International Journal of Computing Academic Research (IJCAR)
ISSN 2305-9184 Volume 1, Number 2 (December 2012), pp. 79-87
© MEACSE Publications
http://www.meacse.org/ijcar

Performance Evaluation of Exhaustive-Search Equivalent Pattern Matching under Chebyshev distance

Mohamed Yousef[1]and Khaled F. Hussain[2]

[1, 2] Faculty of Computers and Information, Assiut University, Assiut, Egypt.

Abstract
Pattern Matching is a fundamental problem in computer vision, and image and video processing. Exhaustive-Search equivalent algorithms yield the same results as exhaustively searching all patterns in the image but significantly faster. Though much work have been done over the L_1 and L_2 distances, only small amount of work has been dedicated to the Chebyshev distance though its importance in many applications. In this paper, we provide an evaluation of available state-of-art exhaustive-search equivalent algorithm that targets the Chebyshev distance; we also provide detailed analysis of the performance characteristics of evaluated algorithms.

Keywords: Pattern matching, template matching, fast algorithms, full search equivalent algorithm, Chebyshev distance, NOM.

Introduction

Template Matching or Pattern Matching is a fundamental problem in computer graphics, computer vision, signal processing, image and video processing, with a wide variety of applications such as video block motion estimation [1], image based rendering [2], image inpainting[3], object detection [4], super resolution [5], texture synthesis [6], texture edge extraction [7], image filtering [8], image summarizing [9], and even image compression [10].

The Template matching problem can be defined as follows: given an N=nx n pixels image sub-window q (called Template or Patch) find either the most similar patch to it in the image, or all patches p where the distance between p and $q$is below a certain threshold T according to a predefined dissimilarity measure. The basic algorithm for solving this problem is Full Search or Exhaustive Search, in which the distance between q and all template-sized sub-windows in the image (overlapped) is computed and we return either the patch with the smallest distance or all patches with distance below T, according to the instance of the problem we are considering.

In this work we are interested in the first instance where only the closest patch is required and many number of queries is done on the same image, as this is the interest of most of the applications that require template matching. This has important implication on our evaluation method of algorithms, as the most important property we require is exact fast evaluation of many number of queries on the same image, so a relatively slow initialization of an algorithm is acceptable as long as the total time (initialization + query processing) is fast compared to the respective times of other algorithms.

Though much work have been done in exhaustive search equivalent matching in the L_1 and L_2 only very small, scattered work have been dedicated to the Chebyshev distance [11]. In this paper we provide an evaluation of state-of-art algorithms that either directly targets or can be modified to target the Chebyshev distance, using the dataset introduced in [12].

Related Work

There is a huge history of work in accelerating pattern matching process, methods can be classified into two broad categories: Exact and Approximate pattern matching. A Complete review is beyond the scope of this article, the reader might refer to [13] for extensive reviews. Here we are only interested in exact methods and will give a brief overview on them. We here discuss exact methods generally and are not tied to those using the Chebyshev distance.

One of the most simple and powerful trails to accelerate Exact pattern matching is Partial Distortion Elimination (PDE)[14] which terminates the evaluation of the current candidate sub-window as soon as the current value of norm is guaranteed to exceed the current minimum distance.

Fast-Fourier Transform-based approaches [15] have also been traditionally used for accelerating pattern matching in the L_2 norm, especially for large pattern sizes. The idea is based on observing that the L_2 norm between two *M* pixels-sized sub-windows *X, Y* can be written as

$$\|X-Y\|_2^2 = \|X\|_2^2 + \|Y\|_2^2 - 2 \cdot \sum_{i=1}^{M} x_i \cdot y_i$$

FFT allows fast computation of the third term$\sum_{i=1}^{M} x_i \cdot y_i$ in the frequency domain using the correlation theorem.$\|X\|_2^2$is calculated for the all sub-windows in the image using fast incremental techniques like Summed Area Tables [16] or Box-Filtering [17],$\|Y\|_2^2$ is computed once where *Y* is the input query.

Projection Kernels (PK) based algorithms [18,19,20] are based on the idea that we can get approximate value of the distance *d(X-Y)* (where *d* is a norm) by projecting both *X* and *Y* into a number of basis vectors$U = \{u_1, \dots, u_k\}$, generally if *U* are mutually orthonormal it can be shown that:

$$d(X-Y) \geq \sum_{n=1}^{k} \frac{1}{d(u_n)} d(X^T u_n - Y^T u_n)$$

this provides a lower bound on the actual distance value between *X,Y.* The algorithm works by providing very fast method of projecting all image sub-windows to *U*, then when examining candidates, it keeps summing the values of$d(X^T u_n - Y^T u_n)$(assuming *U* are unit vectors) and rejects the candidate once the sum exceeds the current minimum or the input threshold value. In all these methods *U*basis vectors of the Walsh-Hadamard Transform. Though it is claimed that the method works for any norm, it is only proven for L_2 in [18], for other norms, the authors suggest using inter-norm relations to induce bounds on the distance derived from the L_2, but this loses the algorithm most of its exact rejection capabilities, as it loosens its bounds.

Low Resolution Pruning (LRP) [21] is based on the idea of transforming sub-windows in the image into a lower resolution variants, by summing the pixel values of partitions of the sub-windows of size *M*, this is repeated for *T* levels of resolution, after that an upper bound on the value of *d(X-Y)* (where *d* is an L_p norm) is established for every resolution level *t*, assuming *min* is current minimum distance or input threshold value, as follows

$$d(X-Y) \leq M^{\frac{t\cdot(p-1)}{p}} \cdot min$$

for the Chebyshev distance (L_∞), we have

$$\lim_{p\to\infty} M^{\frac{t\cdot(p-1)}{p}} = M^t$$

so the inequality used by our implementation is$d(X-Y) \leq M^t \cdot min.$

Incremental Dissimilarity Approximations Algorithm (IDA) [22] is based on the idea of partitioning each candidate sub-window into a number *r* of disjoint sets by defining a partition *P* of set $S = \{S_1, \dots, S_M\}$, using this partitioning it is able to establish a lower bound on the value *d(X-Y)* (where *d* is an L_p norm), at level *k* we have

$$d(X-Y) \geq \sum_{t=1}^{k-1}\left(\sum_{i\in S_t}|x_i - y_i|^p\right) + \sum_{l=k}^{M}\left|\|X\|_{p,S_t} - \|Y\|_{p,S_t}\right|^p$$

The right side of the right hand side of the inequality represents the source of the speed-up as it is an estimation of the rest of the distance that is not actually calculated (the calculated part is the left-side). So a candidate is rejected at the point when right hand side of the inequality exceeds current minimum distance or input threshold value *min*. As IDA is proven correct for L_p norms, we here modify its public L_2 implementation for the Chebyshev distance, then we compare to it.

Dataset	**Image Size**	**Template Size**
0-0	160x120	8x8
0-1	160x120	16x16
0-2	160x120	32x32
0-3	160x120	64x64
1-0	320x240	8x8
1-1	320x240	16x16
1-2	320x240	32x32
1-3	320x240	64x64
1-4	320x240	128x128
2-0	640x480	8x8
2-1	640x480	16x16
2-2	640x480	32x32
2-3	640x480	64x64
2-4	640x480	128x128
3-0	1280x960	8x8
3-1	1280x960	16x16
3-2	1280x960	32x32
3-3	1280x960	64x64
3-4	1280x960	128x128
4-0	2560x1920	8x8
4-1	2560x1920	16x16
4-2	2560x1920	32x32
4-3	2560x1920	64x64
4-4	2560x1920	128x128

Table 1: List of image and template sizes combination used in our experiments

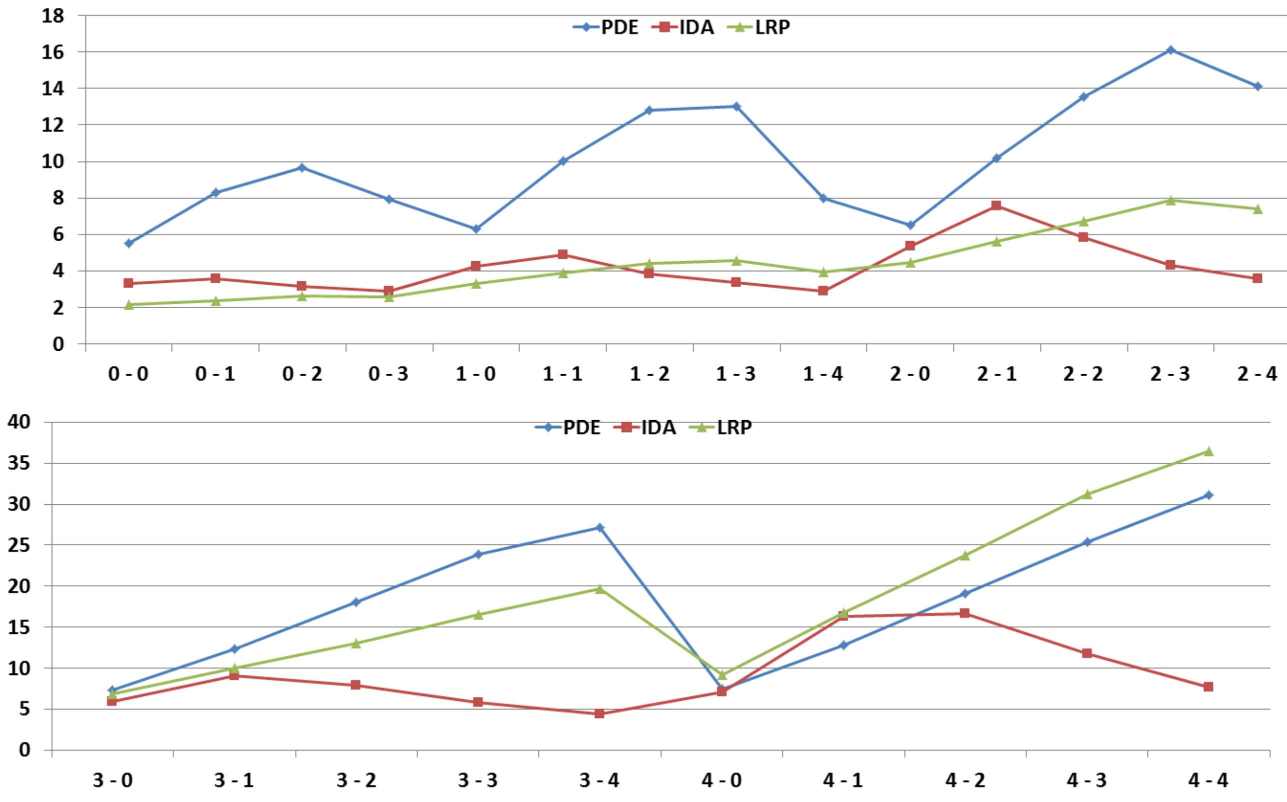

Figure 1: Results from images without noise, dataset numbers are those presented in Table 1

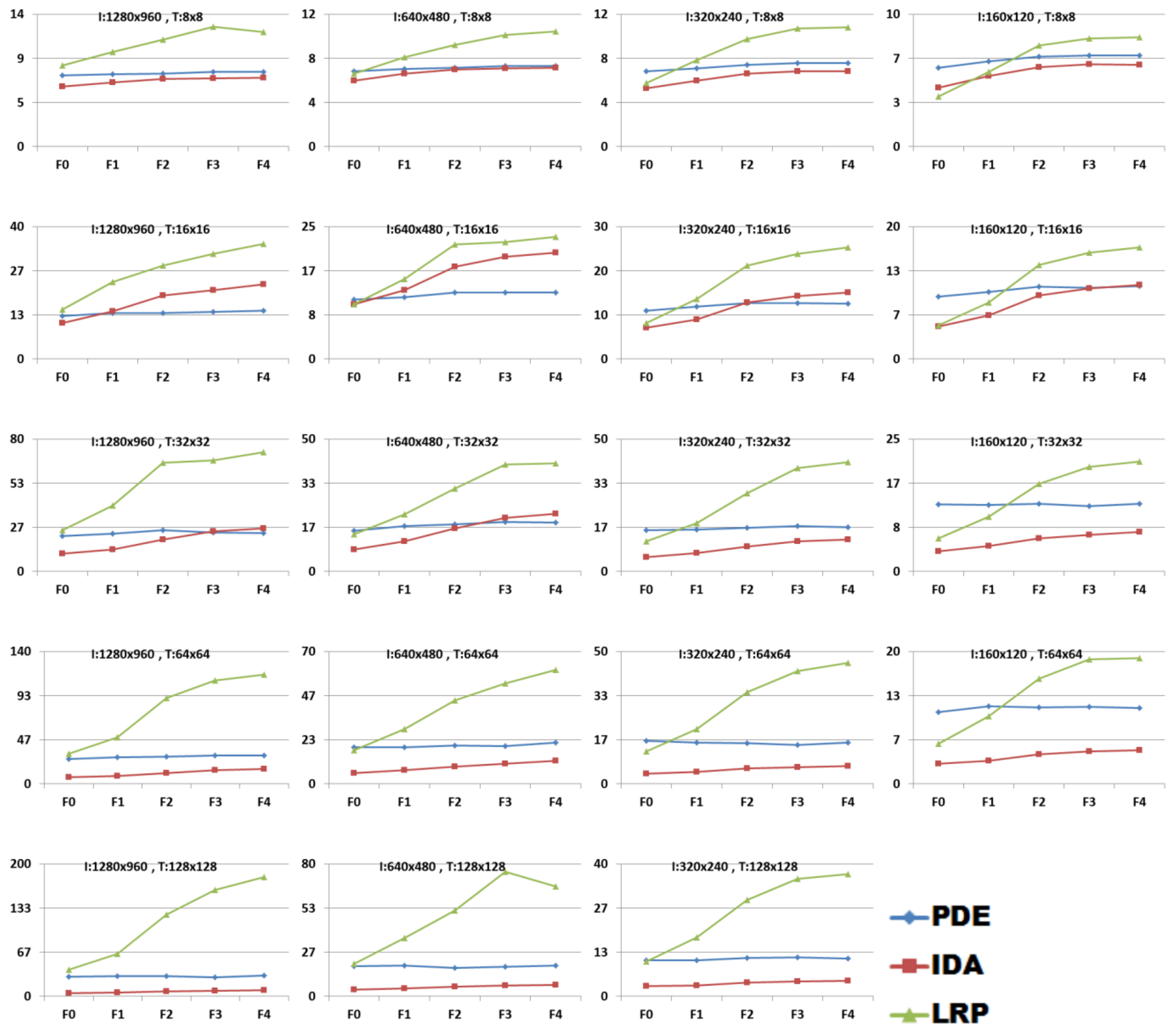

Figure 2: Results from images with blurring noise, F0-F4 represent five increasing blurring noise levels. "I:160x120 , T:8x8" means that the image is 160x120 pixels and template is 8x8 pixels, we use the same notation in all diagrams.

Experimental Results

In this section, we empirically compare the performance of current state-of-art, exhaustive-search equivalent pattern matching algorithms. We use the data set introduced here [12]. This dataset consists of 150 images divided into 5 different sizes (160x120, 320x240, 640x480, 1280x960, and 2560x1920) each comprising 30 images, these images are examined first without noise and then with 3 different types of noise (Blurring, JPEG compression and Gaussian noise) each of which are applied in 5 increasing levels which mean 15 types of noises are applied to images. Each image is examined using 5 different template sizes (8x8, 16x16, 32x32, 64x64, and 128x128). We compared PDE, IDA, and LRP, showing their speed-up over exhaustive search, in terms of the number of CPU clock cycles the algorithm spends executing. In order to examine each algorithm fairly enough and insure consistent timings, for each image we apply 20 queries using randomly choosen image patches. We set the number of partitions N_P of IDA to 8 in all our experiments. For LRP, we set $M = 4$ (as suggested in original paper).

Images without noise. Figure 1 shows results from images without noise. The figure is split into two diagrams with different scales to prevent small speed-ups at small sizes from being diminished by large speed-ups at big sizes. We can see that for image sizes less than or equal 1280x960, PDE is always the fastest, with a speed-up the generally increases with template size. For bigger images, LRP is always the fastest.

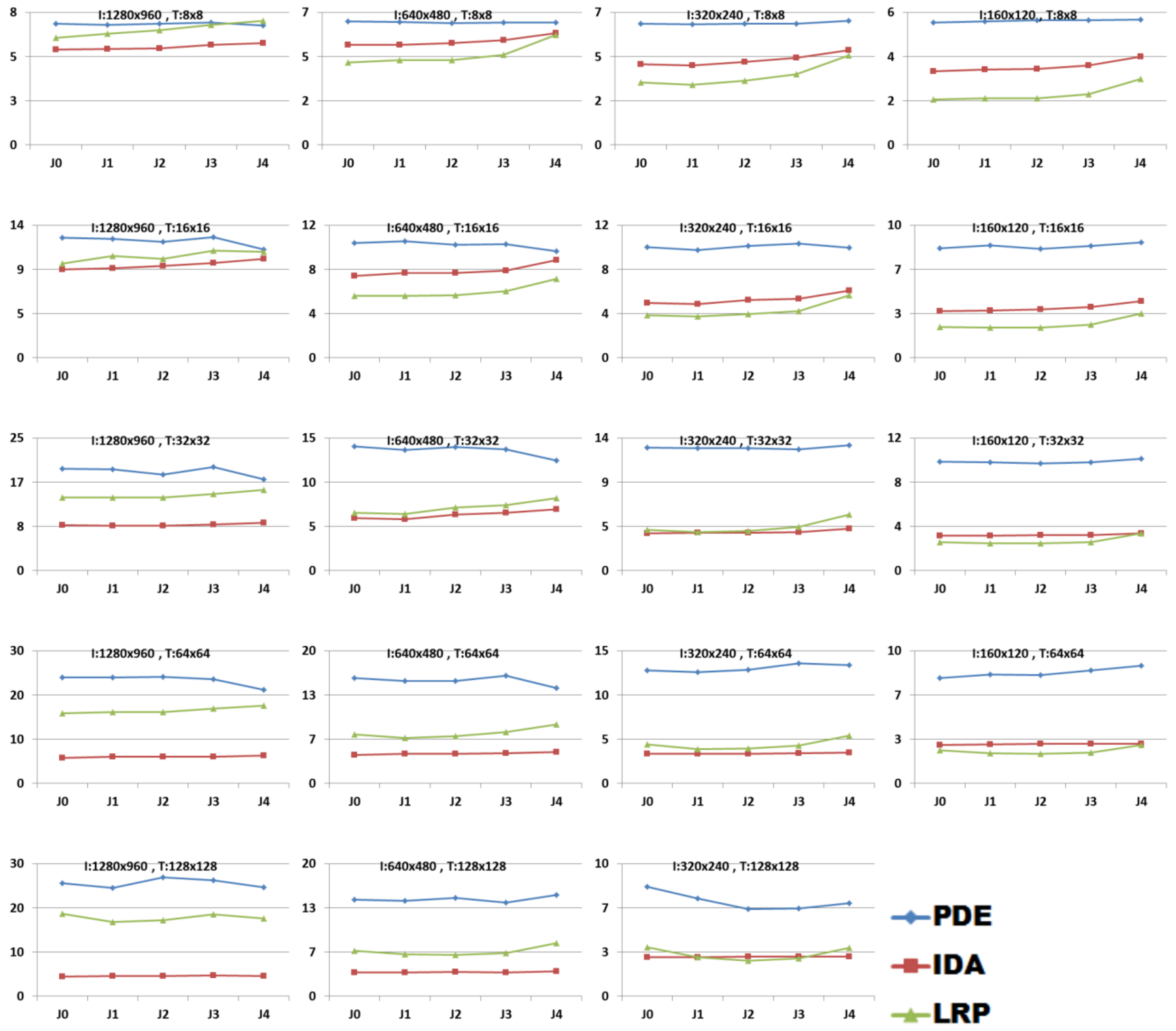

Figure 3: Results from images with JPEG compression noise, J0-J4 represent five increasing JPEG compression levels.

Blurred Images. Figure 2 shows results from images with Blurring noise. Five different levels of Gaussian low pass filter are used for blurring each image of the datasets, named F0, F1, F2, F3, and F4 in the diagrams, corresponding to Gaussian low pass filter having standard deviation $\sigma = 0.2, 0.9, 1.6, 2.3, 3$.

We can see that for noise levels F0-F4, LRP is almost always the fastest. For the F0 noise level, for the 1280x960 image size, LRP is always the fastest; otherwise PDE is always the fastest.

JPEG compressed images. Figure 3 shows results from images with JPEG noise. Five different JPEG compression quality levels are used for encoding the original images. The JPEG compression quality levels, named J0, J1, J2, J3, and J4 in the diagrams, correspond to quality measure $Q_{JPG} = 90, 70, 50, 30, 10$ respectively.

We can see that PDE is always the fastest. For most small image sizes 160x120, and 320x240 and small template sizes 8x8, and 16x16, LRP is generally the slowest, for bigger images and template sizes, IDA is the slowest.

Images with Gaussian noise. Figure 4 shows results from images with Gaussian noise. Four different levels of i.i.d. zero-mean Gaussian noise are added to each image, the four Gaussian noise levels having variances 325, 650, 1300 and 2600 referred to as G0, G1, G2 and G3 respectively.

Unlike other types of noise, the speed up decreases steadily as the noise level increases. For template size smaller than 128x128, PDE is always the fastest. For noise level G4, IDA is always
the fastest.

Analysis. From previous results we can conclude that under the Chebyshev distance, PDE is always the fastest except for large image sizes or for blurred images.

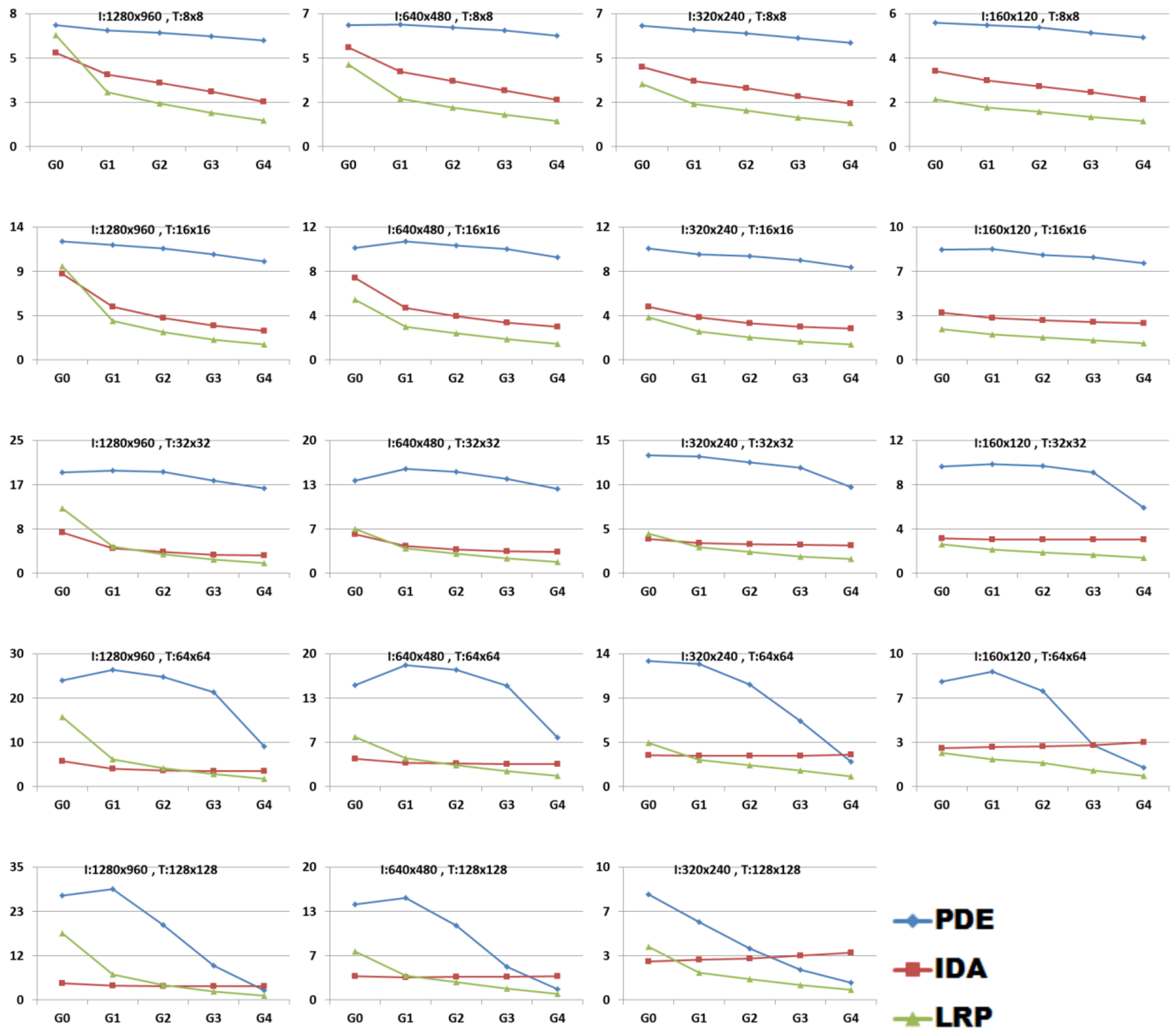

Figure 4: Results from images with Gaussian noise, G0-G4 represent four increasing Gaussian noise levels.

We can see that all algorithms are always faster than exhaustive search. The largest speed-up for images without noise is 36, for the LRP algorithm at 128x128 template size and 2560x1920 image sizes. For noisy images, it is 180 for LRP at 128x128 template sizes and 1280x960 image sizes.

References

[1] I. E. Richardson, I. E. G. Richardson, H.264 and MPEG-4 Video Compression: Video Coding for Next Generation Multimedia, Wiley, 2003.

[2] A. Fitzgibbon, Y. Wexler, A. Zisserman, Image-based rendering using image-based priors, Int. J. Comput. Vision 63 (2005) pp.141-151.

[3] A.Criminisi, P.Prez, K.Toyama, Region filling and object removal byexemplar-based image inpainting., IEEE Transactions on Image Processing13 (9) (2004),pp.1200-1212.

[4] R. M. Dufour, E. L. Miller, N. P. Galatsanos, Template matching based object recognition with unknown geometric parameters., IEEE Transactionson Image Processing 11 (2002), pp.1385-1396.

[5] W. T. Freeman, T. R. Jones, E. C. Pasztor, Example-based super-resolution, IEEE Comput. Graph. Appl. 22 (2002) 56-65.

[6] A. A. Efros, T. K. Leung, Texture synthesis by non-parametric sampling, in:Proceedings of the International Conference on Computer Vision-Volume2, IEEE Computer Society, 1999, pp. 1033-1038.

[7] L. Wolf, X. Huang, I. Martin, D. Metaxas, Patch-based texture edges and segmentation, in: Computer Vision ECCV 2006, Vol. 3952 of Lecture Notesin Computer Science, Springer Berlin Heidelberg, 2006, pp. 481-493.

[8] A. Buades, B. Coll, J.-M. Morel, A non-local algorithm for image denoising, in: Proceedings of the 2005 IEEE Computer Society Conference onComputer Vision and Pattern Recognition (CVPR'05) - Volume 2, IEEEComputer Society, 2005, pp. 60-65.

[9] D. Simakov, Y. Caspi, E. Shechtman, M. Irani, Summarizing visual datausing bidirectional similarity, IEEE Conference on Computer Vision andPattern Recognition (2008),pp. 1-8.

[10] T. Luczak, W. Szpankowski, A suboptimal lossy data compression based onapproximate pattern matching, IEEE Trans. Information Theory 43 (1996),pp:1439-1451.

[11] C. D. Cantrell, Modern Mathematical Methods for Physicists and Engineers, Cambridge University Press., 2000.

[12] W. Ouyang, F. Tombari, S. Mattoccia, L. Stefano, W. K. Cham, Performance evaluation of full search equivalent pattern matching algorithms,IEEE Trans. Pattern Anal. Mach. Intell. 34 (2012),pp.127-143.

[13] G. Shakhnarovich, T. Darrell, P. Indyk, Nearest-Neighbor Methods inLearning and Vision: Theory and Practice (Neural Information Processing), The MIT Press, 2006.

[14] L.-J. Liu, X.-B.Shen, X.-C.Zou, An improved fast encoding algorithm forvector

quantization, J. Am. Soc. Inf. Sci. Technol. 55 (2004),pp.81-87.
[15] J. Lewis, Fast template matching, Vision Interface (1995)pp.120-123.
[16] F. C. Crow, Summed-area tables for texture mapping, SIGGRAPH Comput. Graph. 18 (1984)pp.207-212.
[17] M.J., McDonnell, Box-filtering techniques, Computer Graphics and Image Processing 17 (1) (1981)pp.65-70.
[18] Y. Hel-Or, H. Hel-Or, Real-time pattern matching using projection kernels,IEEE Trans. Pattern Anal. Mach. Intell. 27 (9) (2005) pp.1430-1445.
[19] G.Ben-Artzi, H.Hel-Or, Y.Hel-Or, Thegray-code filter kernels, IEEETrans. Pattern Anal. Mach. Intell. 29 (2007), pp.382-393.
[20] W. Ouyang, W. K. Cham, Fast algorithm for walshhadamard transformon sliding windows, IEEE Trans. Pattern Anal. Mach. Intell. 32 (1) (2010),pp.165-171.
[21] M. Gharavi-Alkhansari, A fast globally optimal algorithm for templatematching using low-resolution pruning, IEEE Transactions on Image Processing 10 (2001),pp.526-533.
[22] F. Tombari, S. Mattoccia, L. Di Stefano, Full-search-equivalent patternmatching with incremental dissimilarity approximations, IEEE Trans. Pat-tern Anal. Mach. Intell. 31 (2009), pp.129-141.