Research Journal of Applied Sciences, Engineering and Technology 8(9): 1066-1074, 2014
DOI:10.19026/rjaset.8.1071
ISSN: 2040-7459; e-ISSN: 2040-7467

Submitted: March 19, 2014 Accepted: May 08, 2014 Published: September 05, 2014

Research Article

Parallel Vectoring Algorithm for Pattern Matching

[1, 2]Khaled Ragab and [1, 3]Y.M. Fouda
[1]College of Computer Science and Information Technology, King Faisal University, P.O. Box 400, Al-Ahsa 31982, Kingdom of Saudi Arabia
[2]Department of Math, Computer Science Division, Faculty of Sciences, Ain Shams University, Cairo, Egypt
[3]Department of Math, Faculty of Science, Mansoura University, Mansoura 35516, Egypt

Abstract: The main aim of this study is to propose efficient algorithms for image template matching on parallel machines. To our knowledge there is no a template matching algorithm that converts the template image and all candidate blocks in the source image from two-dimensions into one-dimension. This study proposes a Template Matching Vectoring Algorithm (TMVA) that reduces the amount of data to be analyzed by transforming 2-D images into 1-D images. Moreover, to speed up the vectoring pattern matching algorithm this study implements it in parallel. Complexity analysis and experimental results demonstrate that the performance of the proposed algorithm is superior to the other basic template matching algorithms.

Keywords: Image dimensions reduction, image processing, parallel processing, template matching

INTRODUCTION

Template matching is a task of seeking a given pattern in a given image. It is widely used in signal processing, computer vision and image and video processing. Locating information from pictures has been widely used in industry, medicine, military, satellite communications and publishing. For example, using digital image processing techniques the X-ray pictures may be greatly enhanced to help medical diagnosis in Niblack (1986). In Printed Circuit Board (PCB) industry, it can be used for the visual inspection of PCBs for defects in Kidorf and Peigorsch (1984). Military stations may detect objects in infrared pictures taken in dark. Template matching tries to answer one of the most basic questions about image: Is there a certain object in that image? If so, where? The template is a description of that object and is used to search the image by computing a difference measure between the template and all possible portions of the image that could match the template: if any of these produces a small difference, then it is viewed as possible occurrence of the object (Mikhail, 2001). The template matching problem can be defined as follows. Given an $N\times N$ source image and an $M\times M$ template (or window), compare the template with all possible windows of the image. The result of each window operation is stored in the top left corner of the window. The image template matching problem has been studied extensively by several researchers found in the literatures (Kumar and Krishnan, 1987; Jenq and Sahni, 1991; Horng *et al*., 1991; Ranka and Sahni, 1988; Zitova, 2003).

According to Brunelli (2009), the main drawback of template matching is its high computational cost, which has two distinct sources. The first one is the necessity of using multiple templates to capture the variability exhibited by the appearance of complex objects. The second one is related to the size of templates: the higher the resolution, the heavier the computational requirements. With the rapid development of parallel processing systems, parallel computations for image processing and computer vision have received considerable attention during the past few years. The image template matching problem has been studied extensively by several researchers using various parallel computation models found in the literatures (Anderson *et al*., 2010; Armstrong *et al*., 1998; Fang *et al*., 1985; Mendez *et al*., 2011; Horng *et al*., 1991; Ranka and Sahni, 1988).

This study proposes an efficient pattern search parallel algorithm based on Dimension-Reduction approach for images. Dimension-Reduction technique is applied for the template image and all corresponding sub-windows in the source image. We call this approach Vectoring Technique (VT). The *VT* converts the template and all its corresponding sub-windows in the source from 2-D into 1-D. The sum of square difference measure was used as a similarity measure to get the template in the source.

Corresponding Author: Khaled Ragab, College of Computer Science and Information Technology, King Faisal University, P.O. Box 400, Al-Ahsa 31982, Kingdom of Saudi Arabia

LITERATURE REVIEW

Various difference measures have different mathematical properties and different computational properties had been used to find the best match. The most popular similarity measures are the Sum of Absolute Differences (SAD), the Sum of Squared Difference (SSD) and the Normalized Cross Correlation (NCC). Because of *SAD* and *SSD* are computationally fast and algorithms are available which make the template search process even faster, many applications of gray-level image matching use *SAD* or *SSD* measures to determine the best match. However, these measures are sensitive to outliers and are not robust to variations in the template, such as those that occur at occluding boundaries in the image. However, the *NCC* measure is more accurate but it is computationally slow. It is more robust than *SAD* and *SSD* under uniform illumination changes, so the *NCC* measure has been widely used in object recognition and industrial inspection such as in Du-Ming and Chien-Ta (2003) and Costa and Petrou (2000). An empirical study of five template matching algorithms in the presence of various image distortions has found that *NCC* provides the best performance in all image categories (Li *et al*., 1994).

Many improvements were achieved on *SAD* and *NCC* algorithms to improve their complexities. Mahmood and Khan (2012) proposed a partial computation elimination technique in which at a particular search location, computations are prematurely terminated as soon as it is found that this location cannot compete already known best-match location. They showed that partial elimination technique may be applied to correlation coefficient by using a monotonic formulation and proposed a basic mode for small template size and an extended mode for medium and larger template. Chen *et al*. (2001) proposed a fast block matching algorithm based on the winner-update strategy, which can significantly reduce the computation and guarantee to find the globally optimal solution. In their algorithm, only the current winner location with a minimal accumulated distortion is considered for updating the accumulated distortion. This updating process is repeated until the winner has gone through all levels in the pyramids that are constructed from the template and all the candidate windows for the distortion calculation.

Wei and Lai (2008) proposed a fast pattern matching algorithm based on *NCC* criterion by combing adaptive multilevel partition with the winner update scheme to achieve very efficient search. This winner update scheme is applied in conjunction with an upper bound for the cross correlation derived from Cauchy-Schwarz inequality. To apply the winner update scheme in an efficient way, they partition the summation of cross correlation into different levels with the partition order determined by the gradient energies of the partitioned regions in the template. Thus, this winner update scheme in conjunction with the upper bound for *NCC* can be employed to skip unnecessary calculation. Alsaade *et al*. (2012) used Cellular Automata with Rule 170 (CA-R170) on the images after converting it into binary images. Your technique based on eliminating some of the undesirable area in the binary source images and their corresponding binary template images. Essannouni *et al*. (2007) proposed a frequency algorithm to speed up the process of *SAD* matching using Fast Fourier Transforms (FFT). They introduced an approach to approximate the *SAD* metric by cosine series which can be expressed in correlation terms. These latter can be computed using *FFT* algorithms. Alsaade and Fouda (2012) proposed a matching algorithm based on *SAD* as a measure of similarity and pyramid structure. They applied the pyramid concept to obtain a number of levels of original and template images. Then the *SAD* measure is applied for each level of image from bottom to up to obtain the correct match in the original image.

Parallel template matching technique aims at improving performance by splitting a match computation into several match tasks and executing these tasks in parallel, e.g., on multi-core servers or on a cloud infrastructure (Fang *et al*., 1985). A key prerequisite for effective parallel template matching is the suitable partitioning of the input entities which influences the processor utilization, communication overhead and load balancing. Furthermore, template matching is typically memory-sensitive so that the definition of match tasks should consider the available memory that is typically shared among several cores. Image correlation is a computationally intensive algorithm that is used in several applications (e.g., image defect detection, image registration or pattern recognition). There are several trails to implement image correlation algorithms in Single-Instruction and Multiple-Data (SIMD) computers. Ranganathan *et al*. (1999) select 12 algorithms for image and video processing and measure a speedup in the range from 1.1 to 4.2 on a *SIMD* unit. Moreover, Armstrong *et al*. (1998) study the implementation of image correlation algorithms on various network-based *SIMD* and *MIMD* parallel architectures in Armstrong *et al*. (1998). However, this study implements parallel template matching algorithm. The proposed parallel algorithm speeds up the process for two reasons. First, it reduces the required computing time by converting the template and all its corresponding sub-windows in the source from two dimensions into one dimension. Second, it will be implemented based on parallel threads given to a multi-cores machine.

VECTORING METHODS

This section introduces the Vectoring Technique (VT) that converts the template and all its corresponding sub-windows in the source from *2-D* into *1-D*. Moreover, it analyzes the complexity of the

proposed *VT* technique and compares it with other four techniques.

Template matching vectoring algorithm: The proposed technique has been motivated by a need to develop an efficient matching technique so that the detection of objects in a source image can be effective and fast. The proposed Template Matching Vectoring Algorithm (TMVA) involves three phases. The first phase reduces the amount of data analyzed by transforming *2-D* images (template image and sub-windows which has the same size with template in the source image) into *1-D* vector information. This can be done by adding all intensity values for each column in *2-D* image as seen in Eq. (1), so we get *1-D* vector information. This vector information will be used in the matching process instead of the 2-D image. This transformation reduces the amount of image data from $m \times n$ to n. Subsequently, this allows the search to be performed with fewer data, while still taking all pixel intensity values into account. The second phase measures the similarity between the template image and all possible sub-windows in the source image. The Euclidean distance, the sum of absolute differences or the sum of square differences can be used as a similarity measure between the *1-D* template and all *1-D* converted sub-windows in the source. In the third phase, the decision will be taken based on the similarity values. The sub-window in the source with the minimum similarity value will be the best match for the template in the source.

The basic idea of the *TMVA* is to convert *2-D* template image into a *1-D* vector and also the corresponding windows in the source image. To illustrate the idea, suppose that we have a source image S of size $p \times q$ and template image T of size $m \times n$ where $m<p$ and $n<q$. The problem is to find the best match of template T from the source image S with minimum distortion. The proposed *TMVA* steps are as follows.

First converts the template image $T(i, j)$ into1-D vector NT by the following equation:

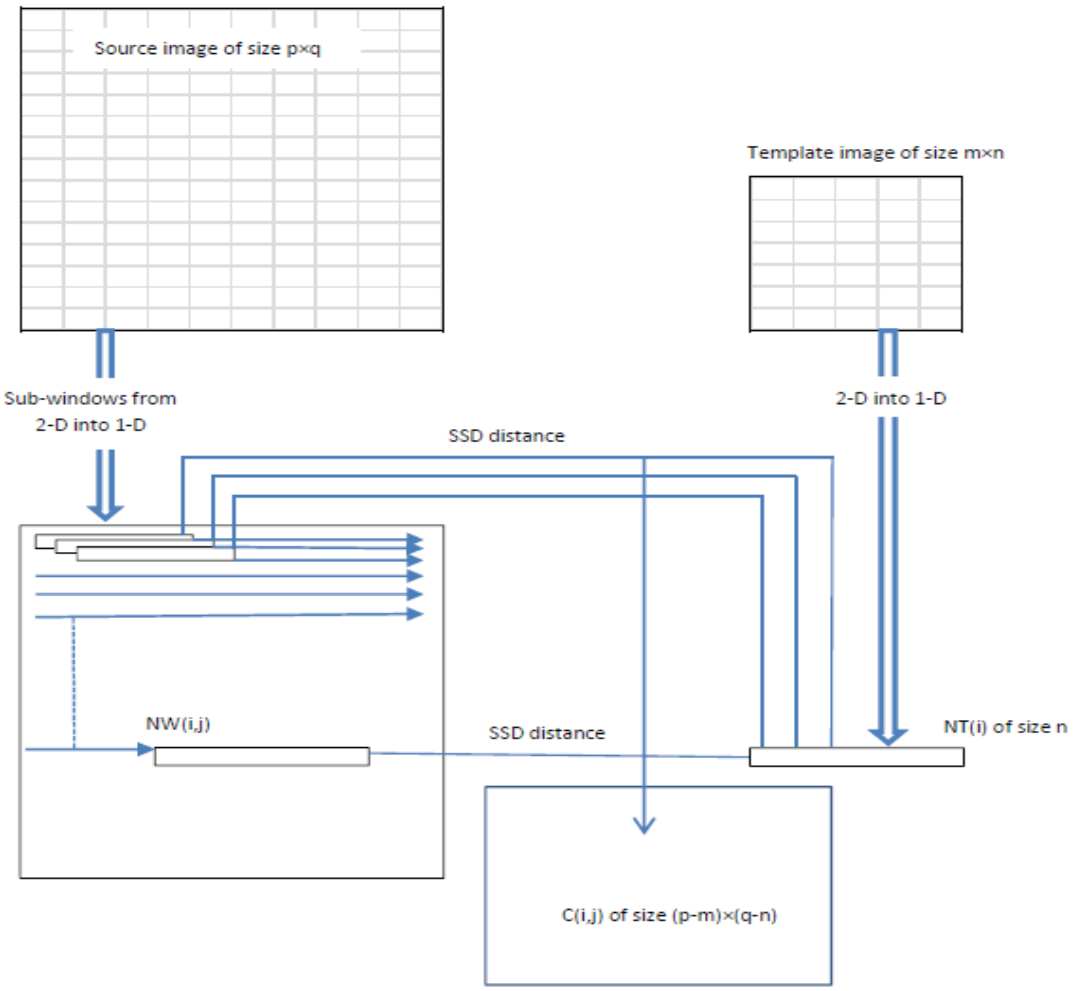


Fig. 1: Illustration of the proposed vectoring technique. The template image is converted into 1-D and also the corresponding windows in the source. SSD is calculated at every possible location in the source. The (i, j) location of matrix C with the minimum value is considered as the location of best match

$$NT(i)=\left(\sum_{i=1}^{m}T(i,1),\sum_{i=1}^{m}T(i,2),\sum_{i=1}^{m}T(i,3),\ldots\ldots,\sum_{i=1}^{m}T(i,n)\right) \quad (1)$$

where, $T(i, j)$ is the pixel value at location (i, j) of the template image.

Second, for each pixel (i, j) in the source image S, $1\leq i<p-m$ and $1\leq j<q-n$, we determine a window $W(i, j)$ of size $m\times n$ and all these windows converted into 1-D vector NW by the equation:

$$NW(i,j)=\begin{pmatrix}\sum_{k=i}^{m+i-1}W(k,j),\sum_{k=i}^{m+i-1}W(k,j+1),\sum_{k=i}^{m+i-1}W(k,j+2),\\ \ldots\ldots,\sum_{k=i}^{m+i-1}W(k,n+j-1)\end{pmatrix} \quad (2)$$

where, W (k, j) is the pixel value at location (k, j) of the source image.

Third, the likeness between template image and each corresponding window in the source are measured by sum of square difference distance between NT and NW. All these distances compute and store in new storage $C(i, j)$ where:

$$C(i,j)=\left(\sum_{k=i}^{m+i-1}W(k,j)-\sum_{i=1}^{m}T(i,1)\right)^2+\left(\sum_{k=i}^{m+i-1}W(k,j+1)-\sum_{i=1}^{m}T(i,2)\right)^2\ldots$$
$$+\left(\sum_{k=i}^{m+i-1}W(k,n+j-1)-\sum_{i=1}^{m}T(i,n)\right)^2 \quad (3)$$

where, $1\leq i<p-m$ and $1\leq j<q-n$.

Hence the left upper corner of best template match template position can be determined as the positive ordered pair $(\bar{\imath}, \bar{\jmath})$ where C $(\bar{\imath}, \bar{\jmath})$ is the lowest obtained distance. Figure 1 shows the concept of the proposed VT.

Complexity analysis: In order to evaluate the efficiency of the $TMVA$, we discuss the complexity of our algorithm compared with four important methods of template matching NCC, SAD, $NCCP$ and $SADP$. For NCC algorithm, the cross correlation coefficient between template image T of size $n\times n$ pixels and an $n\times n$ pixels block in source image S of size $p\times q$ is given by:

$$\lambda(i,j)=\frac{\sum_{x=1}^{n}\sum_{y=1}^{n}\left(S(i+x,j+y)-\bar{S}(i,j)\right)\left(T(x,y)-\bar{T}\right)}{\sqrt{\sum_{x=1}^{n}\sum_{y=1}^{n}\left(S(i+x,j+y)-\bar{S}(i,j)\right)^2\sum_{x=1}^{n}\sum_{y=1}^{n}\left(T(x,y)-\bar{T}\right)^2}}$$
$$1\leq i<(p-n),\ 1\leq j<(q-n) \quad (4)$$

where,

$$\bar{S}(i,j)=\frac{1}{n\times n}\sum_{x=1}^{n}\sum_{y=1}^{n}S(i+x,j+y)\ \ and\ \ \bar{T}=\frac{1}{n\times n}\sum_{x=1}^{n}\sum_{y=1}^{n}T(x,y)$$

Direct computation of $\lambda(i, j)$ require $n\times n = n^2$ (addition/multiplication) operations at each point (i, j) in the source image where $1\leq i<(p-n)$, $1\leq j<(q-n)$. Then the operations in Eq. (4) is proportional *to* n^2 $(p-n+1)$ $(q-n+1)$. So the computational cost of NCC is O $(n^2 (p-n+1) (q-n+1))$ which is very time consuming.

For SAD method the sum of absolute difference between the template T of size $n\times n$ pixels and blocks of sizes $n\times n$ pixels in the source image S is given as follows:

$$SAD(i,j)=\sum_{x=1}^{n}\sum_{y=1}^{n}\left|S(i+x,j+y)-T(x,y)\right| \quad (5)$$

The computation of $SAD(i, j)$ requires a number of operations proportional to the template area $(n\times n)$. These operations are computed for each (i, j) in the source image where $1\leq i<(p-n)$, $1\leq j<(q-n)$. Then the computational cost for SAD method is O $(n^2 (p-n+1) (q-n+1))$ the same in NCC algorithm but the SAD method is faster than NCC method because the number of operations in SAD is less than number of operations in NCC for each position (i, j) in the source image about seventy percentage.

The two methods $NCCP$ and $SADP$ can achieve the same estimation accuracy as NCC and SAD while needing much less computation requirement than these two methods. When the pyramid is applied for NCC and SAD, a sequence of compressed template and source images are created using the following Eq. (6):

$$I^k(x,y)=\frac{1}{4}\begin{pmatrix}I^{k-1}(2x,2y)+I^{k-1}(2x+1,2y)+\\ I^{k-1}(2x,2y+1)+I^{k-1}(2x+1,2y+1)\end{pmatrix} \quad (6)$$

where, $I^k(x, y)$ is the intensity value for the image in the level k. The search is conducted using NCC or SAD Eq. (4) or (5) with the most compressed template and source image. The resulting pixel location provides a coarse location of the template pattern in the next lower level of the source image. Therefore, instead of performing a complete search in the next level, it requires to only search a close neighborhood of the area computed from the previous search. This sequence is iterated until the search in the source image is completed.

In the pyramid concept the complexity of the algorithm depend on the position of the template in the source. If the template coordinates are far from the x-direction and y-direction then the application of the pyramid outperforms the original method. But, if the x-coordinate for template in source is close to x-axis and/or y-coordinate for template in source is close y-axis the original method outperform the pyramid concept. To overcome the problem in pyramid concept and the computational intensive in NCC and SAD we introduced our approach. The computation in Eq. (3) require a number of operations proportional to the length n of the converted vector from *2-D* into *1-D*. These operations are computed for each position (i, j) in the source image where $1\leq i<(p-n)$, $1\leq j<(q-n)$. Then

Table 1: Steps and pseudo code of the proposed P-TMVA algorithm

Steps	Algorithm

• Determines a window W (i, j) of size m×n using Eq. (2)
• Converts that window into 1-D vector, called NW
• Calculates the likeness between template image and each corresponding window in the source by measuring the sum of square difference distance between NT and NW as in Eq. (3)
• Stores these distances into new storage C (i, j) for each row where 1≤ j< (q-n)
	NT = Convert (T);
	for i = 1, …p-m do
	for j = 1 … q-n do in parallel
	W = Create_W (S);
	NW = Convert (W);
	C [i, j] = SSD (NT, NW, W, T);
	End for
	End for

the computational costs of the proposed algorithm *TMVA* is O (*n* (*p-n+1*) (*q-n+1*)) and this justifies why the proposed method outperform the others.

Parallel template matching vectoring algorithm: The efficiency of a parallel algorithm, however, depends not only on the number of processors but also on the computation method and the inter-processor communications. In a Single-Instruction and Multiple-Data (SIMD) computer, the computation time is the same for each processing element. This study implements a Parallel Template Matching Vectoring Algorithm (*P-TMVA*) for template matching on *SIMD* multi-cores machine. Template matching tasks can be averagely assigned to different cores in order to balance the computations as far as possible. After the multi-core processor completes the computation, the data can be stored into the shared memory for direct read later and the registration computation. The *P-TMVA* has a master thread that creates a number of parallel threads. These threads will be assigned to the available cores. First the master thread converts the template image *T* (*i*, *j*) into*1-D* vector *NT* using Eq. (1). Second, both the master thread and the other threads perform the following steps in parallel to calculate each row *C* (*i*, *j*) where *1*≤*j*< (*q-n*) as shown in Table 1.

The computational cost of the proposed *P-TMVA* algorithm is O (*n* (*p-n+1*) (*q-n+1*) /*k*) where *k* is the number of cores. Next section presents the performed experiments to evaluate the efficiency of the proposed *TMVA* and *P-TMVA* algorithms against conventional template matching algorithms.

EXPERIMENTAL RESULTS

To explore the effectiveness of the proposed algorithm, this study implemented and carried out sequential and parallel experiments to examine the processing time for four different algorithms the proposed vectoring algorithm, the full-search *NCC* algorithm, *NCC* Pyramid (NCCP) algorithm and Sum of Absolute Difference (SAD) algorithm.

Setup: Theses algorithms were implemented using Microsoft Visual Studio Professional 2012 on a HP server (ProLiant ML350p Gen8) with two Intel Xeon (R) processors (E5-2620 @ 2.00 GHz), each processor has 6 cores and 32 GB RAM. The total number of physical cores is 12. Hence, it is capable of running 12 threads simultaneously. The multi core CPU

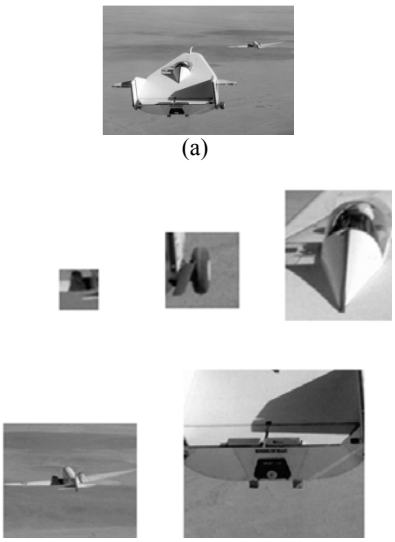
(a)

(b)

Fig. 2: (a) Original greens image (b) the cropped template images form the original with sizes varying from 25×25 to 150×200 pixels

implementation was performed using the *OpenMP* programming model as in Pacheco (2011). The matching operations for each template were distributed among the multiple CPU threads. The number of threads is determined by the maximum number of physical cores in the system. Two types of images are used for the testing purpose color images and gray scale images. Greens image of size 300×500 is a representative for color case as shown in Fig. 2a. However, Fig. 3a shows a gray scale case called Lifting-body of size 512×512. In our experiments we cropped the templates as shown in Fig. 2b and 3b from the source images, so we know in advance the correct position for template in the source image. The size of these templates is varying from 25×25 to 200×200 pixels.

Results: In the clean data, the *TMVA*, *P-TMVA* and compared algorithms are guaranteed to find the correct match from the source image, so we only focus on the comparison of search time required for these algorithms. In the *TMVA* algorithm, the similarity between template and sub-windows in the source image can be measure using more than one similarity function. For example the Euclidean distance or sum of absolute difference or sum of square difference can be used. The



(a) (b)

Fig. 3: (a) Original greens image (b) the cropped template images form the original with sizes varying from 25×25 to 200×200 pixels

mathematical formula for sum of square difference is used in our method Eq. (3). To apply the Euclidean distance and sum of absolute difference functions in our proposed TMVA algorithm their mathematical formula is given by the following equations, respectively:

$$C(i,j)=\sqrt{\left(\sum_{k=i}^{m+i-1}W(k,j)-\sum_{i=1}^{m}T(i,1)\right)^2+\left(\sum_{k=i}^{m+i-1}W(k,j+1)-\sum_{i=1}^{m}T(i,2)\right)^2\ldots+\left(\sum_{k=i}^{m+i-1}W(k,n+j-1)-\sum_{i=1}^{m}T(i,n)\right)^2} \quad (7)$$

and

$$C(i,j)=\left|\sum_{k=i}^{m+i-1}W(k,j)-\sum_{i=1}^{m}T(i,1)\right|+\left|\sum_{k=i}^{m+i-1}W(k,j+1)-\sum_{i=1}^{m}T(i,2)\right|\ldots+\left|\sum_{k=i}^{m+i-1}W(k,n+j-1)-\sum_{i=1}^{m}T(i,n)\right| \quad (8)$$

Figure 4a to d shows a comparison among NCC, NCCP, SAD and the proposed algorithm TMVA for five templates with different sizes varied from 25×25 to 200×200. Figure 5a and b show that the processing time of these algorithms in sequential for greens and lifting-body images respectively. Clearly, they show that the proposed algorithm TMVA is performing the best. Similarly, Fig. 5c and d show that the processing time of the algorithm running in parallel over 12 multi cores for two different images.

As a result, Fig. 4 shows that parallel algorithms speed up their processing time around 4-8 times.

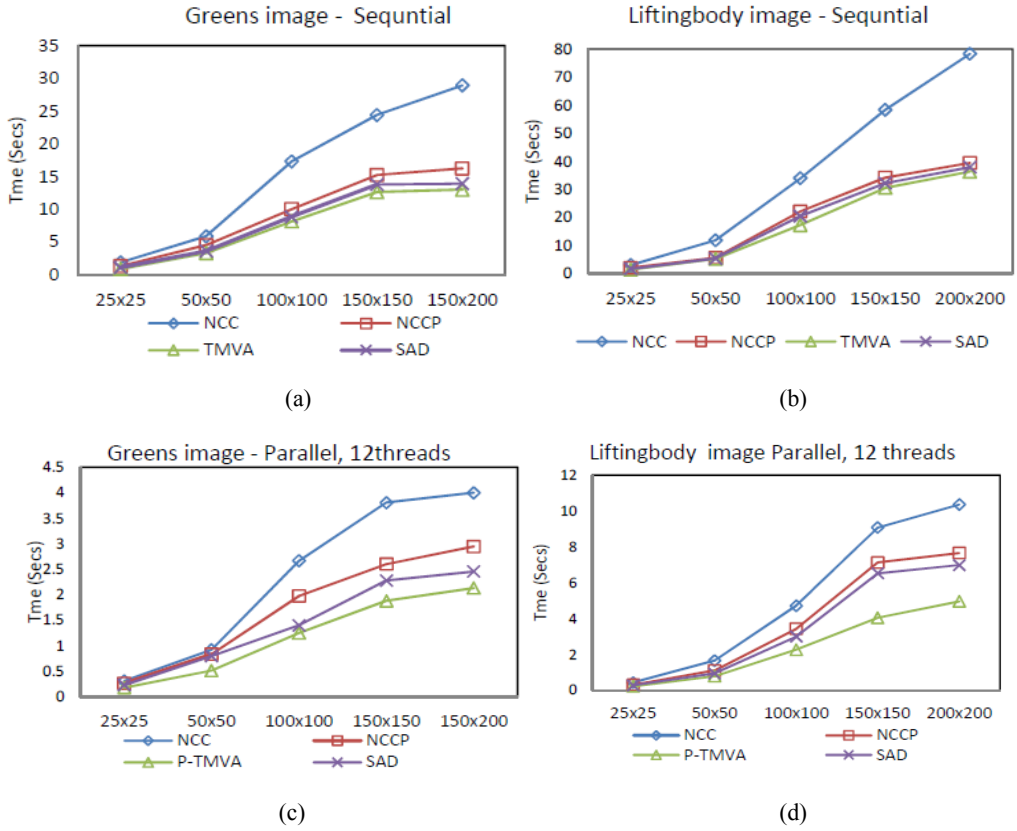


Fig. 4: Comparison between proposed TMVA and P-TMVA algorithms and other conventional algorithms

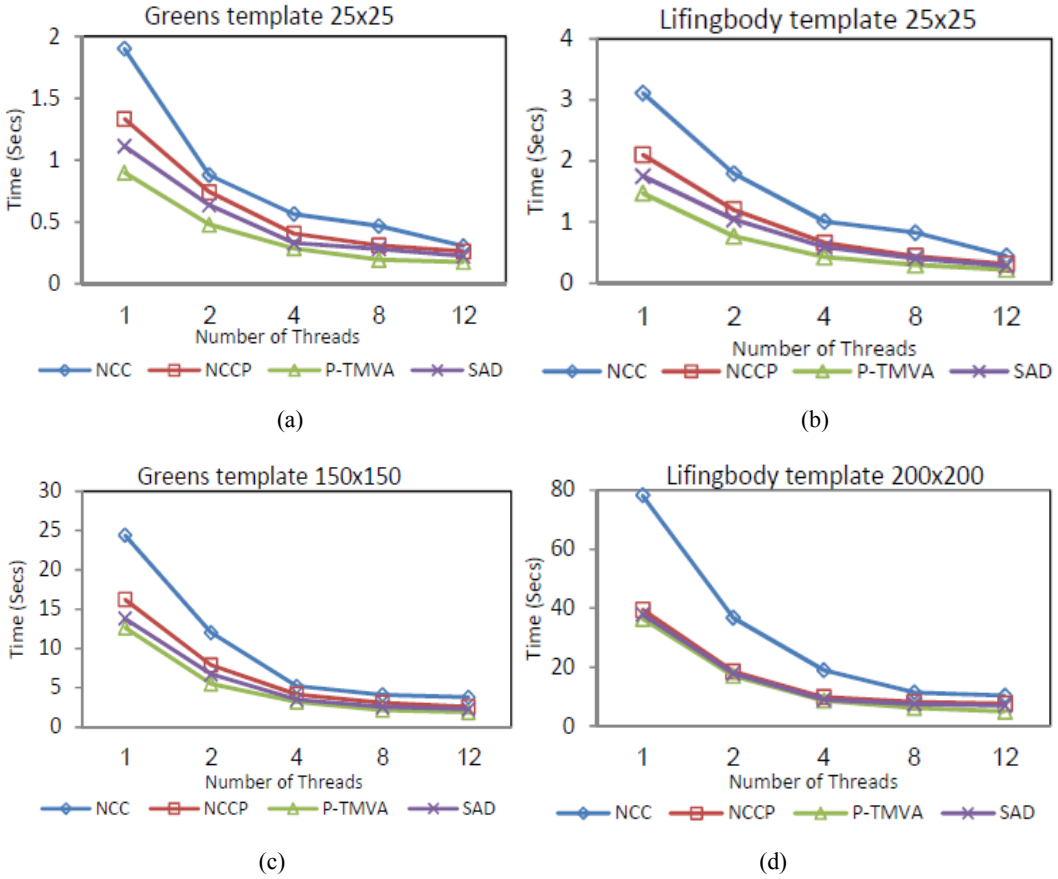


Fig. 5: Proposed P-TMVA algorithm and other conventional algorithms in parallel using different number of threads

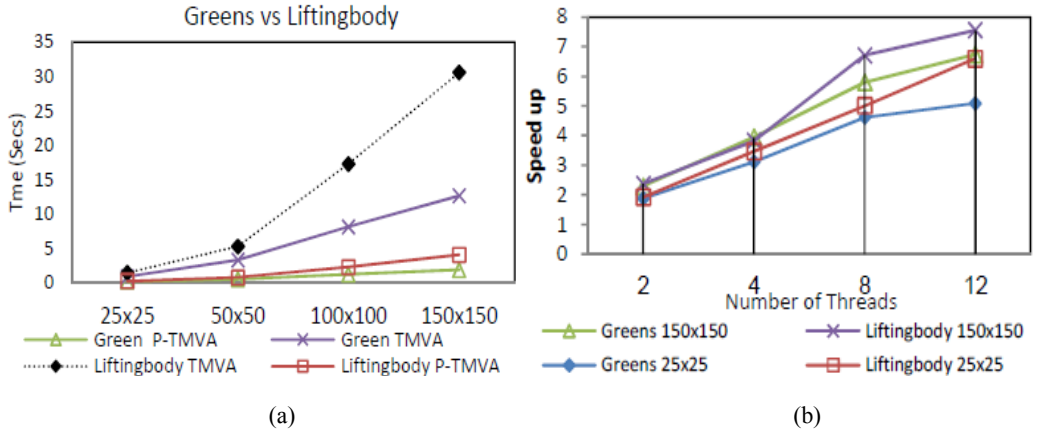


Fig. 6: Comparison of TMVA and P-TMVA with two source images

Moreover, it is observed from Fig. 5 that as the size of the templates is increasing the parallel algorithms illustrate increasing in performance. Moreover, it depicts the improvement of the parallel algorithms in compared to the sequential ones. That is caused by the great time saved with the parallel image reading and processing methods.

Multi core CPU implementations were tested with different numbers of threads, ranging from 1 to 12 as shown in Fig. 5. Speedups of up to 7.56 were achieved. We observed linearly increasing speedups until the number of threads reached the number of cores in the system as shown in Fig. 6b. Figure 5a and b show that the processing time of the algorithms running as parallel threads over 1-12 multi-cores for two different images with small 25×25 template size. Similarly, Fig. 5c and d show that the processing time of the algorithms running as parallel threads over 1-12 multi-

cores for two different images with 150×150 template size. In Fig. 6, it is noticed that while the number of threads is increasing the processing time is decreasing. Moreover, it is noticed that the processing time of the *P-TMVA* algorithm is the shortest in compared with the other algorithms for all templates. Since the complexity of the proposed is more efficient than the other methods.

Figure 4 and 5 shows the performance of the proposed parallel algorithm compared with other four algorithms using greens image (color case) and lifting-body (gray scale case) respectively. It is clear that the proposed algorithm outperform the others for all template sizes. It is a superior to the other basic template matching algorithms. In the above experiments, the correct match position is assumed to be the position where the minimum similarity distance value is obtained when the entire template is used in the search process.

Moreover, Fig. 6a shows that the size of the source images and templates are significant factors for the processing time of both the proposed parallel algorithm *P-TMVA* and the proposed sequential algorithm *TMVA*. They show better performance in greens image with size 300×500 than lifting body image with size 512×512. Figure 6b depicts that the speedup of the *P-TMVA* is increasing as the source image and template size are increasing.

CONCLUSION

To speed up the computation of block matching while still guaranteeing the correct match for template in the source this study proposed a new Template Matching Vectoring Algorithm (TMVA). It converted the template and each corresponding block in the source from 2-D into 1-D. The *TMVA* has been applied to the template matching using three different similarity measures and shown a reduction in computation time. Moreover, to speed up the proposed algorithm *TMVA*, this study implemented it as parallel algorithm *P-TMVA*. Two different types of image (color and gray scale) were used for comparison between proposed *TMVA* algorithm and other conventional algorithms. The templates were cropped from the source images. The experimental results demonstrate the efficiency of the proposed algorithm for pattern matching under uniform illumination.

ACKNOWLEDGMENT

This study is financially supported by the Scientific Research Deanship, King Faisal University under grant number SDN-150166.

REFERENCES

Alsaade, F. and Y. Fouda 2012. Template matching based on SAD and pyramid. Int. J. Comput. Sci. Inform. Secur., 10(4): 11-16.

Alsaade, F., Y. Fouda and A.R. Khan, 2012. Efficient cellular automata algorithm for template matching. J. Artif. Intell., 5(3): 122-129.

Anderson, R.F., J.S. Kirtzic and O. Daescu, 2010. Applying parallel design techniques to template matching with GPUs. Proceeding of 9th International Conference on High Performance Computing for Computational Science (VECPAR'10), pp: 456-468.

Armstrong, J.B., A. Maheswaran, M.D. Theys, M.A. Nichols and K.H., 1998. Casey: Parallel image correlation: Case study to examine trade-offs in algorithm-to-machine mappings. J. Supercomput., 12: 7-35.

Brunelli, R., 2009. Template Matching Techniques in Computer Vision: Theory and Practice. John Wiley and Sons Ltd., ISBN: 978-0-470-51706-2.

Chen, Y.S., Y.P. Huang and C.S. Fuh, 2001. A fast block matching algorithm based on the winner-update strategy. IEEE T. Image Process., 10(8): 1212-1222.

Costa, C.E. and M. Petrou, 2000. Automatic registration of ceramic tiles for the prop use of fault detection. Mach. Vision Appl., 11: 225-230.

Du-Ming, T. and L. Chien-Ta, 2003. Fast normalized cross correlation for detect detection. Pattern Recogn. Lett., 24: 2625-2631.

Essannouni, F., R.O.H. Thami, D. Aboutajdine and A. Salam, 2007. Adjustable SAD matching algorithm using frequency domain. J. Real-Time Image Proc., 1(4): 257-265.

Fang, Z., X. Li and L.M. Ni, 1985. Parallel algorithms for image template matching on hypercube SIMD computers. Proceeding of the IEEE Workshop on Computer Architecture for Pattern Analysis and Image Database Management, pp: 33-40.

Horng, S.J., W.T. Chen and M.Y. Fang, 1991. Optimal speed-up algorithms for template matching on SIMD hypercube multiprocessors with restricted local memory. Inform. Process. Lett., 38(1): 29-37.

Jenq, J.F. and S. Sahni, 1991. Reconfigurable mesh algorithms for image shrinking, expanding, clustering and template matching. Proceeding of the International Parallel Processing Symposium, pp: 208-215.

Kidorf, H. and W. Peigorsch, 1984. A practical Fast Fourier Transform (FFT)-based implementation for image correlation. Proceeding of SPIE 0504, Applications of Digital Image Processing VII, pp: 135.

Kumar, V.K.P. and V. Krishnan, 1987. Efficient image template matching on hypercube SIMD arrays. Proceeding of International Conference on Processing, pp: 765-771.

Li, R., B. Zeng and M.L. Liou, 1994. A new three-step search algorithm for block motion estimation. IEEE T. Circ. Syst. Vid., 4(4): 438-442.

Mahmood, A. and S. Khan, 2012. Correlation coefficient based fast template matching through partial elimination. IEEE T. Image Process., 21(4): 2099-2108.

Mendez, J., J. Lorenzo and M. Castrillon, 2011. Comparative performance of GPU, SIMD and OpenMP systems for raw template matching in computer vision. Proceeding of 19th International Conference on Computer Graphics, Visualization and Computer, pp: 9-18.

Mikhail, I.A., 2001. Faster image template matching in the sum of the absolute value of differences measures. IEEE T. Image Process., 10(2): 659-663.

Niblack, W., 1986. An Introduction to Digital Image Processing. Prentice-Hall International, Englewood Cliffs, N.J.

Pacheco, P., 2011. An Introduction to Parallel Programming. 1st Edn., Morgan Kaufmann, Amsterdam, Boston.

Ranganathan, P., S. Adve and N.P. Jouppi, 1999. Performance of image and video processing with general-purpose processors and media ISA extensions. Proceeding of 26th Annual International Symposium on Computer Architecture (ISCA'99).

Ranka, S. and S. Sahni, 1988. Image template matching on SIMD hypercube computers. Proceeding of International Conference on Parallel Process, pp: 84-91.

Wei, S. and S. Lai, 2008. Fast template matching based on normalized cross correlation with adaptive multilevel winner update. IEEE T. Image Process., 17(11): 2227-2235.

Zitova, B.F., 2003. Image registration methods: A survey. Image Vision Comput., 21(11): 977-1000.