accelerated fuzzy c-means segmentation," *IEEE,* pp. 602-607, 2015.

[6] N. Aitali, B. Cherrad, O. Bouattane, M. Youssfi and A. Raihani, "New Fine-Grained Clustering Algorithm on GPU Architecture for Bias Field Correction and MRI Image Segmentation," *IEEE,* pp. 118-121, 2015.

[7] C. S. Cho and S. Lee, "Low-Complexity Topological Derivative-Based Segmentation," *IEEE,* pp. 734-741, 2015.

[8] J. Chalfoun, M. Majurski, T. Blattner, W. Keyrouzl, P. Bajcsy and M. Brady, "MIST: Microscopy Image Stitching Tool," *IEEE,* p. 1757, 2015.

[9] Arthur D. Costea and Sergiu Nedevschi , "Multi-Class Segmentation for Traffic Scenarios at Over 50 FPS," *IEEE,* pp. 1390-1395, 2014.

[10] X. Zhang, G. Tan and M. Chen, "A Reliable Distributed Convolutional Neural Network for Biology Image Segmentation," *IEEE,* pp. 777-780, 2015.

[11] Mohammed A. Shehab, , Mahmoud Al-Ayyoub and Yaser Jararweh, "Improving FCM and T2FCM Algorithms Performance using GPUs for Medical Images Segmentation," *IEEE,* pp. 130-135, 2015.

[12] Z. Liu, X. Li, P. Luo , C. C. Loy and X. Tang, "Semantic Image Segmentation via Deep Parsing Network," *IEEE,* pp. 1377-1785, 2015.

[13] Z. Yi, X. Hu, B. Jang and K. K. Kim, "A Robust and Parallel-Friendly Distance Image Based Hand Detection," *IEEE,* pp. 33-34, 2015.

[14] J. Sirotkovic, H. Dujmic and V. Papic , "Image segmentation based on complexity mining and mean-shift algorithm," *IEEE,* pp. 1-6, 2014.

[15] A.-R. Baek, K. Lee and H. Choi , "Speed-up Image Processing on Mobile CPU and GPU," *IEEE,* pp. 79-81, 2015.

[16] Z. Juhasz and G. Kozmann, "A GPU-based Simultaneous Real-Time EEG Processing and Visualization System for Brain Imaging Applications," *IEEE,* pp. 299-304, 2015.

[17] H. L. Khor, S.-C. Liew and Jasni Mohd. Zain , "A Review on Parallel Medical Image Processing on GPU," *IEEE,* pp. 45-48, 2015.

[18] A. Asaduzzaman, A. Martinez and A. Sepehri , "A Time-Efficient Image Processing Algorithm for Multicore/Manycore Parallel Computing," *IEEE,* pp. 1-5, 2015.

[19] A. Fabija and J. Gocławski, "New accelerated graph-based method of image segmentation applying minimum spanning tree," *IET Image Processing,* pp. 239-251, 2013.

[20] A. S. Baby and Balachandran K, "A Parallel Approach For Region-Growing Segmentation," *IEEE,* pp. 196-200, 2015.

[21] C. Cho and S. Lee, "Effective Five Directional Partial Derivatives-based Effective Five Directional Partial Derivatives-based," *IEEE,* pp. 1-9, 2015.

[22] K. Y. Lee, G. Kyung, T. R. Park, J. C. Kwak and Y. S. Koo, "A Design of a GP-GPU based Stream Processor for an Image Processing," *IEEE,* pp. 535-539, 2015.

[23] M. G. McGaffin and Jeffrey A. Fessler, "Edge-Preserving Image Denoising via Group Coordinate Descent on the GPU," *IEEE,* pp. 1273-1281, 2015.

[24] A. amamra, T. mouats and N. aouf, "GPU based GMM segmentation of Kinect data," *International Symposium ELMAR,* pp. 99-102, 2014.

[25] R. Q. F. B. R. F. P. N. Happ, "A PARALLEL IMAGE SEGMENTATION ALGORITHM ON GPUS," *Proceedings of the*

4th GEOBIA, pp. 580-585, 2012.

[26] H. Choi, W. Choi, T. M. Quan, , David G. C. Hildebrand, Hanspeter Pfister and Won-Ki Jeong, "Vivaldi: A Domain-Specific Language for Volume Processing and Visualization on Distributed Heterogeneous Systems," *IEEE* , pp. 2407-2416, 2014.

[27] Y. Li, B. Sheng, Lizhuang Ma, Wen Wu and Zhifeng Xie, "Temporally Coherent Video Saliency Using Regional Dynamic Contrast," *IEEE ,* pp. 2067-2076, 2013.

[28] M. Rajchl, J. Yuan, E. Ukwatta, James A. White, J. Stirrat, Cyrus M. S. Nambakhsh, Feng P. Li and Terry M. Peters, "Interactive Hierarchical-Flow Segmentation of Scar Tissue From Late-Enhancement Cardiac MR Images," *IEEE,* pp. 159-172, 2014.

[29] Patrick Nigri Happ, Raul Queiroz Feitosa, Cristiana Bentes and Ricardo Farias, "A Region-Growing Segmentation Algorithm for GPUs," *IEEE,* pp. 1612-166, 2013.

[30] X. G. W. Souleymane Balla-Arabé, "GPU Accelerated Edge-Region Based Level Set Evolution Constrained by 2D Gray-Scale Histogram," *IEEE,* pp. 2688-2698, 2013.