Image Segmentation Using Parallel Mean Shift Algorithm

Raj Kumar Sah[1], Md. Zahidul Islam[2]
[1]A P Goyal Shimla University, Shimla (H.P), India
[2]Khulna University, Khulna, Bangladesh
(E-mail: errazks@gmail.com, zahid@cse.ku.ac.bd)

Abstract - In automatic image interpretation, the Process of extracting different objects that composes an image is one of the primary steps. This process is known as image segmentation and consists of sub dividing an image into meaningful regions, also called segments, which has been classified. Many of the existing segmentation algorithms have high computational cost for large images due to the high-resolution of the images. Parallel programming is becoming popular to the practitioners for reducing computation time. Due to the availability of multicore processors, various tools and techniques are being developed by researchers and programming platform developers. The idea is to explore current multi-core commercial processors in order to speedup the segmentation process. In this thesis, a multi core parallel implementation is presented that aims at providing better execution time, while delivering a similar outcome produced by the sequential version. The presented framework is able to work with any number of cores and any number of images from the system to take full advantage of the upcoming processors having unseen number of cores. The current parallel implementation tested on five different images on multiple systems having different cores and speed.

Keyword*-Finding modes; image segmentation; image processing; visual tracking; and space analysis & object tracking*

I. INTRODUCTION

Mean shift algorithm was introduced by Fukunaga and Hostetler (in 1976) [1]. A finite mixture or the kernel density estimate often defines the algorithm. Mean shift has twofold use, it is often used as a nonparametric clustering method that aids finding methods, and in recent applications in computer vision it functions as image segmentation or object tracking [2]. Image segmentation is the process of partitioning a digital image into multiple segments also known as super pixels. The main purpose of image segmentation is to represent the image in a much meaningful manner to understandable by the computer. The process is typically used to locate objects in images. In other word, image segmentation is the process of assigning a label to every pixel in an image in such a manner that same label carries out certain characteristics. The Parallel Patterns Library (PPL) is a programming model that gives a great flexibility to developing parallel processed applications. It provides an interrelation between application code and the threading mechanism by providing algorithms and containers that act greatly on data parallelism [6]. The PPL also offers developing applications that provides application to shared state. The goal of this thesis is to develop a framework for image segmentation using parallel mean shift algorithm. Mean shift is a multi-used tool for feature space analysis that provides solution for many vision tasks. I have chosen mean shift over the other method, because mean shift procedure inherits an interesting property, its path towards the mode follows a smooth trajectory, the angle between two consecutive mean shift vectors being always less than 90 degrees [3] which provides a smooth result for image segmentation. However, it is observed that the full process of mean shift algorithm is very time consuming. To decrease the run time of the algorithm we have used the multicore programming system. The target of this study is to decrease the runtime of the algorithm using parallel programming. So the use of PPL makes the mean shift algorithm to run parallel on multiple processors. Implementing the algorithm using Microsoft's parallel pattern library (PPL) dynamically scales the degree of parallelism efficiently using all the processing cores that are available. In addition, PPL assists in the partitioning of work and the scheduling of tasks in threads. The library provides cancellation support, state management, and other services required for parallel programming. These libraries make use of the Concurrency Runtime, which is part of the Visual C++ platform [1], [4], [8].

II. MEAN SHIFT ALGORITHM USING PARALLEL PATTERN LIBRARY (PPL)

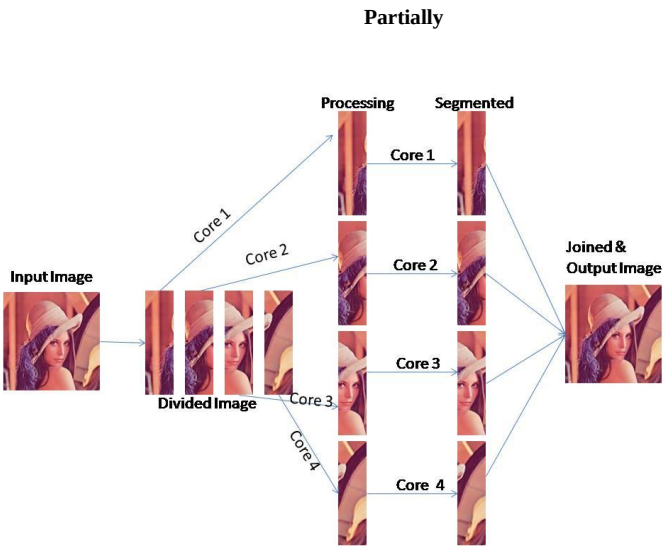


Figure 1: Parallel segmentation process using PPL.

Parallel high performance mean shift algorithm using PPL. This dynamic mean shift algorithm updates both the sample space and the "mean", which is a subspace of the sample space. The process divides the data and fastens the process. In general mean shift uses a window and repeat it until it finds the data till the last point. Used the algorithm with parallel processing that divides the data into multiple cores and takes the data parallel which increases the speed of the algorithm due to its degree of core number. For example: if CPU having 4 cores, it will take approximately ¼ times than the general process. The algorithm will work this way with an image in a 4 core system as in Figure 1.

A. Image Blending Test Using PPL

We did an experiment on image processing to see the performance of parallel processing, where images are created with layers. Separate images from different sources are processed independently and then combined with alpha blending, shown in figure 2.

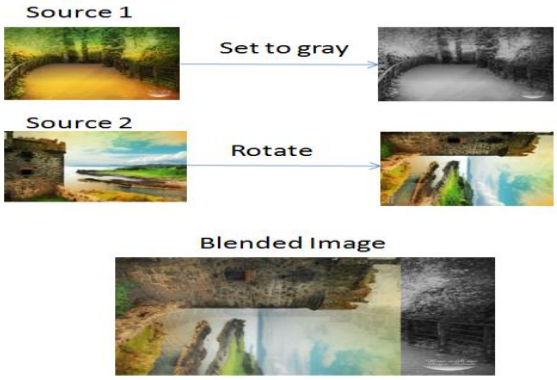


Figure 2: Image Blending.

This Process merges two semitransparent layers to create a new single image. In this process, two image sources named source1 and source2 are taken which are the original source images; layer1 and layer2 are two bitmaps that have been prepared with optional information to blend the images, blender is a Graphics instance to perform the blending. Internally, SettoGray, Rotate and Blend are the methods to perform the image processing. The SettoGray and Rotate methods are entirely independent of each other. If two or more cores are available, the tasks might run in parallel, and the image blending process might finish in low cost time than a sequential version would. The average timing is calculated by running the program in different configured computers.

Result on an Intel Pentium Dual CPU E2180 @ 2.00 GHZ

Sequential = 988.17
Task_Group = 580.85
Structure task_Group = 580.66
Parallel_invoke = 725.88

Result on an Intel Pentium Core 2 Duo @ 2.10GHZ
Sequential = 904.04
Task_Group = 583.18
Structure task_Group = 581.61
Parallel_invoke = 564.74

Result on an Intel Pentium Core i3 @ 2.53GHZ
Sequential = 563.12
Task_Group = 371.33
Structure task_Group = 366.55
Parallel_invoke = 376.46

Average time processing for different processor for set to gray, rotate and blends.

B. Parallel Processing

Parallel computation is the accompanying use of more than one CPU or processor cores to complete a task. In general, parallel processing makes programs run faster using multiple processing units [20], [21]. *Task parallelism* and *data parallelism* are two programming models for parallel programming. The task parallelism focuses on distributing threads beyond different parallel computation nodes in which different processor executes the same or different programs on same or different data. Different processors communicate with each other during execution. On the other hand, data parallelism is a programming technique for splitting a large data set into smaller chunks that can be operated in parallel. In this model, multiple CPUs execute same program with a part of the input data set. We adapted data parallelism in our proposed system. PPL is designed by native C++ developers and bundled with Microsoft Visual Studio providing features for multi core programming.
PPL builds on the scheduling and resource management components of the Concurrency Runtime. The features of PPL include: a mechanism to execute several work items (tasks) in parallel, generic algorithms that act on collections of data in parallel, generic containers providing safe concurrent access to their elements, and procedures to exploit data parallelism. PPL provides a convenient and readily usable toolkit that combines the simplicity of managed-language equivalents with the elegance and expressiveness of C++. PPL uses more abstract concepts than threads and fibers as the base unit of scheduling [22].

III. EXPERIMENTAL RESULT ANALYSIS AND IMPLEMENTATION TOOLS

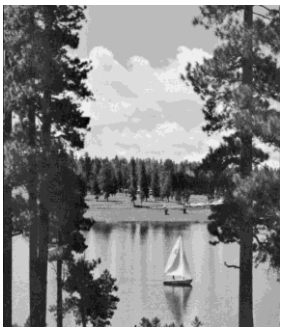
The overall performance of the system depends on the CPU's available core numbers. If the number of CPU core is higher, then the performance of the system will be improved. We have used Microsoft visual studio, OpenCV C++ and parallel pattern library to implement the proposed system. The visual studio of Microsoft provides them both. Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. Visual Studio uses Microsoft software development platforms such as Windows API, Windows Forms, Windows

Presentation Foundation, Windows Store and Microsoft Silverlight.

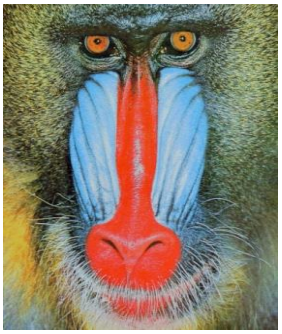
A. Test Environment

Five standard images with same sizes (512×512) are used. They are named as Cameraman, Lake, Mandrill, pepper and Leena. All images are used to evaluate the performance gains and we have compared the result of parallel segmentation to the result from sequential segmentation.

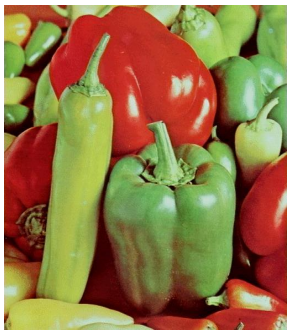
Cameraman



Lake



Mandrill

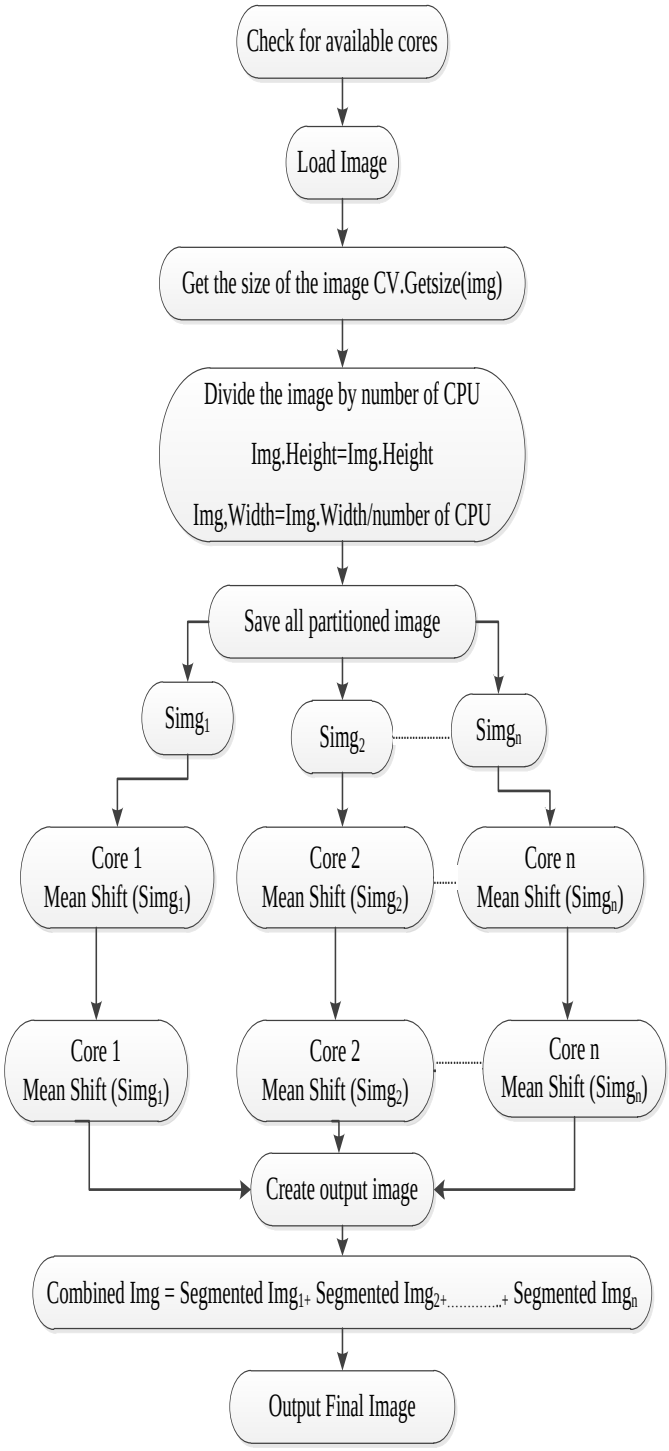


Pepper

LEENA

Figure 3: Test Environment Image Input

B. Architecture of the Proposed System



C. Results of Sequential and Parallel Segmentation

The performance of the proposed parallel algorithm has been evaluated using the same images. The results are given below for sequential segmentation and parallel segmentation. Note that execution time of the segmentation is reduced with the increase in the number of cores.

TABLE I: EXPERIMENTAL RESULT USING DUEL CORE PROCESSOR

The experiment result performed on an Intel Duel Core 2.60GHz, 2 GB of RAM.		
IMAGE NAME	**Sequential segmentation (MS)**	**Parallel segmentation (MS)**
Camera Man	13275.60	4746.59
Lake	10505.53	6617.28
Mandrill	10986.33	8439.29
Pepper	13769.10	6878.19
Leena	10807.80	6902.16

TABLE II: EXPERIMENTAL RESULT USING CORE 2DUO PROCESSOR

The experiment result performed on an Intel Core 2duo 2.93 GHz, 4 GB of RAM.		
IMAGE NAME	**Sequential segmentation (MS)**	**Parallel segmentation (MS)**
Camera Man	11192.87	4011.78
Lake	8377.275	5584.85
Mandrill	8579.32	7144.87
Pepper	11578.61	5455.41
Leena	8677.33	5436.07

TABLE III: EXPERIMENTAL RESULT USING CORE 2QUAD PROCESSOR

The experiment result performed on an Intel Core 2quad 2.3 GHz, 2 GB of RAM.		
IMAGE NAME	**Sequential segmentation (MS)**	**Parallel segmentation (MS)**
Camera Man	8534.7	3545.73
Lake	6096.2	4321.54
Mandrill	7860.3	5480.34
Pepper	10962.9	4367.73
Leena	7483.1	4323.73

TABLE IV: EXPERIMENTAL RESULT USING CORE I3 PROCESSOR

The experiment result performed on an Intel Core i3 3.1 GHz, 2 GB of RAM.		
IMAGE NAME	**Sequential segmentation (MS)**	**Parallel segmentation (MS)**
Camera Man	4632.15	2316.07
Lake	4413.37	2942.36
Mandrill	7331.32	4665.66
Pepper	6045.8	3144.13
Leena	6124.5	3141.15

TABLE V: EXPERIMENTAL RESULT USING CORE I5 PROCESSOR

The experiment result performed on an Intel Core i5 2.9 GHz, 4 GB of RAM.		
IMAGE NAME	**Sequential segmentation (MS)**	**Parallel segmentation (MS)**
Camera Man	3708.479	1951.84
Lake	4015.26	2342.05
Mandrill	6022.64	3824.01
Pepper	5045.69	2557.12
Leena	5481.98	2464.33

TABLE VI: EXPERIMENTAL RESULT USING CORE I7 PROCESSOR

The experiment result performed on an Intel Core i7 3.4 GHz, 8 GB of RAM.		
IMAGE NAME	**Sequential segmentation (MS)**	**Parallel segmentation (MS)**
Camera Man	2436.87	1224.02
Lake	1961.565	1307.71
Mandrill	5874.45	2048.41
Pepper	3712.23	1502.32
Leena	2735.464	1532.69

D. Results of Parallel Segmentation Images Output

The output segmented image generated by existing survey depicts that most the parallel algorithms require special hardware which are costly. Due to the revolution in computer architecture, multicore devices are very common today inside nearly all desktops and laptops, most gaming consoles, and the newest smart phones. This circumstance demands user applications to scale accordingly and perform sophisticated operation System (on the left) and our proposed system (on the right) are same. Even though our implemented system is faster than the existing one, it does not hamper the segmentation quality. With all the five images both multicolored and gray scale the test is done and all result is same as the existing one.

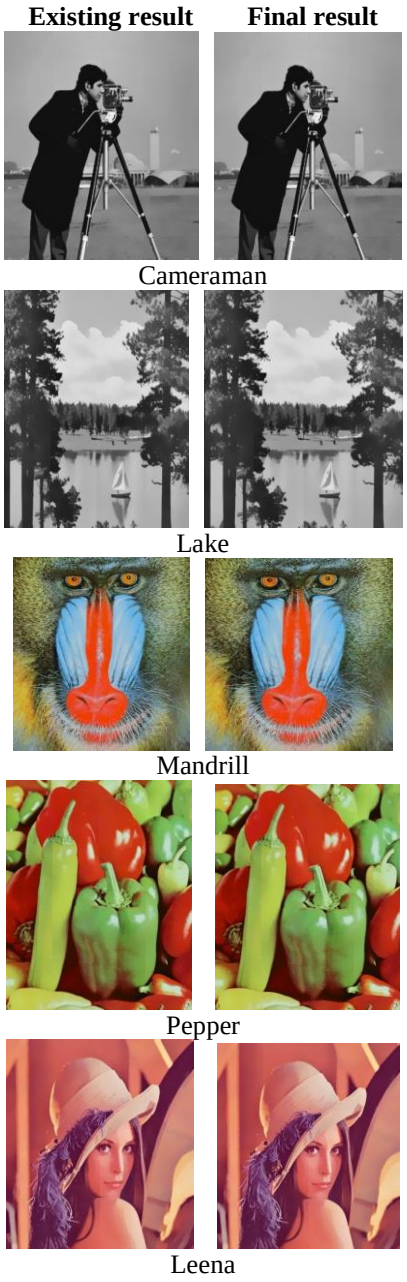


Figure 4: Results of parallel segmentation images output.

IV. CONCLUSION

Computer applications, exploiting the parallel platforms, are increasingly needed to meet the demand of efficient high volume of data processing. The existing literature survey depicts that most the parallel algorithms require special hardware which are costly. Due to the revolution in computer architecture, multicore devices are very common today inside nearly all desktops and laptops, most gaming consoles, and the newest smart phones. This circumstance demands user applications to scale accordingly and perform sophisticated operation speedily. In this paper, we introduced a low cost high performance mean shift algorithm to meet this demand. Our system uses the available CPU architecture and a library of functions providing parallel programming constructs. Learning PPL is also fairly easy for anyone having a little experience with C or C++ languages. Our experimental analysis confirmed that our system runs faster without sacrificing the quality of the output. In terms of performance, the parallel implementation is about two and a half times faster than the sequential one. Although the developed system successfully exploited the available parallel computing platform, it has some limitation. Our approach requires additional computation for partitioning the image, without proper partitioning the output will be distorted. Sometimes parallel computing takes longer time due to inter process communication. Another demerit of our system is that for a single processing unit it is not very effective. While generating the output image our system leaves a linear pixel difference which could be removed using carefully chosen adaptive partitioning mechanism

V. SEGMENTATION IN FUTURE

In the future, our intention is to use the same principle of division of work to develop systems capable of dealing with every kind of data, not only images. Thus, it is expected that our image segmentation procedure can handle extremely large images efficiently and without requiring special hardware.

REFERENCES

[1] B. Varga And K. Karacs. High-Resolution Image Segmentation Using Fully Parallel Mean Shift. Varga And Karacs Eurasip Journal On Advances In Signal Processing, 2011.

[2] F. Zhou, Y. Zhao, and K-Liu Ma. Parallel Mean Shift for Interactive Volume Segmentation. Wanget al. (Eds.): Springer-Verlag Berlin Heidelberg. MLMI 2010, LNCS6357, pp.67–75, 2010.

[3] D. Comaniciu and P. Meer. Mean shift: A Robust Approach Toward Feature Space Analysis. IEEE Transaction on pattern Analysis and machine intelligence, Vol24No (5), 603-619. May 2002.

[4] K LWu, and M S Yang. Mean shift-based clustering. The journal of the pattern recognition society. February 2007.

[5] S. Theodorids, and K. Koutroumbas. Pattern Recognition. Fourth Editions pp595-765, Academic Press. October 20, 2008.

[6] C. Compbell, and A. Miller. Parallel Programming with Microsoft visual C++, *Publisher*: *Microsoft* Press; 1st Edition April 11, 2011.740-741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].

[7] Y Cheng, Mean shift, mode seeking, and clustering. IEEE Transaction on pattern Analysis and machine intelligence, vol 17 No 8, 790–799. August 1995.

[8] L. Men, M. Huang, and J. Gauch. Accelerating Mean Shift Segmentation Algorithm on Hybrid CPU/GPU Platforms. October 2009.

[9] J. Wang, B Thiesson, Y Xu, and M. Cohen. Image and Video Segmentation by Anisotropic Kernel Mean Shift. Microsoft Research (Aisa and Redmond), 2004.

[10] YJing and Shukui Bo. Image Clustering Using Mean Shift Algorithm. Fourth International Conference on Computational Intelligence and Communication Networks, 2012.

[11] B. Georgescu, I .Shimshoni, and P. Meer. Mean shift based Clustering in High Dimension:A Texture Classifications Example. IEEE International Confe- rence on computer Vision vol set-2 ICCV 2003.

[12] KA. Shah, HK. Kapadia, VA. Shah, and Maurya N. Shah. Application of Mean-Shift Algorithm for License Plate Localization. IEEE Digital objects Identifier Ahmedabad Gujarat, 2011.

[13] P F. Felzenszwalb and D P. Huttenlocher, Efficient Graph-Based Image Segmentation, International Jour- nal of Computer Vision vol 59 Issue 2, pp 167-181 September 2004.

[14] P.N. Happ, R.S. Ferreira, C. Bentes, G.A.O.P. Costa, and R.Q. Feitosa. Multi resolution Segmentati on:A Parallel Approach For High Resolution Image Segmentation In Multi core Architectures. 3rd International Conference on Geographic Object-Based Image Analysis, Brazil, 2010.

[15] W. Khan. Image Segmentation Techniques: A Survey .COMSATS Institute of Information Technology, Wah Cantt, Pakistan Journal of Image and Graphics vol 1, No 4, December 2013

[16] DS. Hochbaum. An Efficient Algorithm for Image Segmentation, Markov Random Fields and Related Problems. Journal of the ACM, vol. 48, No. 4, pp. 686–701.July 2001.

[17] A.B.M. Faruquzzaman, N. R. Paiker, J Arafat, and M. A. Ali. A survey report on image segmentation Based on split and merge algorithm. IETECH Journal of Advanced Computations, vol. 2, No: 2, 086–101, 2008.

[18] A.B.M. Faruquzzaman, N. R. Paiker, J Arafat, and M. A. Ali. A survey report on image segmentation Based on split and merge algorithm. IETECH Journal of Advanced Computations, vol. 2, No: 2, 086–101, 2008.

[19] P. Li and L. Xiao. Mean Shift Parallel Tracking On GPU. 4th Iberian Conference on Pattern Recognition and Image Analysis, pp120-127. 2009.

[20] J. Shi and J.Mali. Normalized Cuts and Image Segmentation. IEEE transactions on pattern an alysisand machine intelligence, vol. 22, No. 8, August 2000.

[21] J. Malik, S. Belongie, T. Leungand, J. Shi. Contour and Texture Analysis for Image Segmentation. International Journal of Computer Vision vol.43, No 1, pp 7–27, Kluwer Academic Publishers. Manufact- ured in The Netherlands. 2001.

[22] J. Chen and N. Pappas. Adaptive Perceptual Color-Texture Image Segmentation. IEEE transactions on image processing, vol. 14, No. 10, October 2005.

[23] Z Tu and S c Zhu. Image segmentation by data-driven Markov chain Monte Carlo. IEEE Transactions On Pattern Analysis and Machine Intelligence, vol. 24, No 5, May 2002.

[24] B. Peng, L. Zhang, and J. Yang. Iterated Graph Cuts for Image Segmentation.9th Asian Conference on Computer Vision vol 5995, pp 677-686, 2010.

Raj Kumar Sah received his B.Sc. degree in Computer Science and Engineering from Khulna University ,Khulna, Bangladesh, in 2014, and currently studying in M.Tech. Research Scholar in Computer Science and Engineering from A P Goyal Shimla Universit, Shimla (H.P.)India (2016-2018) .His research interet includes Image Processing, At present; He is engaged in image segmentation application.

Md Zahidul received B.Sc degree in Computer Science and Engineering. Khulna University, Khulna, Bangladesh, in 2006, and the M.S. degree in Comp- uter Science from the Department of Mathematics, Statistics and Computer Science of St. Francis Xavier University, NS, Canada .in 2012, He has been teaching as lecturer, assistant professor of Computer Science and Engineering discipline at Khulna University.in 2008-2016 respectively. His research interests include Machine Learning, Computer Vision, and Artificial intelligence.