World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:1, No:11, 2007

A Modified Cross Correlation in the Frequency Domain for Fast Pattern Detection Using Neural Networks

Hazem M. El-Bakry and Qiangfu Zhao

***Abstract*—**Recently, neural networks have shown good results for detection of a certain pattern in a given image. In our previous papers [1-5], a fast algorithm for pattern detection using neural networks was presented. Such algorithm was designed based on cross correlation in the frequency domain between the input image and the weights of neural networks. Image conversion into symmetric shape was established so that fast neural networks can give the same results as conventional neural networks. Another configuration of symmetry was suggested in [3,4] to improve the speed up ratio. In this paper, our previous algorithm for fast neural networks is developed. The frequency domain cross correlation is modified in order to compensate for the symmetric condition which is required by the input image. Two new ideas are introduced to modify the cross correlation algorithm. Both methods accelerate the speed of the fast neural networks as there is no need for converting the input image into symmetric one as previous. Theoretical and practical results show that both approaches provide faster speed up ratio than the previous algorithm.

***Keywords*—**Fast Pattern Detection, Neural Networks, Modified Cross Correlation

I. INTRODUCTION

Pattern detection is a fundamental step before pattern recognition. Its reliability and performance have a major influence in a whole pattern recognition system. Nowadays, neural networks have shown very good results for detecting a certain pattern in a given image [6,9,11,12]. But the problem with neural networks is that the computational complexity is very high because the networks have to process many small local windows in the images [8,10].

Manuscript received May 3, 2004.

H. M. El-Bakry, is assistant lecturer with Faculty of Computer Science and Information Systems – Mansoura University – Egypt. Now, he is PhD student in University of Aizu, Aizu Wakamatsu, Japan 965-8580 (phone +81-242-37-2519, Fax. +81-242-37-2743, E-mail: helbakry20@yahoo.com).

Q. Zhao is professor with the Information Systems Department, University of Aizu, Japan (e-mail: qf-zhao@u-aizu.ac.jp).

In our previous papers [1-5], a fast algorithm for pattern detection using neural networks was presented. Such algorithm was designed based on cross correlation in the frequency domain between the input image and the weights of neural networks. In [5], practical realization using MATLAB proved that a symmetry condition is necessary and must be found either in the input image or in the neural weights so that those fast neural networks can give the same correct results as conventional neural network for detecting a certain pattern in a given image. In our previous papers [3,4], we succeeded in improving the speed up of the detection process by converting the input image into a symmetric form with less dimensions compared with the old symmetric image introduced in [5]. Mathematical analysis and simulation results for this symmetric configuration proved that the number of computation steps required by fast neural networks is reduced. Although the speed up ratio is increased, the symmetric condition still causes the fast neural networks to consume a large number of computation steps. This is because image conversion into symmetric form increases the dimensions of the input image.

Here, the cross correlation, implemented in the frequency domain, is modified in order to compensate for the symmetric condition which is required in the input image. Two approaches are presented in order to compensate for the symmetric condition. Fast neural networks for pattern detection are described in section II. In section III, the previous symmetric configuration for the input image to speed up the detection process is discussed. Two new ideas to compensate for this symmetry condition are presented.

II. THEORY OF FAST NEURAL NETWORKS BASED ON CROSS CORRELATION IN THE FREQUENCY DOMAIN FOR PATTERN DETECTION

Finding a certain pattern in the input image is a search problem. Each subimage in the input image is tested for the presence or absence of the required pattern. At each pixel position in the input image each subimage is multiplied by a

window of weights, which has the same size as the subimage. The outputs of neurons in the hidden layer are multiplied by the weights of the output layer. A high output implies that the tested subimage contains the required pattern and vice versa. Thus, we may conclude that this searching problem is cross correlation between the image under test and the weights of the hidden neurons.

The convolution theorem in mathematical analysis says that a convolution of f with h is identical to the result of the following steps: let F and H be the results of the Fourier Transformation of f and h in the frequency domain. Multiply F and H in the frequency domain point by point and then transform this product into the spatial domain via the inverse Fourier Transform. As a result, these cross correlations can be represented by a product in the frequency domain. So, by using cross correlation in the frequency domain, speed up in an order of magnitude can be achieved during the detection process [1,2,3,4,5,6,7,9].

In the detection phase, a sub image I of size mxn (sliding window) is extracted from the tested image, which has a size PxT, and fed to the neural network. Let X_i be the vector of weights between the input sub image and the hidden layer. This vector has a size of mxn and can be represented as mxn matrix. The output of hidden neurons h_i can be calculated as follows:

$$h_i = g\left(\sum_{j=1}^{m} \sum_{k=1}^{n} X_i(j,k) I(j,k) + b_i \right) \quad (1)$$

where g is the activation function and b_i is the bias of each hidden neuron (i). Eq.1 represents the output of each hidden neuron for a particular subimage I. It can be obtained from image Z as follows:

$$h_i(u,v) = g\left(\sum_{j=-m/2}^{m/2} \sum_{k=-n/2}^{n/2} X_i(j,k)\, Z(u+j, v+k) + b_i \right) \quad (2)$$

Eq.2 represents a cross correlation operation. Given any two functions f and d, their cross correlation can be obtained by [1]:

$$f(x,y) \otimes g(x,y) = \left(\sum_{m=-\infty}^{\infty} \sum_{n=-\infty}^{\infty} f(x+m, y+n) g(m,n) \right) \quad (3)$$

Therefore, Eq.2 can be written as follows [1]:

$$h_i = g\left(Z \otimes X_i + b_i \right) \quad (4)$$

where h_i is the output of the hidden neuron (i) and $h_i(u,v)$ is the activity of the hidden unit (i) when the sliding window is located at position (u,v) and $(u,v) \in [P-m+1, T-n+1]$.

Now, the above cross correlation can be expressed in terms of the Fourier Transform:

$$Z \otimes X_i = F^{-1}\left(F(Z) \bullet F^{*}\left(X_i \right) \right) \quad (5)$$

Hence, by evaluating this cross correlation, a speed up ratio can be obtained comparable to conventional neural networks. Also, the final output of the neural network can be evaluated as follows:

$$O(u,v) = g\left(\sum_{i=1}^{q} w_o(i)\, h_i(u,v) + b_o \right) \quad (6)$$

where q is the number of neurons in the hidden layer. O(u,v) is the output of the neural network when the sliding window is located at the position (u,v) in the input image Z.

The complexity of cross correlation in the frequency domain can be analyzed as follows [5]:

1- For a tested image of NxN pixels, the 2D-FFT requires a number equal to $O(N^2 \log_2 N^2)$ of complex computation steps. Also, the same number of complex computation steps is required for computing the 2D FFT of the weight matrix for each neuron in the hidden layer.

2- At each neuron in the hidden layer, the inverse 2D FFT is computed. So, q backward and (1+q) forward transforms have to be computed. Therefore, for an image under test, the total number of the 2DFFT to compute is $O((2q+1)N^2 \log_2 N^2)$.

3- The input image and the weights should be multiplied in the frequency domain. Therefore, a number of complex computation steps equal to $O(qN^2)$ should be added.

4- The number of computation steps required by fast neural networks is complex and must be converted into a real version. It is known that the two dimensions Fast Fourier Transform requires $O((N^2/2)\log_2 N^2)$ complex multiplications and $O(N^2 \log_2 N^2)$ complex additions. Every complex multiplication is realized by six real floating point operations and every complex addition is implemented by two real floating point operations. So, the total number of computation steps required to obtain the 2D-FFT of an NxN image is [5]:

$$\rho = O(6((N^2/2)\log_2 N^2) + 2(N^2 \log_2 N^2)) \quad (7)$$

which may be simplified to:

$$\rho = O(5N^2 \log_2 N^2) \quad (8)$$

Performing complex dot product in the frequency domain also requires $O(6qN^2)$ real operations.

5- In order to perform cross correlation in the frequency domain, the weight matrix must have the same size as the input image. So, a number of zeros = (N^2-n^2) must be added to the weight matrix. This requires a total real number of computation steps = $O(q(N^2-n^2))$ for all neurons. Moreover, after computing the FFT2 for the weight matrix, the conjugate of this matrix must be obtained. So, a real number of computation steps $=O(qN^2)$ should be added in order to obtain the conjugate of the weight matrix for all neurons. Also, a number of real computation steps equal to O(N) is required to create butterflies complex numbers ($e^{-jk(2\Pi n/N)}$), where 0<K<L. These (N/2) complex numbers are multiplied by the elements of the input image or by previous complex numbers during the computation of FFT2. To create a complex number requires two real floating point operations. So, the total number of computation steps required by fast neural networks becomes [5]:

$$\sigma=O((2q+1)(5N^2\log_2N^2)+6qN^2+q(N^2-n^2)+qN^2+N) \quad (9)$$

which can be reformulated as:

$$\sigma=O((2q+1)(5N^2\log_2N^2)+q(8N^2-n^2)+N) \quad (10)$$

6- Using a sliding window of size nxn for the same image of NxN pixels, $O(q(2n^2-1)(N-n+1)^2)$ computation steps are required when using traditional neural networks for pattern detection process. The theoretical speed up factor η can be evaluated as follows [5]:

$$\eta=O\left(\frac{q(2n^2-1)(N-n+1)^2}{(2q+1)(5N^2\log_2N^2)+q(8N^2-n^2)+N}\right) \quad (11)$$

7- But as proved in [5], this cross correlation in the frequency domain (Fast Neural Networks) gives the same results as conventional cross correlation (Conventional Neural Networks) only in two cases. Either the weights are symmetric or the input image is symmetric. It is very complex to allow the weights to be symmetric in the required form which needs to be as follows [5]:

$$W=\begin{bmatrix} w & 0 \\ 0 & w_d \end{bmatrix} \quad (12)$$

Adding this constraint to the learning rules will cause many well known problems during the training process of the neural network. Another solution is to convert the input image into one of the required symmetric forms as shown in Fig. 1. As the input image has a dimension of (N), the new symmetric image will have a length of (2N). In this case, the number of computation steps required by fast neural networks can be calculated as follows [5]:

$$\sigma_{2N}=O((2q+1)(5(2N)^2\log_2(2N)^2)+q(8(2N)^2-n^2)+2N) \quad (13)$$

But, converting the non-symmetric input image into a symmetric one will slow down the proposed fast neural networks more compared to conventional neural networks. In this case, for any size of the input image, dividing the number of operations required for conventional neural networks by those needed by fast neural networks (Eq. 11) gives a lower speed up ratio than the one listed in Table 1 [5].

III. A Modified Cross Correlation Algorithm for Fast Neural Networks

In this section, a modification in the cross correlation function, implemented in the frequency domain, is established by two methods in order to improve the speed of the fast neural networks. In [3,4], the symmetric form shown in Fig. 2 for the input image was presented. This form reduced the number of computation steps required by fast neural networks for pattern detection. The input image was converted into symmetric form by rotating it 180 degrees. Then, both the up and down images were tested as one (symmetric) image consisting of two images. In this case, this symmetric image has (2NxN) dimensions. By substituting in Eq. 9 for the these dimensions, the number of computation steps required for cross correlating this symmetric image with the weights in the frequency domain can be calculated as follows [3,4]:-

$$\sigma=O((2q+1)(5(2N^2\log_2N+2N^2\log_22N))+q6(2N^2)+q(2N^2-n^2)+q(2N^2)+2N) \quad (14)$$

which can be simplified to:

$$\sigma=O((2q+1)(10N^2(\log_22N+\log_2N))+q(16N^2-n^2)+2N) \quad (15)$$

So, the speed up ratio in this case can be calculated as:

$$\eta=O\left(\frac{q(2n^2-1)(N-n+1)^2}{(2q+1)(10N^2(\log_22N+\log_2N))+q(16N^2-n^2)+2N}\right) \quad (16)$$

The theoretical speed up ratio in this case with different sizes of the input image and different in size weight matrices is listed in Table 2. Practical speed up ratio for manipulating images of different sizes and different in size weight matrices is listed in Table 3 using 700 MHz processor and MATLAB.

Here, two methods are presented to compensate for this symmetry condition by modifying the cross correlation implemented in the frequency domain.

A) The First Method

Modification in cross correlation is performed by rotating the input image down and computing the Fast Fourier Transform in two dimensions for the rotated version. Then, the conjugate matrix of the resulted matrix is obtained. The conjugate matrix of the Fast Fourier Transform in two dimensions for the weight matrix is also calculated. This method provides a correct result as conventional neural networks when cross correlation is done in the frequency domain. Using this procedure, there is no need to convert either the input image or the weights into symmetric form. In this case, Eq. 10 can be written as follows:

$$\sigma=O((2q+1)(5N^2\log_2N^2)+q(8N^2-n^2)+2N^2+N) \quad (17)$$

The term $2N^2$ is added to the number of computation steps required by fast neural networks (Eq. 10). N^2 computation steps are required for image rotation into down direction. The other N^2 computation steps corresponds to the computation of the conjugate matrix of the Fast Fourier Transform for the input image in two dimensions.

The speed up ratio can be calculated as follows:

$$\eta=O\left(\frac{q(2n^2-1)(N-n+1)^2}{(2q+1)(5N^2\log_2N^2)+q(8N^2-n^2)+2N^2+N}\right) \quad (18)$$

The theoretical speed up ratio in this case with different sizes of the input image and different in size weight matrices is listed in Table 4. Also, practical speed up ratio for manipulating images of different sizes and different in size weight matrices is listed in Table 5 using 700 MHz processor and MATLAB.

On the other hand, instead of rotating the input image, the weight matrix can be rotated into down direction. Therefore, qN^2 operations are added to the total number of computation steps required for Fast Neural Networks. But the final matrix resulted at the output of neural network should be rotated into down direction. This adds N^2 operations to the number computation steps required by Fast Neural Networks. The conjugate of the Fast Fourier Transform for the input image is automatically computed in the dot function. Thus, N^2 operations are added to the number computation steps required by Fast Neural Networks. In this case, Eq. 10 can be written as follows:

$$\sigma=O((2q+1)(5N^2\log_2N^2)+q(9N^2-n^2)+2N^2+N) \quad (19)$$

The speed up ratio can be calculated as follows:

$$\eta=O\left(\frac{q(2n^2-1)(N-n+1)^2}{(2q+1)(5N^2\log_2N^2)+q(9N^2-n^2)+2N^2+N}\right) \quad (20)$$

B) The Second Method

The problem is that the MATLAB dot function computes the conjugate of the input image automatically which is not needed to perform cross correlation in the frequency domain. Therefore, we have to remove the effect of computing this conjugate matrix so that fast neural networks can give the same results as conventional neural networks. The idea is to compensate for the symmetry condition by performing a permutation between the input image and the weight matrix. In this case, there is no need to compute the conjugate of the weight matrix as it will be obtained automatically during the computation of the dot function. The dot function in MATLAB computes the conjugate of the first matrix automatically before performing the dot multiplication (See Appendix A). So, a permutation is made between the first and the second matrix before the dot multiplication is started. In this case, the first matrix after permutation will be the weight matrix. By performing this permutation and applying the dot multiplication, there is no need to convert the input image into symmetric one. Furthermore, there is no need to rotate either the input image or the weight matrix. Moreover, there is no need to compute the conjugate matrix of the Fast Fourier Transform either for input image or the weight matrix as the conjugate matrix of the Fast Fourier Transform for the weight matrix is computed automatically through the dot function. In this case, the total number of computation steps required by fast neural networks is:

$$\sigma=O((2q+1)(5N^2\log_2N^2)+q(8N^2-n^2)+N) \quad (21)$$

and the speed up ratio can be calculated as follows:

$$\eta=O\left(\frac{q(2n^2-1)(N-n+1)^2}{(2q+1)(5N^2\log_2N^2)+q(8N^2-n^2)+N}\right) \quad (22)$$

The theoretical speed up ratio in this case with different sizes of the input image and different in size weight matrices is listed in Table 6. Also, practical speed up ratio for manipulating images of different sizes and different in size weight matrices is listed in Table 7 using 700 MHz processor and MATLAB. It is clear that the results listed in Tables 6 and 7 are the same as those listed in Tables 4 and 5. This is because the effect of the term $2N^2$ on the number of computation steps in Eq. 17 is very small. So, the speed up ratio given by Eq. 18 is approximately the same as that in Eq. 22.

Note that, in all of these experiments the Fast Fourier Transform of the weight matrix as well as all related computation for the weights (such as rotation and conjugate

of the Fast Fourier Transform) can be rotated off line before starting the test phase. Thus, all the calculations and terms corresponding to the weight matrix can be removed. Although, this reduces the number of computation steps required by fast neural networks, a restriction on the input image to be with fixed dimensions is added.

VI. Conclusion

Two new approaches for fast neural networks have been presented. The number of computation steps required by both methods has been proved to be less than the previous method. This has been accomplished by modifying the frequency domain cross correlation algorithm implemented in MATLAB. By using these new methods, there is no need for image conversion into symmetric shape as presented in our previous work. Simulation results have shown that both methods give the same speed up ratio which is faster than the previous one.

Appendix "A"

For the "dot(a,b)" function implemented in MATLAB, when dealing with complex numbers (i.e. a and b are complex numbers), this dot function provides different result unless the conjugate of the first number "a" is computed at the first. The next is an example for computing the dot product of two complex numbers (a and b) and executed at the MATLAB dot prompt:

```
» a=1-2i
a =
   1.0000 - 2.0000i
» b=4-3i
b =
   4.0000 - 3.0000i
» c=a*b
c =
  -2.0000 -11.0000i
» d=dot(a,b)
d =
  10.0000 + 5.0000i
» e=conj(a)
e =
   1.0000 + 2.0000i
» dnew=dot(e,b)
dnew =
  -2.0000 -11.0000i
```

The correct result "c" is different from "d=dot(a,b)". Unless we compute "e=conj(a)", and then perform dot(e,b). The result is "dnew=c". The same error occurred when "a" and "b" are matrices "dot(a,b,dim)", unless the conjugate of matrix "a" is computed before the dot multiplication is performed.

References

[1] Hazem M. El-Bakry, and Qiangfu Zhao, "Fast Sub-Matrix Detection Using Neural Networks and Cross Correlation in the Frequency Domain," Second Workshop of Tohoku Branch, IPSJ, (Information Processing Society of Japan), University of Aizu, Aizuwakamatsu, Japan, Jan. 21, 2005.

[2] Hazem M. El-Bakry, and Qiangfu Zhao, "Fast Object/Face Detection Using Neural Networks and Fast Fourier Transform," the International Journal of Signal Processing, vol.1, no.3, pp. 182-187, 2004.

[3] H. M. El-Bakry, and Qiangfu Zhao, "A New Symmetric Form for Fast Sub-Matrix (Object/Face) Detection Using Neural Networks and FFT," under publication in the International Journal of Signal Processing.

[4] Hazem M. El-Bakry, and Qiangfu Zhao, "Fast Pattern Detection Using Normalized Neural Networks and Cross Correlation in the Frequency Domain," under publication in the European Journal of Applied Signal Processing.

[5] Hazem M. El-Bakry, "Comments on Using MLP and FFT for Fast Object/Face Detection," Proc. of IEEE IJCNN'03, Portland, Oregon, July, 20-24, 2003, pp. 1284-1288.

[6] Hazem M. El-Bakry, "Human Iris Detection Using Fast Cooperative Modular Neural Networks and Image Decomposition," Machine Graphics & Vision Journal (MG&V), vol. 11, no. 4, pp. 498-512, 2002.

[7] Hazem M. El-Bakry, "Face detection using fast neural networks and image decomposition," Neurocomputing Journal, vol. 48, pp. 1039-1046, 2002.

[8] S. Srisuk and W. Kurutach, "A New Robust Face Detection in Color Images," Proc. of IEEE Computer Society International Conference on Automatic Face and Gesture Recognition (AFGR'02), Washington D.C., USA, May 20-21, 2002, pp. 306-311.

[9] Hazem M. El-Bakry, "Automatic Human Face Recognition Using Modular Neural Networks," Machine Graphics & Vision Journal (MG&V), vol. 10, no. 1, pp. 47-73, 2001.

[10] Ying Zhu, Stuart Schwartz, and Michael Orchard, "Fast Face Detection Using Subspace Discriminate Wavelet Features," Proc. of IEEE Computer Society International Conference on Computer Vision and Pattern Recognition (CVPR'00), South Carolina, June 13 - 15, 2000, vol.1, pp. 1636-1643.

[11] R. Feraud, O. Bernier, J. E. Viallet, and M. Collobert, "A Fast and Accurate Face Detector for Indexation of Face Images," Proc. of the Fourth IEEE International Conference on Automatic Face and Gesture Recognition, Grenoble, France, 28-30 March, 2000.

[12] S. Baluja, H. A. Rowley, and T. Kanade, "Neural Network - Based Face Detection," IEEE Trans. on Pattern Analysis and Machine Intelligence, vol. 20, no. 1, pp. 23-38, 1998.

TABLE 1
A COMPARISON BETWEEN THE NUMBER OF COMPUTATION STEPS (IN MILLIONS) REQUIRED FOR CONVENTIONAL AND FAST NEURAL NETWORKS TO MANIPULATE IMAGES SHOWN IN FIG.1 WITH DIFFERENT SIZES (N=20).

Image size	Conventional Neural Networks	Fast Neural Networks (2N)	Speed up ratio
100x100	157,267170	196,098091	.802
200x200	785,281170	882,028364	.890
300x300	1892,695170	2113,036584	.896
400x400	3479,509170	3918,549456	.888
500x500	5545,723170	6319,116413	.877
600x600	8091,337170	9330,582337	.867
700x700	11116,351170	12965,856005	.857
800x800	14620,765170	17235,856005	.848
900x900	18604,579170	21986,745146	.846
1000x1000	23067,793170	27716,501654	.832

TABLE 2
THE THEORETICAL SPEED UP RATIO IN CASE OF CONVERTING AN IMAGE INTO SYMMETRIC ONE THROUGH ROTATION INTO DOWN DIRECTION.

Image size	Speed up ratio (n=20)	Speed up ratio (n=25)	Speed up ratio (n=30)
100x100	1.71	2.36	2.96
200x200	1.88	2.79	3.79
300x300	1.89	2.85	3.96
400x400	1.86	2.85	3.99
500x500	1.84	2.82	3.98
600x600	1.82	2.80	3.96
700x700	1.80	2.77	3.93
800x800	1.78	2.74	3.90
900x900	1.76	2.72	3.87
1000x1000	1.74	2.70	3.84

TABLE 3
SIMULATION RESULTS FOR SPEED UP RATIO IN CASE OF CONVERTING AN IMAGE INTO SYMMETRIC ONE THROUGH ROTATION INTO DOWN DIRECTION.

Image size	Speed up ratio (n=20)	Speed up ratio (n=25)	Speed up ratio (n=30)
100x100	5.29	6.74	11.16
200x200	4.46	6.24	9.94
300x300	4.17	5.08	8.66
400x400	3.59	4.78	7.45
500x500	3.40	4.34	6.87
600x600	3.30	4.42	6.16
700x700	3.12	4.20	5.74
800x800	2.60	3.58	4.76
900x900	2.97	4.10	5.38
1000x1000	2.57	3.47	4.63

TABLE 4
THE THEORETICAL SPEED UP RATIO IN CASE OF ROTATING THE INPUT IMAGE DOWN AND COMPUTING THE CONJUGATE OF FAST FOURIER TRANSFORM IN TWO DIMENSIONS FOR THE INPUT IMAGE.

Image size	Speed up ratio (n=20)	Speed up ratio (n=25)	Speed up ratio (n=30)
100x100	3.67	5.04	6.34
200x200	4.01	5.92	8.05
300x300	4.00	6.03	8.37
400x400	3.95	6.01	8.42
500x500	3.89	5.95	8.39
600x600	3.83	5.88	8.33
700x700	3.78	5.82	8.26
800x800	3.73	5.76	8.19
900x900	3.69	5.70	8.12
1000x1000	3.65	5.65	8.05

TABLE 5
SIMULATION RESULTS FOR SPEED UP RATIO IN CASE OF ROTATING THE INPUT IMAGE INTO DOWN DIRECTION AND COMPUTING THE CONJUGATE OF FAST FOURIER TRANSFORM IN TWO DIMENSIONS FOR THE INPUT IMAGE.

Image size	Speed up ratio (n=20)	Speed up ratio (n=25)	Speed up ratio (n=30)
100x100	7.88	10.75	14.69
200x200	6.21	9.19	13.17
300x300	5.54	8.43	12.21
400x400	4.78	7.45	11.41
500x500	4.68	7.13	10.79
600x600	4.46	6.97	10.28
700x700	4.34	6.83	9.81
800x800	4.27	6.68	9.60
900x900	4.31	6.79	9.72
1000x1000	4.19	6.59	9.46

TABLE 6
THE THEORETICAL SPEED UP RATIO IN CASE OF MODIFYING THE CROSS CORRELATION ALGORITHM IN THE FREQUENCY DOMAIN BY PERMUTING THE INPUT IMAGE AND THE WEIGHT MATRIX IN DOT FUNCTION.

Image size	Speed up ratio (n=20)	Speed up ratio (n=25)	Speed up ratio (n=30)
100x100	3.67	5.04	6.34
200x200	4.01	5.92	8.05
300x300	4.00	6.03	8.37
400x400	3.95	6.01	8.42
500x500	3.89	5.95	8.39
600x600	3.83	5.88	8.33
700x700	3.78	5.82	8.26
800x800	3.73	5.76	8.19
900x900	3.69	5.70	8.12
1000x1000	3.65	5.65	8.05

TABLE 7
SIMULATION RESULTS FOR SPEED UP RATIO IN CASE OF MODIFYING THE CROSS CORRELATION ALGORITHM IN THE FREQUENCY DOMAIN BY PERMUTING THE INPUT IMAGE AND THE WEIGHT MATRIX IN DOT FUNCTION.

Image size	Speed up ratio (n=20)	Speed up ratio (n=25)	Speed up ratio (n=30)
100x100	7.88	10.75	14.69
200x200	6.21	9.19	13.17
300x300	5.54	8.43	12.21
400x400	4.78	7.45	11.41
500x500	4.68	7.13	10.79
600x600	4.46	6.97	10.28
700x700	4.34	6.83	9.81
800x800	4.27	6.68	9.60
900x900	4.31	6.79	9.72
1000x1000	4.19	6.59	9.46

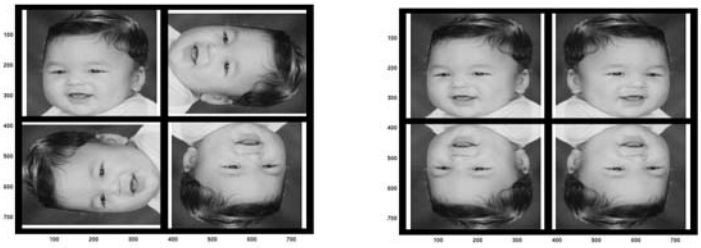

Fig. 1. Image conversion from non-symmetric to symmetric one.

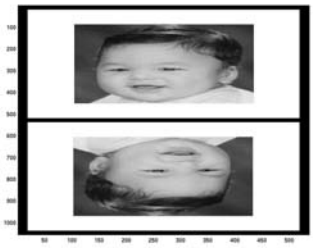

Fig. 2. Image conversion from non-symmetric to symmetric one through rotation into down direction.

Eng. Hazem Mokhtar El-Bakry (Mansoura, EGYPT 20-9-1970) received B.Sc. degree in Electronics Engineering, and M.Sc. in Electrical Communication Engineering from the Faculty of Engineering, Mansoura University – Egypt, in 1992 and 1995 respectively. Since 1997, he has been an assistant lecturer at the Faculty of Computer Science and Information Systems – Mansoura University – Egypt. Currently, he is a doctoral student at the Multimedia device laboratory, University of Aizu - Japan. In 2004, he got a Research Scholarship from Japanese Government based on a recommendation from University of Aizu.

His research interests include neural networks, pattern recognition, image processing, biometrics, cooperative intelligent systems and electronic circuits. In these areas, he has published more than 39 papers as a single author in major international journals and conferences. He is the first author in 8 refereed international journal papers and more than 60 refereed international conference papers.

Eng. El-Bakry has the patent No. 2003E 19442 DE HOL / NUR, Magnetic Resonance, SIEMENS Company, Erlangen, Germany, 2003. He is a referee for the International Journal of Machine Graphics & Vision and many different international conferences. He was selected as a chairman for the Facial Image Processing Session in the 6th International Computer Science Conference, Active Media Technology (AMT) 2001, Hong Kong, China, December 18-20, 2001 and for the Genetic Programming Session, in ACS/IEEE International Conference on Computer Systems and Applications Lebanese American University Beirut, Lebanon, June 25-29, 2001. He was invited for a talk in the Biometric Consortium, Orlando, Florida, USA, 12-14 Sep. 2001, which co-sponsored by the United States National Security Agency (NSA) and the National Institute of Standards and Technology (NIST).

Dr. Zhao received the Ph. D degree from Tohoku University of Japan in 1988. He joined the Department of Electronic Engineering of Beijing Institute of Technology of China in 1988, first as a post doctoral fellow and then associate professor. He was associate professor from Oct. 1993 at the Department of Electronic Engineering of Tohoku University of Japan. He joined the University of Aizu of Japan from April 1995 as associate professor, and became tenure full professor in April 1999. Prof. Zhao research interests include image processing, pattern recognition and understanding, computational intelligence, neurocomputing and evolutionary computation.