Parallel Detection of Objects in Large Images

Fazal Noor
Computer Science and Software Engineering Department
University of Hail
Hail, Saudi Arabia

***Abstract*— Accurate object detection in large images (aerial images) is extremely time consuming when carried out by sequential methods and in some cases needs to be performed close to real time. In this paper, we study the Phase Correlation method with Log-Polar Transform and the Normalized correlation method to detect objects in parallel in an image using a PC Cluster. In order to carry out the detection process efficiently and rapidly, the large image is broken into grid blocks of different sizes of 256 x 256 bits to 4096 x 4096 bit sizes and Correlations are performed. As the smaller image blocks cause greater communication time in their transmission from and to the master node, the computation times are reduced at the worker nodes. The main problem is when an object image is spilled over into 2 or more neighbor blocks. This scenario produces multiple PC peaks and neighbor blocks are checked and then stitched together to form a larger block and verification is carried out again.**

Keywords-Phase Correlation; PC Cluster; Fast Fourier Transform;

I. Introduction

Detecting objects in a cluttered environment is a challenging task. Normal image processing methods do not work properly for large images and may break down. First, large images require high capacity memory to store them. Second, large images can cause problems to process. In order to remedy these problems, the large images may be processed incrementally or by using parallel distributed methods [1],[2].

On PC clusters the large image problems may occur in communications and as well in computations [3]-[5]. Transmission of large communications would require large bandwidths. The image is divided into blocks of specified size and the blocks distributed to the worker nodes. The frequent transmission of images becomes communication intensive [5]. On the other hand, the computational problem would be in reading an image, processing an image, and third writing the results to the hard disk [4].

The objective of this paper is to use a PC Cluster for super fast computations at an affordable costs. The main contribution of this paper is to use the PC Cluster in conjunction with parallel phase correlation method and log-polar transform and normalized correlation method to detect and recognize an object in a large image. The large image is first broken into grids and experiments were carried on grids of size 256 x 256 and 512 x 512 bits, respectively.

The paper is organized as follows, in Section II, the methods are presented. In Section III, the results and discussions are presented and in Section IV, conclusion and future directions are given.

II. METHODS

In order to devise efficient algorithms, one has to look at factors which may affect performance. These factors are many and related to both hardware (e.g. CPU clock speed, memory size), network (e.g. cable type, switches, routers), and software (operating system, types of MPI routines, efficiency of algorithms) [5]. Therefore the performance depends on the execution time which in turn is a function of both hardware and software. The transmission of large images require communication time which cannot be ignored in evaluating performance. Therefore, the communication times and computation times are important factors to be considered in performance analysis. The computational complexity of algorithms have to be checked and also taken into consideration.

The time taken to send an image from one node A to another node B in terms of computation times and communication time can be written as,
[5],[7],

$$T = t_{compA} + t_{comm} + t_{compB} \ , \tag{1}$$

where t_{comp} time is given as,

$$t_{com\,p} = I \cdot C / f \ . \tag{2}$$

where I is instructions per program, C is clock cycles per instruction and f is CPU frequency measured in Hertz. The time involves the message's route from the transmitting computer, through the transmission media, and then to the receiver computer.

In applications where the topology is of one master node and many slaves nodes, MPI collective operations are useful such as broadcast, scatter, and gather routines [5]. The MPI collective operation's time would depend on the size of each message, number of messages, interconnection structure, and network contention.

The communication time can be written as [5],[7],

The author would like to thank the CSSE Information Systems and Networks Research centre to carry out the experiments. This research was supported by University of Hail research grant no. RS-2013-CS-12.

$$t_{comm} = t_s + m \cdot t_t + t_c \ . \quad (3)$$

where t_s is the message latency and takes into consideration the overhead time at the source and destination; t_t is the transmission time computed as *1/Bw* where *Bw* is link bandwidth given in *U* bits/sec; and m represents message to send. The time t_c represents contention time and is burst dependent. The time complexity is $O(k)$ for k data items. In the case, a master node transmits messages to multiple destination nodes, the 1-to-*N* fan-out broadcast is defined as the transmission of the same message to *N* destination nodes sequentially. The communication time in this case is defined as,

$$t_{comm} = N(t_s + m \cdot t_t), \quad (4)$$

and for the *gather* and *scatter* communication models is defined as,

$$t_{comm} = N(t_s + m \cdot t_t / n) \ . \quad (5)$$

The scatter routine is used to send a unique message from the source to every other destination and in the gather routine a unique message is received from every other node.

The time complexity is $O(Nm)$ for one source transmitting to *N* destination nodes. Note, in (5), *n* represents the total number of nodes and $N = n$-1.

The communication time for an Ethernet LAN network can be defined as:

$$t_{comm} = t_m + m_E \cdot t_t + t_q + t_p + t_E \quad (6)$$

where all terms depend on message size and message time at Media Access Control (MAC); t_m is dependent on message size and would be same for a homogeneous network and would be some factor multiplied by message size, $\alpha \cdot m_{size}$; m_E represents the number of bits in an Ethernet packet; t_q is the queuing delay; t_p is the propagation delay defined as *d/c*, where *d* is the length of the link and *c* is speed of light in medium having a value less than $3x10^8$ m/sec; t_E is the Ethernet interframe gap which is 12 bytes (96 bits) [7].

A. PC Cluster

Now a day PC Clusters are much more affordable than a super computer. PC Clusters offer fast computation speeds at very affordable price. PC Cluster can be built using off the shelf computers networked together using fast Ethernet switches. Our testbed consists of a PC cluster of 20 Lenovo machines with the following specifications: Intel Core™ 2 Duo CPU, E4400 2.00GHz, 1.00 GB of RAM. Network Card: Broadcom Netlink, Gigabit Ethernet. The PC are connected to a Gigabit Ethernet switch. RedHat Enterprise AS Linux operating system is installed on each machine, and use LAM 7.0.6/MPI 2.

B. MPI Collective Communication Routines

The Message Passing Interface (MPI) is a message passing library standard interface that is broadly used for writing message passing programs on PC platforms. The main MPI collective communication functions are: *All-to-all*: every node sends a message to every other node. *Broadcast*: one node sends one message to every other node. *Gather*: all nodes send a different message to a single node. *Scatter*: a single node sends a different message to every other node. *Point-to-Point*: a single message is sent/received between 2 specific nodes.

C. Correlation Methods

In signal processing, correlation filters are used to detect how similar two signals are. In image processing, correlation methods are used in object detection and recognition problems. These correlation filters are found in many applications such as medical imaging, voice recognition, biometrics iris, face recognition and many more.

In image processing, phase correlation is a method used to for object tracking, motion estimation, and image registration [9]. Phase correlation may be used to check the similarity of two images of equal pixel size. Recognition of a scaled, rotated, and translated image can be performed using phase correlation with Log-polar Transform [7],[8].

Given two images of size *P x Q*, f(*x,y*) and h(*x,y*), their 2D Discrete Fourier Transforms (2D DFTs) are denoted by F(*u,v)* and H(*u,v*). The cross spectrum $R_{FH}(u,v)$ of $F(u,v)$ and $H(u,v)$ is given by

$$R_{GH}(l,m) = F(l,m)\, H^*(l,m) \quad (10)$$

where * is used to denote the complex conjugate. Cross-phase spectrum $R_{FH}^N(l,m)$ is defined by

$$R_{FH}^N(l,m) = \frac{F(l,m)H(l,m)*}{|F(l,m)H(l,m)*|} = e^{j\theta(l,m)} \quad (11)$$

Phase correlation function $r_{FH}^N\ (p,q)$ is the 2D inverse DFT of (11) and is given by

$$r_{fh}^N\ (p,q) = \frac{1}{P \cdot Q} \sum\sum R_{FH}^N(l,m) W_P^{-lp} W_Q^{-mq} \quad (12)$$

In the case f(x, y) and h(x, y) are the same image then the Phase Correlation function is given by

$$r_{ff}^N\ (p,q) = \frac{1}{P \cdot Q} \sum\sum W_P^{-lp} W_Q^{-mq} = \delta(p,q) \quad (13)$$

where $\delta(p,q)$ is the Dirac delta function. The peak point location is given by

$$\delta(x,y) = \max_{(x,y)} \{r^N\} \tag{14}$$

In the case where the two images are same their Phase Correlation function gives a distinct sharp peak. The peak height is used to measure the similarity between two images. The Phase Correlation function has the following properties, shift invariance, amplitude invariance, and immunity against additive noise. The properties can be verified directly from the Fourier Transform's properties [4].

In case one image is a translated version of the same image then the images are related by

$$g(x,y) = f(x-x_0, y-y_0), \tag{15}$$

that is g(x, y) is translated by an offset (x_0, y_0). From the Fourier transform shift property the images are related by

$$F(u,v) = H(u,v)e^{-j(ux_0+vy_0)}. \tag{16}$$

From the normalized cross-power spectrum the exponential phase shift factor can be obtained as,

$$P(u,v) = \frac{H(u,v)F^*(u,v)}{|H(u,v)F^*(u,v)|} = e^{j(ux_0+vy_0)}. \tag{17}$$

The inverse Fourier transform gives the Dirac delta function at the offset (x_0, y_0),

$$p(x,y) = \delta(x-x_0, y-y_0). \tag{18}$$

Let f(x, y) be a scaled, rotated, and translated version of h(x, y), defined as,

$$f(x,y) = h(s\cdot x\cos\theta - s\cdot y\sin\theta - x_0, \; s\cdot x\sin\theta + s\cdot y\cos\theta - y_0) \tag{19}$$

Then the Fourier transform would be

$$F(u,v) = \frac{1}{|s^2|} H(u'\cos\theta - v'\sin\theta, u'\sin\theta + v'\cos\theta) \cdot e^{-j(ux_0+vy_0)} \tag{20}$$

where u' = u/s and v'= v/s. The magnitude of above (20) is

$$M_F(u,v) = w\cdot M_H(u'\cos\theta - v'\sin\theta, u'\cos\theta + v'\sin\theta)) \tag{21}$$

where w is a weighting factor. Using polar coordinates (r,ϕ) and letting $u = r\cos\phi$ and $v = r\sin\phi$. Inserting in (21) above results in,

$$M_F(u,v) = w\cdot M_H(r'\cos\phi\cos\theta - r'\sin\phi\sin\theta, \; r'\cos\phi\sin\theta + r'\sin\phi\cos\theta)\cdot \tag{22}$$

and using trigonometric identities results in

$$M_F(u,v) = w\cdot M_H(r'\cos(\phi+\theta) - r'\sin(\phi+\theta)) \tag{23}$$

which can be represented as,

$$M_F(r,\phi) = w\cdot M_H(r', \phi+\theta). \tag{24}$$

Converting to log-polar form, by taking the logarithm results in

$$M_F(\log r, \phi) = w\cdot M_H(\log r', \phi+\theta) \tag{25}$$

where $r' = r/s$ and $\log r' = \log r - \log s$. From the Fourier properties, a positional shift produces a phase shift and linear scaling of a spatial variable x,y produces an inverse scaling of spatial frequencies u,v. The phase-correlation method can be used to obtain the translation in log-polar coordinate system. The algorithm to perform distributed Phase Correlation can be used to check if two images match or not. Figures 2 and 3 show the block diagram of the Phase Correlation steps involved.

Phase Correlation Algorithm with Log-polar Transform:

Given two images f(x,y) and h(x,y), where f(x,y) is the reference image and h(x,y) is the image to register or match. The parallel phase correlation and log-polar transforms (PPCLT) are computed in parallel as shown below:

Algorithm PPCLT

In each steps 1 to 6 and then in steps 9 to 11 perform the indicated operations in parallel:

Step 1. Read the images f(x,y) and h(x,y)
Step 2. Take the FFT of the images g(x,y) and h(x,y), shift to center zero frequency.
Step 3. Convolve magnitude of F(u,v) and H(u,v) each with a high pass filter.
Step 4. Use log-polar transform to transform in log polar space.
Step 5. Take the FFT of each.
Step 6. Computer phase correlation of the two in step 5.

Step 7. Find the peak location (x,y).
Step 8. Compute angle and scale factor.

Step 9. Rotate image by angle and another by angle + 180 degree.
Step 10. Take the FFT of both rotated images.
Step 11. Compute phase correlation of both with reference image.

Step 12. Find the peak locations in each.
Step 13. Compare the two peaks and obtain the translation

In parallel computing, speedup is one of the metric used to measure performance and is given as,

$$S_p = \frac{T_s}{T_p} = \frac{T_s}{T_{comp} + T_{comm}} \tag{22}$$

where T_s is the sequential execution time on one node, and T_p is the execution time on P nodes consisting of computation and communication time. Note in parallel processing one would prefer the computation time to be higher than communication time.

Parallel distributed algorithm:
Master Node:

Step 1: Given a Large Image Partition into M blocks.

Step 2: Given a template image to search. Note: template is smaller than partitioned image blocks.

Step 3: Scatter the M/N blocks to N workers.

Step 4. Broadcast the template to N workers.

Worker Nodes:

Step 1:
For i = 1 to M/N blocks

Use Phase Correlation method or Normalized Correlation Method to check if a match occurs.

Step 2: Send Phase Correlation Peaks back to Master node.

III. RESULTS AND DISCUSSION

A large image shown in Fig 1 was partitioned as in Fig 2 below. The partitioned blocks are MPI scattered to the worker nodes by the master node. The template to be searched and matched is MPI broadcast to all the participating worker nodes. The worker nodes use the Phase correlation method to check if a match occurs. Each worker node has a block of an image and a template. A match occurs if the Phase correlation peak is greater than a threshold otherwise it is considered no match. All the workers transmit back their Phase correlation peak value. The master node than makes the judgment of whether there is a match or not.

Case A: Number of image blocks are larger than the number of worker nodes. In this case a worker when it finishes working on one block of image and sends the result back to the master will receive another block of image to work on. The sending of blocks and receiving of results is done by the master node till all the blocks of image are worked on by the worker nodes. In a homogenous PC cluster environment the communication times and computing times are fairly load balanced. In a heterogeneous PC cluster the fast worker nodes would be having a higher work load.

Another issue is the image block sizes, the smaller the image blocks the greater the number of blocks to work on and the greater the communication overhead. However the smaller blocks will be computed faster. In experiments carried out, the block image sizes were 256 x 256 bit size and 128 x 128 bit size. The template (patch) image size was less than the block sizes. They were then zero padded to the block size in order to carry out the Fast Fourier Transforms. The threshold value was chosen as 0.8 and produced results with high accuracy of 99. When the template image does not match the PC peaks are spurious and have very low values in range of .001 to .2. In other words the mismatch produces PC peaks like white noise.

Another issue that may arise is if the object being detected is split across 2 or more blocks of images. In this case, multiple workers will produce a match. Then the blocks are checked if they are in a neighborhood of each other. If they are then the object is considered to be partially in multiple image blocks. Further processing techniques can be used to verify the matching using stitching techniques. Once the blocks are stitched together forming a larger block of image. Again PC method can now be applied on this larger image block to check if the whole object matches or not.

Case B. Detecting multiple objects. In this case multiple templates are sent to worker nodes. Suppose K objects are to be detected in the large image. The K templates are then scattered to worker nodes. Each worker node will have one object template and multiple image blocks to search in and check for a match.

Figures 8 and 9 show the communication time for a scatter routine for image sizes of size 512 x 512, 1024 x 1024, 2048 x 2048, and 4096 x 4096 and for PC cluster of sizes 2, 4, 8, 16 computers, respectively.

The speed up obtained is shown Table 1. The speedup is shown for a image of size 2048 x 2048 and PC cluster of different sizes of 2, 4, 8, and 16. It is shown the communication time increases with PC cluster size. The speedup is fairly increasing with PC cluster size.

Figure 1. Large image of a Crane.

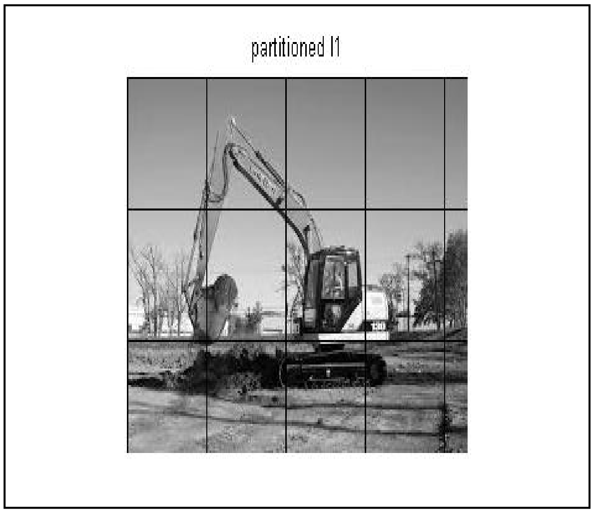


Figure 2. Partition the image into grids of size 256 x 256.

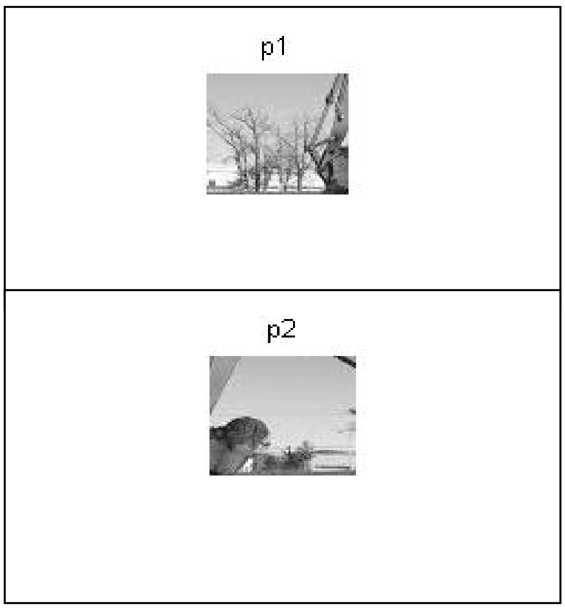
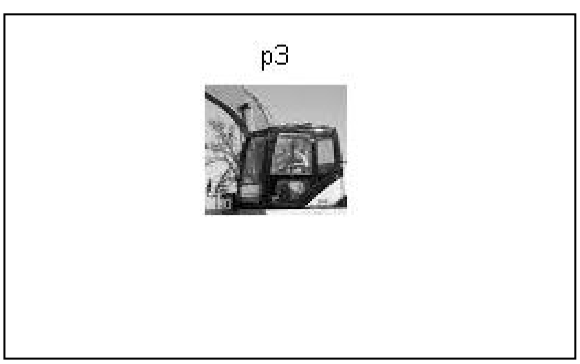


Figure 3. Patches p1 and p2 to be detected in the large image.



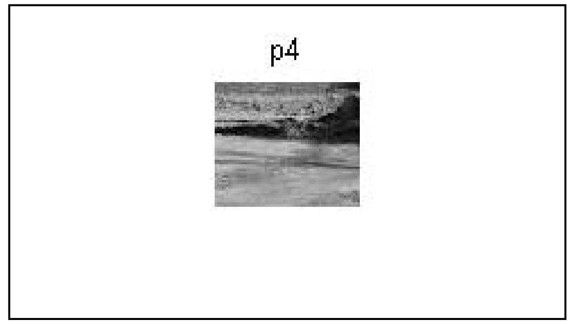


Figure 4. Patches p3 and p4 to be detected in the large image.

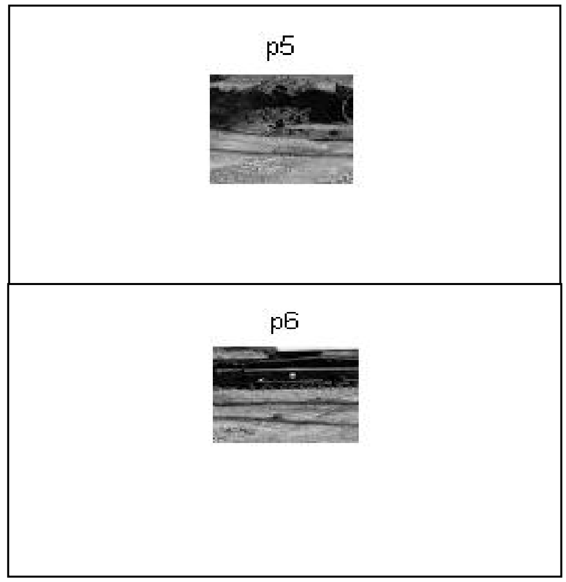


Figure 5. Patches p5 and p6 to be detected in the large image.

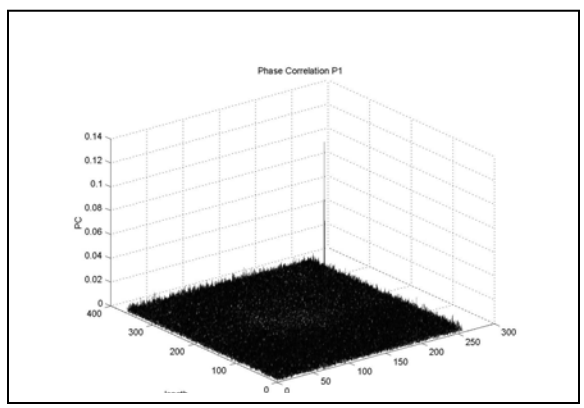


Figure 6. Phase correlation method shows a match with a distinct sharp peak.

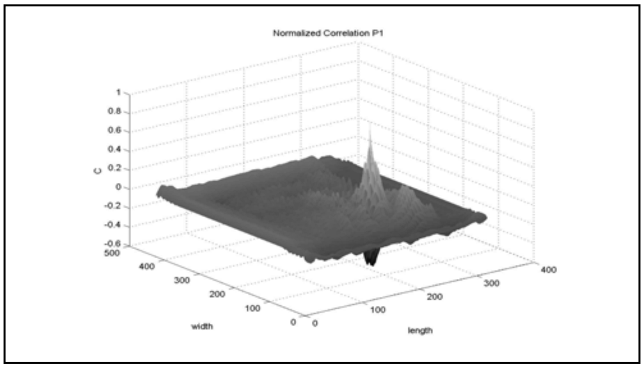


Figure 7. Peak showing there is a match using Normalized Correlation.

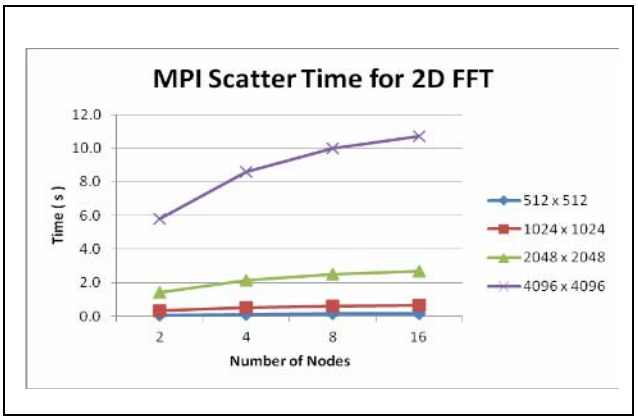


Figure 8. MPI Scatter communication time for different sizes of images and for a PC cluster of sizes 2, 4, 8, and 16.

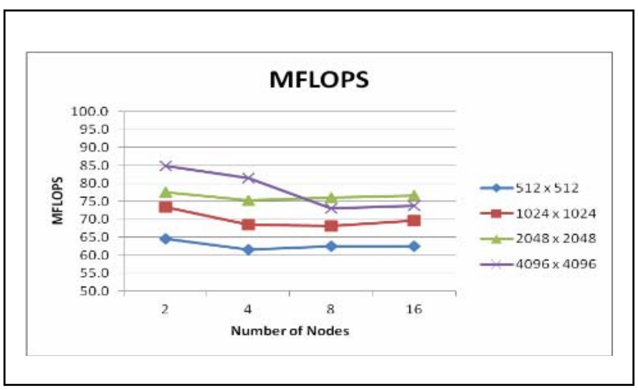


Figure 9. Computation time in millions of floating point operations time for different sizes of images and for a PC cluster of sizes 2, 4, 8, and 16.

TABLE I. SPEED UP FOR PC CLUSTER OF DIFFERENT SIZES.

PC cluster	**Image Size of 2048 x 2048 Ts = 740 msec**		
	Communication time (msec)	***Computation time (msec)***	***Speedup***
2	1.8	398.2	1.8
4	2.3	197.6	3.7
8	2.7	98.5	7.3
16	3.1	49.3	14.2

)

IV. CONCLUSION AND FUTURE WORK

In this paper, an efficient parallel algorithm is presented to detect objects in large images. The algorithm uses parallel methods to distribute the blocks of images using MPI scatter routines to worker nodes and broadcasts template of object to the worker nodes. The workers perform the task of detecting the object in its block. A match is considered if the phase correlation peak is above a defined threshold value otherwise it is considered a mismatch. If the object image spills into neighbor blocks, then the image is split again but with larger grid size and the algorithm repeated. In this scenario, the neighbor blocks are stitched together forming a larger block image. In this situation the blocks are stitched and again verified for a further match. Furthermore, it is seen the speedup increases with increasing PC cluster size. Automatically detecting of partial images of objects is the subject of future research.

ACKNOWLEDGMENT

The author would like to thank the CSSE Information Systems and Network Research Center at the University of Hail for carrying out experiments. This research was supported by University of Hail Research Grant no. R2-2013-CS-12.

REFERENCES

[1] Sitaraman, K. et. al, A parallel algorithm and architecture for object recognition in images, IEEE International Workshop on Computer Architecture for Machine Perception, New Orleans, pp 16-148, May 2003.

[2] Zahid Mahmood, et. al, A Parallel Framework for Object Detection and Recognition for Secure Vehicle Parking, Journal of Experimental Psychology, Vol. 31, No. 6, pp. 1476-1492, 2005.

[3] Yongjun Ma, et. al, Algorithm for Object Detection using Multi-Core Parallel Computation, International Conference on Medical Physics and Biomedical Engineering (ICMPBE2012), Physics Procedia, Vol. 33, pp. 455-461, 2012.

[4] W. K. Pratt, Digital Image Processing, Fourth Edition, John Wiley and Sons, Inc., 2007.

[5] B. Wilkinson, M. Allen, Parallel Programming Techniques and Applications Using Networked Workstations and Parallel Computers, Second Edition, Pearson Prentice Hall, 2005.

[6] S. Haykin, Neural Networks: A Comprehensive Foundation, 2nd Edition, 1998.

[7] Noor F., Alhaisoni M., Liotta A., An Empirical Study of MPI over PC Clusters, AP2PS 2011 The Third International Conference on Advances in P2P Systems, pp 65-70, 2011.

[8] Hongshi Yan*, Jian Guo Liu, Robust Phase Correlation Based Feature Matching for Image Co-registration and Dem Generation, The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences. Vol. XXXVII. Part B7, Beijing 2008.

[9] Nakamori, T., Okabayashi, H. , Kawanaka, A., 3D Object Matching using Correlation Method, IEEE 10th International Conference on Signal Processing, Beijing 24-28 Oct 2010, pp. 1275 - 1278.

[10] Noor F., Alhaisoni, M., Distinguishing Moving Objects using Kalman Filter and Phase Correlation, IEEE 17th International Multi-Topic Conference, 8-10 Dec 2014, pp 299 - 304, 2014.

[11] F. A. Vaughan, D. A. Grove, P. D. Coddington, "Communiation Issues for Two Cluster Computers," ACSC '03 Proceedings of the 26th Austrailasian computer science conference, vol 16, 2003.

[12] T. Kielmann, H. E. Bal, "Fast Measurment of LogP Parameters for Message Passing Platforms", 4th Workshop on Runtime Systems for Parallel Programming (RTSPP), pp. 1176-1183, held in conjunction with IPDPS 2000, Cancun, Mexico, May 1-5, 2000. Lecture Notes in Computer Science, Vol. 1800.

[13] J. P. Grbovic, et al, "Performance Analysis of MPI Collective Operations", Journal Cluster Computing, Vol 10, Issue 2, pp.127-143, June 2007.

[14] R. Riesen, "Communicaton Patterns", Parallel and Distributed Processing Symposium, 25-29 April 2006.

[15] A. Leko, H. Sherburne, et al, "Practical Experiences with Modern Parallel Performance Analysis Tools : An Evaluation", Parallel and Distributed Processing, IPDPS 2008 IEEE Symposium 14-18 April 2008, Miami, Fl, pp. 1-8.

AUTHORS PROFILE

Dr. Fazal Noor: received his PhD in Electrical and Computer Engineering from McGill University, Canada in 1993. He received his M. Eng and B. Eng from Concordia University, in 1986 and 1984, respectively. Since 2006, Dr. Noor is working as assistant professor for the Computer Engineering Department at the University of Hail. Dr. Noor received a Faculty Award for his contributions and achievements in year 2007, from University of Hail. Dr. Noor has numerous publications in international conferences and journals. His current research interests include; Image recognition, Parallel and Distributed Computing, Fingerprint Verification thru Cloud Computing, Embedded Systems, Robotics and Computer Vision, Spread Spectrum Communications and Signal Processing.