Fast Image Matching on Web Pages

HAZEM M. EL-BAKRY

Faculty of Computer Science & Information Systems,
Mansoura University, EGYPT
E-mail: helbakry20@yahoo.com

NIKOS MASTORAKIS

Technical University of Sofia, BULGARIA

Abstract – In this paper, a fast method for image matching on web pages is presented. Such method relies on performing cross correlation in the frequency domain between the web image and the image given in the user query. The cross correlation operation is modified. Instead of performing dot multiplication in the frequency domain, image subtraction is applied in two dimensions. It is proved mathematically that the number of computation steps required for the proposed fast matching method is less than that needed by conventional matching.

Keywords— Fast image subtraction, frequency domain, cross correlation, image matching.

I. Strategy of Fast Image Matching by Applying Cross Correlation in the Frequency Domain

First an image is given in the user query. Then, each sub-image in the web image is tested for the presence or absence of the image in the given query. At each pixel position in the web image, each sub-image is tested for matching with the image in the given query. Each sub-image in the web image is subtracted in two dimensions from the given image in the query, which has the same size as the sub-image. This subtraction is done in the spatial domain. When the final result is zero this means that the sub-image under test contains the required image (given the user query) and vice versa. Thus, we may conclude that this searching problem is cross correlation in the spatial domain between the web image under test and the image given in the user query.

In this section, a fast algorithm for image matching based on applying two dimensional cross correlations between the tested web image and the image given in the user query is described. The image given in the user query can be represented as a sliding window moving pixel by pixel on the web image. 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 spatial domain via the inverse Fourier transform [1]. As a result, these cross correlations can be represented by a product in the frequency domain. Thus, by using cross correlation in the frequency domain a speed up in an order of magnitude can be achieved during the detection process [1-35].

Here, the theory of cross correlation is modified. To achieve the best matching, instead of dot multiplication, matrix subtraction is performed. In the matching phase, a sub-image X of size mxz (sliding window) is extracted from the tested web image, which has a size PxT. Let W be the two dimensional image given in the user query. This image has a size of mxz and can be represented as mxz matrix. The output of this matching h can be calculated as follows:

$$h = \sum_{j=1}^{m} \sum_{k=1}^{z} X(j,k) - W(j,k) \tag{1}$$

Eq.(1) represents the result for a particular sub-image X. It can be computed for the whole image Ψ as follows:

$$h(u,v) = \sum_{j=-m/2}^{m/2} \sum_{k=-z/2}^{z/2} \Psi(u+j, v+k) - W(j,k) \tag{2}$$

Eq.(2) represents the proposed modified cross correlation operation. In general, for any two functions f and g, their modified cross correlation can be performed by applying two dimensional subtraction. These modified cross correlations can be obtained by:

$$g(x,y) \Theta f(x,y) = \left(\sum_{m=-\infty}^{\infty} \sum_{z=-\infty}^{\infty} g(m,z) - f(x+m, y+z) \right) \tag{3}$$

Where Θ denotes the modified cross correlation. Therefore, Eq.(2) can be written as follows:

$$h(i) = W \Theta \Psi \tag{4}$$

where $h(i)$ is the matching result when the sliding window is located at position (u,v) in the web image Ψ and $(u,v)\in[P-m+1,T-n+1]$.
Now, the above cross correlation can be expressed in terms of the Fourier Transform:

$$W\Theta\Psi = F^{-1}\big(F(\Psi) - F^*(W)\big) \qquad (5)$$

(*) means the conjugate of the *FFT* for the image given in the user query. Hence, by evaluating this cross correlation, a speed up ratio can be obtained comparable to conventional image matching.

II. Analysis of the Proposed Fast Image Matching Algorithm

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

1. For a tested web image of *NxN* pixels, the *2D-FFT* requires a number equal to $N^2log_2N^2$ of complex computation steps. Also, the same number of complex computation steps is required for computing the *2D-FFT* of the image given in the user query.

2. The inverse *2D-FFT* is computed. So, *one* backward and *two* forward transforms have to be computed. Therefore, for an image under test, the total number of the *2D-FFT* to compute is $(2+1)N^2log_2N^2$.

3. The web image and the image given in the user query should be subtracted in the frequency domain. Therefore, a number of complex computation steps equal to N^2 should be added.

4. The number of computation steps required by the fast image matching is complex and must be converted into a real version. It is known that the two dimensional Fast Fourier Transform requires $(N^2/2)log_2N^2$ complex multiplications and $N^2log_2N^2$ complex additions [36]. 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:

$$\rho=6((N^2/2)log_2N^2) + 2(N^2log_2N^2) \qquad (6)$$

which may be simplified to:

$$\rho=5N^2log_2N^2 \qquad (7)$$

Performing complex subtraction in the frequency domain also requires $2N^2$ real operations.

5. In order to perform cross correlation in the frequency domain, the image given in the user query must have the same size as the web image. Assume that the the image given in the user query has a size of (nxn) dimensions. So, the search process will be done over sub-images of (nxn) dimensions. Therefore, a number of zeros = (N^2-n^2) must be added to the image given in the user query. This requires a total real number of computation steps = (N^2-n^2). Moreover, after computing the *2D-FFT* for the image given in the user query, the conjugate of this matrix must be obtained. So, a real number of computation steps = N^2 should be added in order to obtain the conjugate of the image given in the user query. Also, a number of real computation steps equal to 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 the *2D-FFT*. To create a complex number requires two real floating point operations. So, the total number of computation steps required for the fast image matching becomes:

$$\sigma=(2+1)(5N^2log_2N^2) +2N^2+ N^2+(N^2-n^2) +N \qquad (8)$$

which can be reformulated as:

$$\sigma=(3)(5N^2log_2N^2) +(4N^2-n^2) +N \qquad (9)$$

6. Using a sliding window of size nxn for the same image of *NxN* pixels, $(2n^2-1)(N-n+1)^2$ computation steps are required for image matching when using traditional two dimensional subtraction. The theoretical speed up factor η can be evaluated as follows:

$$\eta = \frac{(2n^2-1)(N-n+1)^2}{(15N^2log_2N^2)+(4N^2-n^2)+N} \qquad (10)$$

The theoretical speed up ratio given by Eq.(10) with different sizes for the web images and different in size user images (given in user query) is listed in Tables 1-13. An interesting property with Fast image matching is that the number of computation steps does not depend on either the on size of the image given in the user query or the sub-image extracted from the web image. The effect of (n) on the number of computation steps is very small and can be ignored. This is in contrast to conventional image matching in which the number of computation steps is increased with the size of both the extracted sub-image and the image given in the user query.

III. Conclusion

A new fast method for image matching has been presented. This has been achieved by modifying the theory of cross correlation. Instead of performing dot multiplication, image subtraction has been applied in two dimensions. The modified cross correlation has been applied in the frequency domain between web image and the image given in a query. It has been proven mathematically that the number of computation steps required for the presented fast image matching is less than that needed by conventional image matching. This method reducers the time required (ex. by neural networks) for image matching especially when the two images are exactly the same.

References

[1] R. Klette, and Zamperon, "Handbook of image processing operators, " John Wiley & Sonsltd, 1996.

[2] H. M. El-Bakry, " Fast Virus Detection by using High Speed Time Delay Neural Networks ," Accepted for publication in journal of computer virology.

[3] H. M. El-Bakry, " A New Neural Design for Faster Pattern Detection Using Cross Correlation and Matrix Decomposition," Accepted for publication in Neural World journal.

[4] H. M. El-Bakry, " New Fast Principal Component Analysis For Real-Time Face Detection," Accepted for publication in MG&V Journal.

[5] H. M. El-Bakry, " Fast Record Detection in Large Databases Using New High Speed Time Delay Neural Networks," Accepted for publication in International Joint Conference on Neural Networks, June 14-19, 2009, Atlanta, USA.

[6] H. M. El-Bakry, "New Faster Normalized Neural Networks for Sub-Matrix Detection using Cross Correlation in the Frequency Domain and Matrix Decomposition," Applied Soft Computing journal, vol. 8, issue 2, March 2008, pp. 1131-1149.

[7] H. M. El-Bakry, "A Novel Modified Hopfield Neural Network for Perfect Calculation of Magnetic Resonance Spectroscopy," International Journal on Biomedical Science, vol. 3, no. 1, 2008, pp. 63-75.

[8] H. M. El-Bakry, and N. Mastorakis, "A New Technique for Detecting Dental Diseases by using High Speed Neuro-Computers," WSEAS Trans. on Signal Processing, vol. 8, issue 2, October 2008, pp. 1131-1149.

[9] H. M. El-Bakry, and Nikos Mastorakis, "A Real-Time Intrusion Detection Algorithm for Network Security," WSEAS Transactions on Communications, Issue 12, vol. 7, December 2008, pp. 1222-1234.

[10] H. M. El-Bakry, and Nikos Mastorakis, "A New Fast Forecasting Technique using High Speed Neural Networks ," WSEAS Transactions on Signal Processing, Issue 10, vol. 4, October 2008, pp. 573-595.

[11] H. M. El-Bakry, and Nikos Mastorakis, "A Perfect QoS Routing Algorithm for Finding the Best Path for Dynamic Networks ," WSEAS Transactions on Communications, Issue 12, vol. 7, December 2008, pp. 1123-1136.

[12] H. M. El-Bakry, and Nikos Mastorakis, "A Novel Fast Kolmogorov's Spline Complex Network for Pattern Detection," WSEAS Transactions on Systems, Issue 11, vol. 7, November 2008, pp. 1310-1328.

[13] H. M. El-Bakry, and Nikos Mastorakis, " A Modified Hopfield Neural Network for Perfect Calculation of Magnetic Resonance Spectroscopy," WSEAS Transactions on Information Science and Applications, issue 12, vol. 5, December 2008, pp. 1654-1666.

[14] H. M. El-Bakry, "New Fast Principal Component Analysis for Face Detection," Journal of Advanced Computational Intelligence and Intelligent Informatics, vol.11, no.2, 2007, pp. 195-201.

[15] Hazem M. El-Bakry and Mohamed Hamada, " New Fast Decision Tree Classifier for Identifying Protein Coding Regions," Lecture Notes in Computer Science, Springer, ISICA 2008, LNCS 5370, 2008, pp. 489-500.

[16] Hazem M. El-Bakry and Mohamed Hamada, "A New Implementation for High Speed Neural Networks in Frequency Space," Lecture Notes in Artificial Intelligence, Springer, KES 2008, Part I, LNAI 5177, pp. 33-40.

[17] Hazem M. El-Bakry, and Nikos Mastorakis "New Fast Normalized Neural Networks for Pattern Detection," Image and Vision Computing Journal, vol. 25, issue 11, 2007, pp. 1767-1784.

[18] Hazem M. El-Bakry and Nikos Mastorakis, "Fast Code Detection Using High Speed Time Delay Neural Networks," Lecture Notes in Computer Science, Springer, vol. 4493, Part III, May 2007, pp. 764-773.

[19] H. M. El-Bakry, "Fast Painting with Different Colors Using Cross Correlation in the Frequency Domain," International Journal of Computer Science, vol.1, no.2, 2006, pp. 145-156.

[20] H. M. El-Bakry, "New High Speed Normalized Neural Networks for Fast Pattern Discovery on Web Pages," International Journal of Computer Science and Network Security, vol.6, No. 2A, February 2006, pp.142-152.

[21] Hazem M. El-Bakry, and Qiangfu Zhao, "Fast Normalized Neural Processors For Pattern Detection Based on Cross Correlation Implemented in the Frequency Domain," Journal of Research and Practice in Information Technology, Vol. 38, No.2, May 2006, pp. 151-170.

[22] Hazem M. El-Bakry, and Qiangfu Zhao, "Fast Time Delay Neural Networks," International Journal of Neural Systems, vol. 15, no.6, December 2005, pp.445-455.

[23] Hazem M. El-Bakry, and Qiangfu Zhao, "Speeding-up Normalized Neural Networks For Face/Object Detection," Machine Graphics & Vision Journal (MG&V), vol. 14, No.1, 2005, pp. 29-59.

[24] Hazem M. El-Bakry, "A New High Speed Neural Model For Character Recognition Using Cross Correlation and Matrix Decomposition," International Journal of Signal Processing, vol.2, no.3, 2005, pp. 183-202.

[25] Hazem M. El-Bakry, and Qiangfu Zhao, "A Fast Neural Algorithm for Serial Code Detection in a Stream of Sequential Data," International Journal of Information Technology, vol.2, no.1, pp. 71-90, 2005.
[26] Hazem M. El-Bakry, and Qiangfu Zhao, "Fast Complex Valued Time Delay Neural Networks," International Journal of Computational Intelligence, vol.2, no.1, pp. 16-26, 2005.
[27] Hazem M. El-Bakry, and Qiangfu Zhao, "Fast Pattern Detection Using Neural Networks Realized in Frequency Domain," Enformatika Transactions on Engineering, Computing, and Technology, February 25-27, 2005, pp. 89-92.
[28] Hazem M. El-Bakry, and Qiangfu Zhao, "Sub-Image Detection Using Fast Neural Processors and Image Decomposition," Enformatika Transactions on Engineering, Computing, and Technology, February 25-27, 2005, pp. 85-88.
[29] Hazem M. El-Bakry, and Qiangfu Zhao, "Face Detection Using Fast Neural Processors and Image Decomposition," International Journal of Computational Intelligence, vol.1, no.4, 2004, pp. 313-316.
[30] Hazem M. El-Bakry, and Qiangfu Zhao, "A Modified Cross Correlation in the Frequency Domain for Fast Pattern Detection Using Neural Networks," International Journal on Signal Processing, vol.1, no.3, 2004, pp. 188-194.
[31] Hazem M. El-Bakry, and Qiangfu Zhao, "Fast Object/Face Detection Using Neural Networks and Fast Fourier Transform," International Journal on Signal Processing, vol.1, no.3, 2004, pp. 182-187.
[32] H. M. El-Bakry, and H. Stoyan**,** “FNNs for Code Detection in Sequential Data Using Neural Networks for Communication Applications, ” Proc. of the First International Conference on Cybernetics and Information Technologies, Systems and Applications: CITSA 2004, 21-25 July, 2004. Orlando, Florida, USA, Vol. IV, pp. 150-153.
[33] H. M. El-Bakry, “Face detection using fast neural networks and image decomposition,” Neurocomputing Journal, vol. 48, 2002, pp. 1039-1046.
[34] H. M. El-Bakry, “Human Iris Detection Using Fast Cooperative Modular Neural Nets and Image Decomposition,” Machine Graphics & Vision Journal (MG&V), vol. 11, no. 4, 2002, pp. 498-512.
[35] H. M. El-Bakry, “Automatic Human Face Recognition Using Modular Neural Networks,” Machine Graphics & Vision Journal (MG&V), vol. 10, no. 1, 2001, pp. 47-73.
[36] J. W. Cooley, and J. W. Tukey, "An algorithm for the machine calculation of complex Fourier series," Math. Comput. 19, 1965, pp. 297–301.

Table 1
The Theoretical Speed up Ratio for Images with Different Sizes (n=20,25,30).

Image size	Speed up ratio (n=20)	Speed up ratio (n=25)	Speed up ratio (n=30)
100x100	2.5290	3.4807	4.3761
200x200	2.7576	4.0759	5.5420
300x300	2.7505	4.1480	5.7601
400x400	2.7118	4.1285	5.7895
500x500	2.6697	4.0869	5.7636
600x600	2.6300	4.0408	5.7196
700x700	2.5939	3.9955	5.6701
800x800	2.5612	3.9526	5.6201
900x900	2.5317	3.9128	5.5716
1000x1000	2.5049	3.8758	5.5255
1100x1100	2.4804	3.8415	5.4818
1200x1200	2.4579	3.8097	5.4407
1300x1300	2.4371	3.7801	5.4021
1400x1400	2.4180	3.7524	5.3656
1500x1500	2.4001	3.7266	5.3312
1600x1600	2.3834	3.7023	5.2988
1700x1700	2.3678	3.6794	5.2681
1800x1800	2.3532	3.6578	5.2390
1900x1900	2.3393	3.6374	5.2113
2000x2000	2.3263	3.6181	5.1850

Table 2
The Theoretical Speed up Ratio for Images with Different Sizes (n=35,40,45).

Image size	Speed up ratio (n=35)	Speed up ratio (n=40)	Speed up ratio (n=45)
100x100	5.1485	5.7459	6.1305
200x200	7.1099	8.7366	10.3823
300x300	7.5547	9.5010	11.5693
400x400	7.6703	9.7476	11.9983
500x500	7.6805	9.8185	12.1593
600x600	7.6504	9.8176	12.2057
700x700	7.6041	9.7843	12.1975
800x800	7.5518	9.7362	12.1618
900x900	7.4979	9.6813	12.1119
1000x1000	7.4446	9.6241	12.0549
1100x1100	7.3930	9.5667	11.9948
1200x1200	7.3434	9.5103	11.9338
1300x1300	7.2961	9.4554	11.8731
1400x1400	7.2511	9.4025	11.8136
1500x1500	7.2082	9.3516	11.7556
1600x1600	7.1675	9.3028	11.6993
1700x1700	7.1286	9.2560	11.6449
1800x1800	7.0916	9.2111	11.5924
1900x1900	7.0563	9.1680	11.5417
2000x2000	7.0226	9.1267	11.4929

Table 3
The Theoretical Speed up Ratio for Images with Different Sizes (n=50,55,60).

Image size	Speed up ratio (n=50)	Speed up ratio (n=55)	Speed up ratio (n=60)
100x100	6.2791	6.1827	5.8471
200x200	12.0104	13.5873	15.0827
300x300	13.7316	15.9608	18.2313
400x400	14.4004	16.9324	19.5735
500x500	14.6849	17.3776	20.2204
600x600	14.7997	17.5848	20.5465
700x700	14.8307	17.6712	20.7066
800x800	14.8173	17.6917	20.7739
900x900	14.7795	17.6743	20.7866
1000x1000	14.7281	17.6347	20.7661
1100x1100	14.6692	17.5819	20.7249
1200x1200	14.6066	17.5214	20.6709
1300x1300	14.5424	17.4565	20.6089
1400x1400	14.4780	17.3896	20.5421
1500x1500	14.4142	17.3218	20.4727
1600x1600	14.3516	17.2543	20.4020
1700x1700	14.2904	17.1875	20.3310
1800x1800	14.2309	17.1218	20.2603
1900x1900	14.1731	17.0575	20.1905
2000x2000	14.1170	16.9947	20.1219

Table 4
The Theoretical Speed up Ratio for Images with Different Sizes (n=65,70,75).

Image size	Speed up ratio (n=65)	Speed up ratio (n=70)	Speed up ratio (n=75)
100x100	5.2923	4.5528	3.6778
200x200	16.4695	17.7236	18.8242
300x300	20.5188	22.8003	25.0541
400x400	22.3038	25.1038	27.9549
500x500	23.1965	26.2897	29.4840
600x600	23.6706	26.9431	30.3505
700x700	23.9244	27.3127	30.8595
800x800	24.0531	27.5187	31.1602
900x900	24.1067	27.6252	31.3326
1000x1000	24.1135	27.6685	31.4226
1100x1100	24.0905	27.6707	31.4579
1200x1200	24.0480	27.6455	31.4564
1300x1300	23.9929	27.6018	31.4294
1400x1400	23.9295	27.5457	31.3847
1500x1500	23.8610	27.4812	31.3277
1600x1600	23.7893	27.4111	31.2621
1700x1700	23.7160	27.3375	31.1907
1800x1800	23.6419	27.2617	31.1153
1900x1900	23.5678	27.1849	31.0373
2000x2000	23.4942	27.1076	30.9579

Table 5
The Theoretical Speed up Ratio for Images with Different Sizes (n=80,85,90).

Image size	Speed up ratio (n=80)	Speed up ratio (n=85)	Speed up ratio (n=90)
100x100	2.7309	1.7904	0.9491
200x200	19.7535	20.4969	21.0431
300x300	27.2599	29.3985	31.4520
400x400	30.8392	33.7394	36.6389
500x500	32.7642	36.1151	39.5222
600x600	33.8794	37.5167	41.2497
700x700	34.5532	38.3822	42.3355
800x800	34.9673	38.9297	43.0374
900x900	35.2197	39.2774	43.4967
1000x1000	35.3673	39.4945	43.7960
1100x1100	35.4446	39.6231	43.9861
1200x1200	35.4737	39.6904	44.0998
1300x1300	35.4690	39.7143	44.1590
1400x1400	35.4404	39.7070	44.1786
1500x1500	35.3948	39.6772	44.1691
1600x1600	35.3370	39.6308	44.1381
1700x1700	35.2705	39.5723	44.0910
1800x1800	35.1978	39.5049	44.0319
1900x1900	35.1209	39.4311	43.9638
2000x2000	35.0410	39.3527	43.8890

Table 6
The Theoretical Speed up Ratio for Images with Different Sizes (n=95,100,105).

Image size	Speed up ratio (n=95)	Speed up ratio (n=100)	Speed up ratio (n=105)
100x100	0.3148	0.0097	--------------
200x200	21.3837	21.5136	21.4309
300x300	33.4042	35.2397	36.9447
400x400	39.5220	42.3735	45.1789
500x500	42.9715	46.4492	49.9421
600x600	45.0660	48.9534	52.9000
700x700	46.4018	50.5704	54.8307
800x800	47.2807	51.6497	56.1349
900x900	47.8687	52.3846	57.0357
1000x1000	48.2636	52.8895	57.6658
1100x1100	48.5260	53.2357	58.1079
1200x1200	48.6950	53.4694	58.4162
1300x1300	48.7968	53.6215	58.6268
1400x1400	48.8493	53.7134	58.7651
1500x1500	48.8654	53.7604	58.8489
1600x1600	48.8540	53.7734	58.8912
1700x1700	48.8220	53.7604	58.9015
1800x1800	48.7742	53.7275	58.8872
1900x1900	48.7146	53.6792	58.8535
2000x2000	48.6458	53.6191	58.8048

Table 7
The Theoretical Speed up Ratio for Images with Different Sizes (n=110,115,120).

Image size	Speed up ratio (n=110)	Speed up ratio (n=115)	Speed up ratio (n=120)
200x200	21.1368	20.6355	19.9348
300x300	38.5066	39.9142	41.1576
400x400	47.9245	50.5973	53.1848
500x500	53.4374	56.9227	60.3860
600x600	56.8942	60.9248	64.9808
700x700	59.1723	63.5849	68.0585
800x800	60.7270	65.4166	70.1948
900x900	61.8136	66.7099	71.7162
1000x1000	62.5846	67.6384	72.8194
1100x1100	63.1355	68.3115	73.6287
1200x1200	63.5289	68.8009	74.2258
1300x1300	63.8068	69.1553	74.6664
1400x1400	63.9987	69.4086	74.9892
1500x1500	64.1255	69.5850	75.2221
1600x1600	64.2024	69.7021	75.3854
1700x1700	64.2407	69.7732	75.4944
1800x1800	64.2488	69.8080	75.5604
1900x1900	64.2333	69.8142	75.5923
2000x2000	64.1990	69.7976	75.5968

Table 8
The Theoretical Speed up Ratio for Images with Different Sizes (n=125,130,135).

Image size	Speed up ratio (n=125)	Speed up ratio (n=130)	Speed up ratio (n=135)
200x200	19.0451	17.9804	16.7576
300x300	42.2279	43.1180	43.8217
400x400	55.6755	58.0585	60.3234
500x500	63.8158	67.2011	70.5311
600x600	69.0514	73.1263	77.1954
700x700	72.5835	77.1501	81.7491
800x800	75.0526	79.9812	84.9720
900x900	76.8245	82.0266	87.3148
1000x1000	78.1203	83.5337	89.0522
1100x1100	79.0804	84.6598	90.3600
1200x1200	79.7973	85.5089	91.3545
1300x1300	80.3340	86.1524	92.1157
1400x1400	80.7349	86.6402	92.6997
1500x1500	81.0316	87.0083	93.1472
1600x1600	81.2474	87.2831	93.4879
1700x1700	81.3997	87.4845	93.7442
1800x1800	81.5016	87.6273	93.9331
1900x1900	81.5633	87.7231	94.0677
2000x2000	81.5925	87.7810	94.1582

Table 9
The Theoretical Speed up Ratio for Images with Different Sizes (n=140,145,150).

Image size	Speed up ratio (n=140)	Speed up ratio (n=145)	Speed up ratio (n=150)
200x200	15.3970	13.9219	12.3589
300x300	44.3344	44.6525	44.7739
400x400	62.4607	64.4615	66.3178
500x500	73.7957	76.9850	80.0897
600x600	81.2487	85.2768	89.2705
700x700	86.3713	91.0077	95.6497
800x800	90.0167	95.1068	100.2343
900x900	92.6811	98.1180	103.6179
1000x1000	94.6688	100.3762	106.1675
1100x1100	96.1744	102.0965	108.1198
1200x1200	97.3278	103.4228	109.6333
1300x1300	98.2181	104.4539	110.8174
1400x1400	98.9080	105.2598	111.7496
1500x1500	99.4431	105.8911	112.4861
1600x1600	99.8569	106.3853	113.0685
1700x1700	100.1744	106.7705	113.5282
1800x1800	100.4148	107.0682	113.8890
1900x1900	100.5930	107.2949	114.1695
2000x2000	100.7204	107.4637	114.3843

Table 10
The Theoretical Speed up Ratio for Images with Different Sizes (n=155,160,165).

Image size	Speed up ratio (n=155)	Speed up ratio (n=160)	Speed up ratio (n=165)
200x200	10.7375	9.0909	7.4550
300x300	44.6979	44.4250	43.9568
400x400	68.0221	69.5676	70.9483
500x500	83.1009	86.0101	88.8092
600x600	93.2207	97.1189	100.9566
700x700	100.2887	104.9163	109.5245
800x800	105.3913	110.5698	115.7622
900x900	109.1734	114.7772	120.4221
1000x1000	112.0358	117.9743	123.9763
1100x1100	114.2377	120.4441	126.7326
1200x1200	115.9534	122.3771	128.8986
1300x1300	117.3030	123.9052	130.6185
1400x1400	118.3723	125.1226	131.9953
1500x1500	119.2232	126.0974	133.1039
1600x1600	119.9018	126.8806	134.0002
1700x1700	120.4429	127.5103	134.7261
1800x1800	120.8730	128.0161	135.3141
1900x1900	121.2128	128.4207	135.7895
2000x2000	121.4784	128.7422	136.1719

Table 11
The Theoretical Speed up Ratio for Images with Different Sizes (n=170,175,180).

Image size	Speed up ratio (n=170)	Speed up ratio (n=175)	Speed up ratio (n=180)
200x200	5.8691	4.3758	3.0206
300x300	43.2965	42.4485	41.4186
400x400	72.1588	73.1945	74.0514
500x500	91.4905	94.0470	96.4718
600x600	104.7258	108.4188	112.0279
700x700	114.1054	118.6514	123.1548
800x800	120.9610	126.1589	131.3487
900x900	126.1010	131.8071	137.5335
1000x1000	130.0352	136.1445	142.2978
1100x1100	133.0970	139.5313	146.0294
1200x1200	135.5120	142.2118	148.9922
1300x1300	137.4374	144.3566	151.3707
1400x1400	138.9854	146.0878	153.2974
1500x1500	140.2379	147.4946	154.8692
1600x1600	141.2560	148.6436	156.1585
1700x1700	142.0859	149.5853	157.2202
1800x1800	142.7630	150.3586	158.0968
1900x1900	143.3151	150.9936	158.8213
2000x2000	143.7639	151.5144	159.4198

Table 12
The Theoretical Speed up Ratio for Images with Different Sizes (n=185,190,195).

Image size	Speed up ratio (n=185)	Speed up ratio (n=190)	Speed up ratio (n=195)
200x200	1.8526	0.9238	0.2896
300x300	40.2137	38.8422	37.3137
400x400	74.7264	75.2167	75.5207
500x500	98.7585	100.9014	102.8948
600x600	115.5461	118.9664	122.2822
700x700	127.6085	132.0054	136.3387
800x800	136.5234	141.6760	146.8000
900x900	143.2736	149.0207	154.7685
1000x1000	148.4887	154.7112	160.9589
1100x1100	152.5853	159.1933	165.8475
1200x1200	155.8475	162.7724	169.7612
1300x1300	158.4745	165.6627	172.9301
1400x1400	160.6093	168.0186	175.5203
1500x1500	162.3570	169.9534	177.6538
1600x1600	163.7962	171.5522	179.4222
1700x1700	164.9864	172.8795	180.8955
1800x1800	165.9737	173.9853	182.1275
1900x1900	166.7942	174.9086	183.1607
2000x2000	167.4763	175.6804	184.0284

Table 13
The Theoretical Speed up Ratio for Images with Different Sizes (n=200,205,210).

Image size	Speed up ratio (n=200)	Speed up ratio (n=205)	Speed up ratio (n=210)
300x300	35.6392	33.8309	31.9025
400x400	75.6370	75.5654	75.3061
500x500	104.7339	106.4139	107.9307
600x600	125.4872	128.5753	131.5408
700x700	140.6018	144.7881	148.8915
800x800	151.8886	156.9356	161.9346
900x900	160.5106	166.2409	171.9532
1000x1000	167.2260	173.5065	179.7946
1100x1100	172.5423	179.2719	186.0310
1200x1200	176.8087	183.9095	191.0582
1300x1300	180.2718	187.6825	195.1574
1400x1400	183.1096	190.7817	198.5320
1500x1500	185.4534	193.3479	201.3327
1600x1600	187.4019	195.4869	203.6729
1700x1700	189.0302	197.2795	205.6393
1800x1800	190.3965	198.7884	207.2991
1900x1900	191.5466	200.0628	208.7054
2000x2000	192.5167	201.1418	209.9001