Enhanced Low-Resolution Pruning for Fast Full-Search Template Matching

Stefano Mattoccia, Federico Tombari, and Luigi Di Stefano

Department of Electronics Computer Science and Systems (DEIS)
Advanced Research Center on Electronic Systems (ARCES)
University of Bologna, Italy
{federico.tombari,stefano.mattoccia,luigi.distefano}@unibo.it
www.vision.deis.unibo.it

Abstract. Gharavi-Alkhansari [1] proposed a full-search equivalent algorithm for speeding-up template matching based on L_p-norm distance measures. This algorithm performs a pruning of mismatching candidates based on multilevel pruning conditions and it has been shown that, under certain assumptions on the distortion between the image and the template, it is faster than the other full-search equivalent algorithms proposed so far, including algorithms based on the Fast Fourier Transform. In this paper we propose an original contribution with respect to Gharavi-Alkhansari's work that is based on the exploitation of an initial estimation of the global minimum aimed at increasing the efficiency of the pruning process.

1 Introduction

Template matching aims at locating a given template into an image. To perform this operation the *Full-search* (FS) algorithm compares the template with all the template-sized portions of the image which can be determined out of it. Hence, a *search area* can be defined in the image where the subimage candidates are selected and compared, one by one, to the template. In order to perform the comparison and select the most similar candidate, a function measuring the degree of similarity - or distortion - between template and subimage candidate is computed. A popular class of distortion functions is defined from the distance based on the L_p norm:

$$\delta_p(X, Y_j) = ||X - Y_j||_p = \Big(\sum_{i=1}^{N} \left| x_i - y_{j,i} \right|^p \Big)^{\frac{1}{p}} \tag{1}$$

with X being the template and Y_j the generic subimage candidate, both seen as vectors of cardinality N, and with $|| \cdot ||_p$ denoting the L_p norm, $p \geq 1$. With $p = 1$ we get the *Sum of Absolute Differences* (SAD) function, while by taking $p = 2$ and squaring (1) we get the *Sum of Squared Distances* (SSD) function.

The method proposed in [1] is a very fast FS-equivalent algorithm, yielding notable computational savings also compared to FFT-based algorithms. This

J. Blanc-Talon et al. (Eds.): ACIVS 2009, LNCS 5807, pp. 109–120, 2009.
© Springer-Verlag Berlin Heidelberg 2009

method, referred to as *Low Resolution Pruning* (*LRP*), applies several sufficient conditions for pruning mismatching candidates in order to carry out only a fraction of the computations needed by the full search approach.

By analysing the LRP algorithm we devised some modifications aimed at improving the overall performance of the approach. In particular, we devised three different full-search equivalent algorithms, conceptually based on the same idea but deploying different strategies of application. The common point of the three algorithms is to perform a fast initial estimation of the global minimum and consequently trying to exploit this knowledge so as to speed-up the matching process.

2 LRP Method

As in [1], we will refer to the template vector as $X = \{x_1, \cdots, x_N\}$, of cardinality N, and to the K candidate vectors against whom X must be matched as $Y_1, \cdots, Y_K$. Each vector will have the same cardinality as the template vector, i.e. $Y_j = \{y_{j,1}, \cdots, y_{j,N}\}$. In [1], a transformation is introduced, represented by a $N \times N'$ matrix, A, which replaces those elements of a vector corresponding to a block of pixels of size $\sqrt{M} \times \sqrt{M}$ with a single element equal to the sum of those elements. Hence, the resulting vector of the transformation will have cardinality $N' = \frac{N}{M}$ (from this point of view the transformation denoted by A acts as a binning operator). By applying this transformation on the template vector X and on a generic candidate vector Y_j the new vectors $\bar{X}$ and $\bar{Y}_j$ are obtained:

$$\bar{X} = AX \tag{2}$$

$$\bar{Y}_j = AY_j \tag{3}$$

The matrix p-norm, defined in [1] as:

$$||A||_p = \sup_{x \neq 0} \frac{||AX||_p}{||X||_p} = M^{\frac{p-1}{2p}} \tag{4}$$

induces the following inequality:

$$||A||_p \cdot ||X||_p \geq ||\bar{X}||_p \tag{5}$$

By applying the transformation defined by A to the template X and the generic candidate Y_j equation (5) is rewritten as:

$$||A||_p \cdot \delta_p(X, Y_j) \geq \delta_p(\bar{X}, \bar{Y}_j) \tag{6}$$

By introducing a threshold, D, which is obtained by computing δ_p on a *good* candidate Y_b:

$$D = ||A||_p \cdot \delta_p(X, Y_b) \tag{7}$$

a pruning condition can be tested for each candidate Y_j:

$$\delta_p(\bar{X}, \bar{Y}_j) > D \tag{8}$$

If (8) holds, then Y_j does not represent a better candidate compared to Y_b due to (6). Y_b is obtained by performing an exhaustive search between $\bar{X}$ and the K transformed candidates $\bar{Y}_1, \cdots, \bar{Y}_N$ and by choosing the candidate which leads to the global minimum.

The basic LRP approach described so far is extended in [1] by defining several levels of resolution and, correspondingly, a transformation matrix for each pair of consecutive levels. Given $T+1$ levels of resolution, with level 0 being the full resolution one and level T being the lowest resolution one, first the transformation is iteratively applied in order to obtain the several versions of the vectors at reduced cardinality. Each element is obtained by summing corresponding M elements of its upper resolution level (usually, at the lowest level each vector is made out of a single element, which coincides with the sum of all the full-resolution elements). That is, at level t the cardinality of vector Y_j is reduced from N (original size) to $\frac{N}{M^t}$. We will refer to X^t and Y_j^t as the template and candidate vectors transformed to level t.

After this initial step, the basic LRP is iterated T times, starting from the lowest resolution level (i.e. T). At generic step t first the initial candidate is determined by searching between those candidates not pruned so far in the current level (i.e. t) and by choosing the one which leads to the minimum distance, i.e. Y_b. Then, the threshold D^t is computed as:

$$D^t = ||A||_{p,t} \cdot \delta_p(X, Y_b) \tag{9}$$

with the transformation matrix p-norm $||A||_{p,t}$ being as:

$$||A||_{p,t} = M^{\frac{t \cdot (p-1)}{2p}} \tag{10}$$

Finally the pruning test is applied at each left candidate of the current level t:

$$\delta_p(X^t, Y_j^t) > D^t \tag{11}$$

As it can be easily inferred, the strength of the method is to prune the majority of candidates at the lower levels, based on the fact that computing $\delta_p(X^t, Y_j^t)$ requires less operations than computing $\delta_p(X, Y_j)$.

3 Enhanced-LRP Algorithms

Since LRP is a data dependent technique, the choice of Y_b determines the efficiency of the sufficient conditions and the performance of the algorithm. Our idea consists in rapidly finding a better guess of Y_b compared to that found by the LRP algorithm. If Y_b could be conveniently initialized previously to the matching process, the algorithm could benefit of it by deploying a more effective threshold D^t as well as by reducing the number of evaluated candidates at each iteration holding in the same time the property of finding the global minimum (i.e. exhaustiveness).

Hence, we propose to determine an estimation of the global minimum, $\tilde{Y}_b$, by means of any non-exhaustive algorithm which is able to perform this operation

at a small computational cost compared to that of the whole LRP algorithm. The choice of the non-exhaustive algorithm can be made between a number of methods proposed so far in literature, which reduce the search space in order to save computations (i.e. [2], [3], [4], [5]). In our implementation, which will be used for the experimental results proposed in this paper, we have chosen a standard two-stage coarse-to-fine algorithm. More precisely, at start-up image and template are sub-sampled by means of a binning operation, then the FS algorithm is launched on the reduced versions of image and template and a best candidate at sub-sampled resolution is determined. Finally, the search is refined in a neighborhood of the best match that has been found at full resolution, still by means of the FS algorithm, and $\tilde{Y}_b$ is initialized as the candidate referred to the best score obtained.

The use of a non-exhaustive method for estimating $\tilde{Y}_b$ requires a fixed overhead. Nevertheless, this method is more likely to find a candidate closer to the global minimum compared to the Y_b found by the LRP algorithm, especially at the lowest levels of resolution, where candidate vectors are reduced up to a very few elements (typically up to one at the lowest level). Once $\tilde{Y}_b$ has been determined by means of a fast non-exhaustive algorithm, we propose three different strategies for exploiting this information in the LRP framework, thus leading to three different algorithms, referred to as ELRP 1, ELRP 2 and ELRP 3. It is important to point out that, despite the use of a non-exhaustive method for the estimation of $\tilde{Y}_b$, the three proposed algorithms are all exhaustive since they all guarantee that the best candidate found is the global minimum. This is due to the fact that the proposed algorithms throughout the template matching process use the same bounding functions as LRP (i.e. 9, 11) but plugging in different candidates. Hence if a candidate is pruned by any of the conditions applied by the ELRP algorithms, then it is guaranteed to have a score higher than that of the global minimum.

3.1 ELRP 1

The non-exhaustive algorithm applied at the beginning gives us a candidate, $\tilde{Y}_b$, and its distance from the template X computed at highest level (i.e. level 0), $\delta_p(X, \tilde{Y}_b)$. Hence, at each step t, the first threshold D^t is determined by means of $\tilde{Y}_b$:

$$D^t = ||A||_{p,t} \cdot \delta_p(X, \tilde{Y}_b) \tag{12}$$

Then each candidate Y_j is tested with the pruning condition:

$$\delta_p(X^t, Y_j^t) > D^t \tag{13}$$

If (13) holds, candidate Y_j is pruned from the list. Thanks to this approach, differently from [1], at the lower level we avoided to execute an exhaustive search in order to initialize Y_b. Nevertheless, if $\tilde{Y}_b$ has been initialized badly, its use along the several pruning levels would result to bad efficiency of the pruning conditions, yielding to poor performance of the algorithm. Hence, at each level t, subsequently to the pruning test, an updating procedure of candidate $\tilde{Y}_b$ is

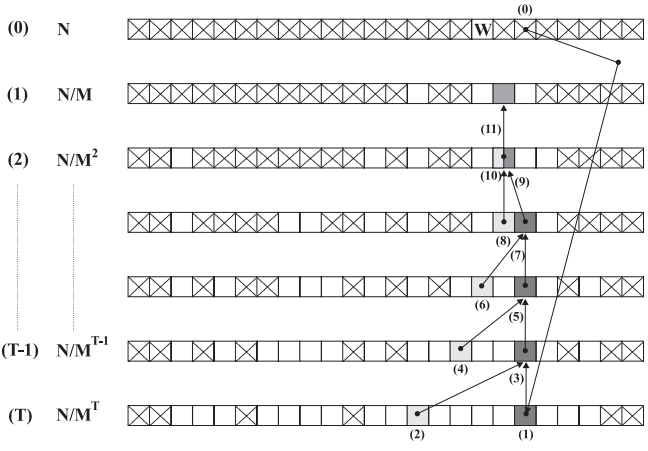

Fig. 1. A graphical visualization of the ELRP 1 algorithm. Each layer represents one of the $T + 1$ levels of the algorithm, while each block represents a candidate at a particular level of resolution. Gray squares correspond to candidate $Y_{b,l}$, while black squares correspond to candidate $\tilde{Y}_b$, each one at its respective stage of the algorithm. The candidate corresponding to the global minimum is denoted by letter W, while at each stage the pruned candidates are denoted with a crossed box.

applied. In particular, we search for the candidate still left in the list L with the minimum distance:

$$Y_{b,l} = \arg \min_{Y_j^t \in L} \{\delta_p(X^t, Y_j^t)\} \tag{14}$$

Then, if $Y_{b,l}$ and $\tilde{Y}_b$ don't coincide, they are compared by looking at their distance at highest resolution, and the candidate which concurs for the minimum distance is selected as the new $\tilde{Y}_b$ to be used for next step:

$$\tilde{Y}_b = \arg \min_{Y = \tilde{Y}_b, Y_{b,l}} \{\delta_p(X, Y)\} \tag{15}$$

It is worth noting that the determination of $Y_{b,l}$ in (14) only requires l additional tests, with l being the number of candidates still left in the list after the application of the pruning condition, since the term $\delta_p(X^t, Y_j^t)$ is already computed for the pruning test (13). Hence, it is expected that ELRP 1 holds the potential to speed-up the LRP method since, compared to LRP, the search space in which to search for the minimum is reduced by all the candidates already pruned by (13). Nevertheless, there might be some cases in which the effectiveness of the bounding threshold deployed by ELRP 1 (12) is lower than the corresponding one of LRP: this aspect, together with the need of an initial overhead to estimate $\tilde{Y}_b$, can cause overall a slower performance of ELRP 1 compared to LRP.

Figure 1 shows a graphical description of the ELRP 1 algorithm. At each level there are K candidates, represented by the squares. Going bottom up, thanks to the elimination conditions applied some candidates are iteratively pruned at

each level (denoted by the crossed boxes), while the others are propagated up to the top level, where the exhaustive search is determined on the left vectors and the best candidate is found (denoted by W). At start-up, i.e. step (0), candidate $\tilde{Y}_b$ is determined by means of the non-exhaustive technique and used to prune the candidates at level T. Then, at each level, after applying the pruning condition (13) on the candidates still in the list, the two vectors $\tilde{Y}_b$ and $Y_{b,l}$ (respectively, the black and the gray squares) are compared and the best between the two at highest level (i.e. 0) is chosen (15) as $\tilde{Y}_b$ for the upper level.

3.2 ELRP 2

As outlined in Section 2, at each step t the LRP algorithm performs an exhaustive search between the candidates left in the list in order to determine Y_b as the candidate which corresponds to the minimum distance, so that it can be used for the computation of the threshold D^t. Conversely, method ELRP 1 reduces this search at each step by means of $\tilde{Y}_b$ and a strategy for updating $\tilde{Y}_b$ by means of $Y_{b,l}$. A different approach is devised by keeping the exhaustive search performed between the candidates left at the lower level. With this approach, at each step t we first determine Y_b as the candidate yielding the minimum score at level t between the candidates left in the list. Then, $\tilde{Y}_b$ is updated as:

$$\tilde{Y}_b = \arg \min_{Y=\tilde{Y}_b, Y_b} \{\delta_p(X, Y)\} \tag{16}$$

It is worth to note that also this approach contains a strategy which allows for using a different candidate in case $\tilde{Y}_b$ is estimated badly by the initial non-exhaustive step. This is performed by means of the comparison in (16). It is also important to point out that, thanks to (16), the bounding terms deployed by ELRP 2 are guaranteed being always more (or, at worst, equally) effective compared than the corresponding ones devised by LRP. In terms of performance, this guarantees that in the worst case ELRP 2 will be slower than LRP only for the amount of time needed to carry out the estimation of $\tilde{Y}_b$, which, as previously mentioned, has to be small compared to the overall time required by the algorithm.

3.3 ELRP 3

In this third approach, we propose to change the rule by which the candidates are tested. Each candidate Y_j is compared (13) against all the possible pruning conditions which can be determined until either it is pruned, or the last level (i.e. level 0) is reached, which means the distance between X and Y_j must be computed. Of course, the succession of the pruning conditions follows the original algorithm, going from the lowest level (i.e. level T) up to the highest one. Hence, each candidate will be individually tested against up to T pruning conditions.

This approach allows to devise an updating procedure as follows. At the beginning of the algorithm, the thresholds D^t are computed at each level by means

of $\tilde{Y}_b$. When a candidate Y_j can not be skipped by any of the T conditions applied, the actual distance $\delta_p(X, Y_j)$ must be computed and the best candidate is updated as:

$$\tilde{Y}_b = \arg \min_{Y=\tilde{Y}_b, Y_j} \{\delta_p(X, Y)\} \tag{17}$$

Furthermore, the algorithm does not need anymore to keep a list where to store the candidates which have not been pruned *so far*, with extra savings for what means operations and memory requirements.

4 Experimental Results

This section compares the results obtained using the ELRP 1, ELRP 2 and ELRP 3 algorithms described in Section 3 with those yielded by the LRP algorithm. In the situation considered in [1], i.e. images affected by artificial noise, LRP yields notable speed-ups with respect to the FS algorithm. Nevertheless, we are also interested in testing the algorithms with more typical distortions found in real-world applications, i.e. those generated by slight changes in illumination and position. Thus, we carry out two experiments.

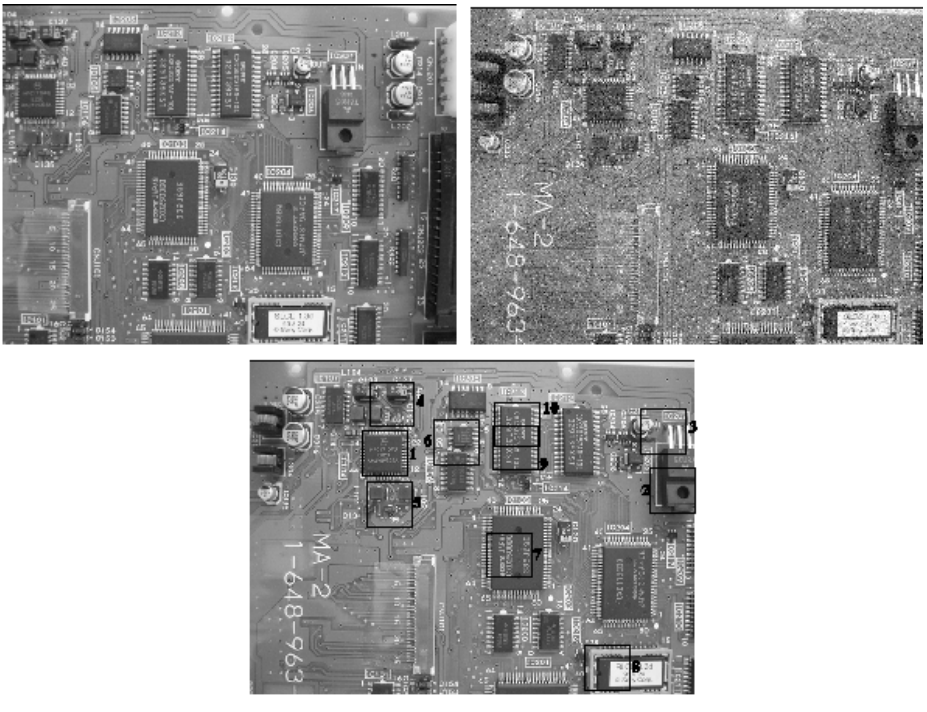

Fig. 2. The reference image used in Experiment 1 (top, left) and 2 (top, right). Bottom: the 10 templates used in the experiments are highlighted by black boxes.

In the first experiment 10 templates are extracted from an image as shown in fig. 2 (bottom), then the reference image is chosen as an image taken at a different time instant from a close-by position and with slightly different illumination conditions (see Figure 2, top left). Hence, the distortions between the templates and the reference image are generated by slight changes in pose and illumination, which can be regarded as the typical distortions found in real template matching scenarios. Then, in order to complete the experimental framework, we propose another experiment where the proposed algorithms are tested in the same conditions as in [1], that is, artificial noise is added to the image where the templates where extracted from (see Figure 2, top right). In particular, the introduced distortion is represented by i.i.d. double exponential (Laplace) distributed noise with parameters $\sigma = 10$ and $\mu = 0$. The 10 templates are the same as in experiment 1 (Figure 2, bottom).

To be coherent with the experimental framework proposed in [1], each template used for the two experiments is of size 64×64. This allows to have $M = 4 = 2 \times 2$, and the number of pruning levels $T = 6$. For the same reason, we use here the SSD function (i.e. $p = 2$). All the algorithms deploy incremental calculation techniques (i.e. [6]) for efficient computation of the transformed candidates at each level of resolution. Finally, the benchmark platform was a Linux workstation based on a *P4 3.056 GHz* processor; the algorithms were implemented in *C* and compiled using *Gcc* with optimization level *O*3.

Table 1. Measured speed-ups against FS in Experiment 1

Template	*LRP*	*ELRP*1	*ELRP*2	*ELRP*3
T1	11.3	17.9	16.4	17.9
T2	26.0	43.8	35.8	47.9
T3	7.1	9.9	9.5	9.8
T4	5.0	9.9	9.7	9.8
T5	18.2	28.6	25.2	29.8
T6	11.4	25.5	22.8	26.4
T7	13.1	25.0	22.2	25.7
T8	22.1	27.7	24.4	28.9
T9	11.5	12.8 $*$	14.2 $*$	24.0 $*$
T10	11.6	11.0 $*$	11.5 $*$	17.2 $*$
Mean	13.7	21.2	19.2	23.7
St.Dev.	6.5	11.0	8.4	11.2

Table 1 and Table 2 show the speed-ups (i.e. ratios of measured execution times) yielded by *LRP, ELRP 1, ELRP 2* and *ELRP 3* with regards to the FS SSD-based algorithm. As for the ELRP algorithms, the execution times include also the initial non-exhaustive step. Table 1 is relative to the dataset of experiment 1, while Table 2 shows the results concerning experiment 2. Furthermore, in the two tables symbol $*$ is used to highlight those cases where the non-exhaustive algorithm applied at the first step does not find the global minimum: overall, this happens in 3 cases out of 20. Nevertheless, though already

Table 2. Measured speed-ups against FS in Experiment 2

Template	LRP	$ELRP1$	$ELRP2$	$ELRP3$
T1	25.7	30.0	25.9	31.6
T2	36.5	39.5	32.7	42.7
T3	20.8	24.6	22.0	25.7
T4	20.8	23.9	21.5	24.9
T5	16.4	18.5	17.0	18.8
T6	21.8	25.0	22.2	26.2
T7	21.2	23.9	21.1	24.6
T8	24.7	32.5	28.1	34.8
T9	15.0	11.1 $*$	14.4 $*$	12.6 $*$
T10	14.6	15.5	14.5	15.6
Mean	21,8	24,5	21,9	25,8
St.Dev.	6,4	8,3	5,8	9,0

specified previously, we remind here that even in the cases marked with an $*$ the three ELRP algorithms are always guaranteed to find the global minimum. Finally, the two tables also report the mean speed-up and its standard deviation yielded by each algorithm over the dataset.

Table 1 shows that in most cases the ELRP algorithms yield notable speed-ups compared to LRP, while in the less favorable cases the behavior of the two classes of algorithms can be regarded as equivalent. In particular, as it can be inferred from Table 1, ELRP 1 and 2 are able to determine computational savings compared to LRP in 9 cases out of 10 while, for what means ELRP 3, it is always faster than LRP. Hence, ELRP 3 can be regarded as the most efficient algorithm in the considered template matching scenario. This can also be inferred by the mean speed-ups, which for all ELRPs is notably higher compared to that of LRP, and it is highest for ELRP3. Nevertheless, the standard deviation yielded by the ELRPs is also higher than that of LRP.

As for Experiment 2, Table 2 shows that even if the computational advantages of ELRPs with respect to LRP are less evident when artificial noise is the only distortion, they are still present. For instance, ELRP 1 and 3 yield to computational savings compared to LRP in 9 cases out of 10. Instead, ELRP 2 obtains results comparable to LRP, being faster in 6 cases out of 10 and with speed-ups often similar to those of the original technique. Overall, compared to LRP, the reported mean speed-ups are higher for ELRP 1 and ELRP 3, and almost equivalent for ELRP 2.

To complement previous results, Table 3 and Table 4 report the percentages of candidates skipped by the conditions applied at each pruning level $P_6, \cdots, P_t, \cdots, P_1$, with the last column showing the total percentage of skipped candidates. By looking at these tables, it can be seen that often the pruning conditions applied at the lowest levels by LRP results not to be effective: i.e., overall the first condition prunes less than 1.0% in 14 cases out of 20. Thus, the pruning load is pushed on the higher levels, which increases the total number of operations required by LRP. Conversely, in the ELRP algorithms the corresponding conditions are usually much more effective due to the estimation of

Table 3. Efficiency of the pruning conditions used by the algorithms in Experiment 1

T	Alg	$P_6\%$	$P_5\%$	$P_4\%$	$P_3\%$	$P_2\%$	$P_1\%$	$P_{TOT}\%$
T1	LRP	9.2	13.8	11.3	60.0	4.7	0.9	100.0
	ELRP1	49.7	13.9	16.4	14.6	4.5	0.9	100.0
	ELRP2	49.7	13.9	16.4	14.6	4.5	0.9	100.0
	ELRP3	49.7	13.9	16.4	14.6	4.5	0.9	100.0
T2	LRP	53.5	23.8	6.0	16.6	0.0	0.0	100.0
	ELRP1	92.9	4.2	2.2	0.7	0.0	0.0	100.0
	ELRP2	92.9	4.2	2.2	0.7	0.0	0.0	100.0
	ELRP3	92.9	4.2	2.2	0.7	0.0	0.0	100.0
T3	LRP	2.9	0.5	30.9	37.7	25.0	3.0	100.0
	ELRP1	11.7	9.6	20.0	41.5	16.9	0.3	100.0
	ELRP2	11.7	9.6	20.0	41.5	16.9	0.3	100.0
	ELRP3	11.7	9.6	20.0	41.5	16.9	0.3	100.0
T4	LRP	0.4	0.0	6.8	27.8	63.6	1.4	100.0
	ELRP1	8.8	6.6	30.5	37.7	15.5	0.8	100.0
	ELRP2	8.8	6.6	30.5	37.7	15.5	0.8	100.0
	ELRP3	8.8	6.6	30.5	37.7	15.5	0.8	100.0
T5	LRP	0.3	11.4	72.1	13.8	2.4	0.0	100.0
	ELRP1	27.6	37.7	30.4	4.1	0.1	0.0	100.0
	ELRP2	27.6	37.7	30.4	4.1	0.1	0.0	100.0
	ELRP3	27.6	37.7	30.4	4.1	0.1	0.0	100.0
T6	LRP	0.2	0.3	15.3	82.8	1.4	0.0	100.0
	ELRP1	15.4	27.6	52.3	4.6	0.1	0.0	100.0
	ELRP2	15.4	27.6	52.3	4.6	0.1	0.0	100.0
	ELRP3	15.4	27.6	52.3	4.6	0.1	0.0	100.0
T7	LRP	8.8	0.1	65.2	17.6	7.3	1.1	100.0
	ELRP1	57.0	15.5	15.7	10.0	1.6	0.2	100.0
	ELRP2	57.0	15.5	15.7	10.0	1.6	0.2	100.0
	ELRP3	57.0	15.5	15.7	10.0	1.6	0.2	100.0
T8	LRP	0.0	2.3	95.1	2.7	0.0	0.0	100.0
	ELRP1	14.3	42.3	40.8	2.7	0.0	0.0	100.0
	ELRP2	14.3	42.3	40.8	2.7	0.0	0.0	100.0
	ELRP3	14.3	42.3	40.8	2.7	0.0	0.0	100.0
T9	LRP	0.9	5.6	7.3	85.9	0.3	0.0	100.0
	ELRP1	15.1	10.3	22.3	45.6	6.7	0.0	100.0
	ELRP2	15.1	10.3	22.3	52.0	0.3	0.0	100.0
	ELRP3	29.7	28.6	28.1	12.9	0.6	0.1	100.0
T10	LRP	0.0	1.4	41.8	48.4	8.4	0.0	100.0
	ELRP1	7.0	6.0	16.1	63.3	7.4	0.1	100.0
	ELRP2	7.0	6.0	30.2	49.3	7.5	0.0	100.0
	ELRP3	13.4	21.5	31.6	32.2	1.3	0.1	100.0

candidate $\tilde{Y}_b$. This happens also when the initial step does not find the global minimum, which means that $\tilde{Y}_b$ still represents a better estimation of the best match (as regards the matching score) with respect to those used by LRP for the initial pruning conditions. For instance, in the worst case the first condition prunes 4.7%. Nevertheless, in a few cases the better effectiveness of the initial pruning conditions does not imply a speed-up with respect to LRP due to the computational overhead associated with the initial search.

For what means the behavior of the 3 proposed algorithms, ELRP 2 is the one which obtains the most similar performance compared to the original algorithm.

Table 4. Efficiency of the pruning conditions used by the algorithms in Experiment 2

T	Alg	$P_6\%$	$P_5\%$	$P_4\%$	$P_3\%$	$P_2\%$	$P_1\%$	$P_{TOT}\%$
T1	LRP	4.3	82.1	6.7	5.0	1.7	0.1	100.0
	ELRP1	76.2	10.3	6.7	5.0	1.7	0.1	100.0
	ELRP2	76.2	10.3	6.7	5.0	1.7	0.1	100.0
	ELRP3	76.2	10.3	6.7	5.0	1.7	0.1	100.0
T2	LRP	86.5	9.3	1.5	2.3	0.4	0.0	100.0
	ELRP1	93.5	2.2	1.5	2.3	0.4	0.0	100.0
	ELRP2	93.5	2.2	1.5	2.3	0.4	0.0	100.0
	ELRP3	93.5	2.2	1.5	2.3	0.4	0.0	100.0
T3	LRP	0.1	11.0	79.9	9.0	0.0	0.0	100.0
	ELRP1	24.1	23.1	43.8	9.0	0.0	0.0	100.0
	ELRP2	24.1	23.1	43.8	9.0	0.0	0.0	100.0
	ELRP3	24.1	23.1	43.8	9.0	0.0	0.0	100.0
T4	LRP	0.0	2.8	89.3	7.7	0.2	0.0	100.0
	ELRP1	19.3	18.8	54.0	7.7	0.2	0.0	100.0
	ELRP2	19.3	18.8	54.0	7.7	0.2	0.0	100.0
	ELRP3	19.3	18.8	54.0	7.7	0.2	0.0	100.0
T5	LRP	0.0	2.8	75.6	18.4	3.0	0.1	100.0
	ELRP1	17.8	23.1	37.6	18.4	3.0	0.1	100.0
	ELRP2	17.8	23.1	37.6	18.4	3.0	0.1	100.0
	ELRP3	17.8	23.1	37.6	18.4	3.0	0.1	100.0
T6	LRP	0.0	5.5	90.8	3.6	0.1	0.0	100.0
	ELRP1	9.5	29.5	57.2	3.6	0.1	0.0	100.0
	ELRP2	9.5	29.5	57.2	3.6	0.1	0.0	100.0
	ELRP3	9.5	29.5	57.2	3.6	0.1	0.0	100.0
T7	LRP	0.1	77.9	11.0	7.7	2.9	0.5	100.0
	ELRP1	65.1	12.9	11.0	7.7	2.9	0.5	100.0
	ELRP2	65.1	12.9	11.0	7.7	2.9	0.5	100.0
	ELRP3	65.1	12.9	11.0	7.7	2.9	0.5	100.0
T8	LRP	0.0	26.8	73.1	0.1	0.0	0.0	100.0
	ELRP1	15.2	64.9	19.8	0.1	0.0	0.0	100.0
	ELRP2	15.2	64.9	19.8	0.1	0.0	0.0	100.0
	ELRP3	15.2	64.9	19.8	0.1	0.0	0.0	100.0
T9	LRP	0.1	0.1	66.7	30.3	2.7	0.1	100.0
	ELRP1	4.7	2.7	9.0	80.8	2.7	0.1	100.0
	ELRP2	4.7	2.7	59.5	30.3	2.7	0.1	100.0
	ELRP3	17.7	14.1	25.4	34.0	8.1	0.5	100.0
T10	LRP	0.0	0.0	55.9	42.7	1.4	0.0	100.0
	ELRP1	9.4	18.1	28.4	42.7	1.4	0.0	100.0
	ELRP2	9.4	18.1	28.4	42.7	1.4	0.0	100.0
	ELRP3	9.4	18.1	28.4	42.7	1.4	0.0	100.0

In particular, it is able to obtain notable speed-ups in Experiment 1, not counterparted by particular negative performances. These results confirm the trend exposed in Section 3, that is the performance of ELRP 2 compared to LRP is lower bounded by the computational weight of the non-exhaustive overhead needed to initialize $\tilde{Y}_b$, which ought to be small. This can be seen by considering that in the worst case, i.e. T2 of Experiment 2, the speed-up of LRP to ELRP 2 is just 1.1. Even when the non-exhaustive initial step can not find the global

minimum (i.e. the $*$-cases in the tables) the behavior of the algorithm turns out to be at worst very close to that of LRP. In addition, it is also interesting to note that tables 3, 4 confirm that the aggregated candidates pruned by ELRP 2 at each conditions are always equal or higher than those pruned respectively by the conditions devised by LRP.

On the other hand, experimental results demonstrate that the strategies deployed by ELRP 1 and ELRP 3 are more effective than that of LRP since they both are able to obtain higher benefits in terms of computational savings compared to ELRP 2 on the average. In particular, ELRP 3 is the algorithm which yields the highest speed-ups and whose behavior is always favorable compared to LRP along the considered dataset, with the exception of a single instance where the speed-up obtained by ELRP 3 is slightly less than that of LRP. Hence, it can be regarded as the best algorithm, especially if the distortions between image and template are those typically found in real template matching applications.

5 Conclusions

We have shown how the LRP technique described in [1] can be enhanced by means of three different full-search equivalent algorithms (referred to as ELRP 1, ELRP 2 and ELRP 3), which deploy an initial estimation of a good candidate to be used in the pruning process. The proposed algorithms are able to yield significant speed-ups compared to the LRP technique in an experimental framework where distortions between templates and reference images are represented by changes in pose and illumination as well as by artificial noise. The variations proposed in our work do not increase the memory requirements of the original algorithm. Besides, it comes natural to expect that further improvements can be obtained if a more efficient non-exhaustive technique is used to determine the initial candidate, $\tilde{Y}_b$, in spite of the naive method we implemented for our tests.

References

1. Gharavi-Alkhansari, M.: A fast globally optimal algorithm for template matching using low-resolution pruning. IEEE Trans. Image Processing 10(4), 526–533 (2001)
2. Goshtasby, A.: 2-D and 3-D image registration for medical, remote sensing and industrial applications. Wiley, Chichester (2005)
3. Barnea, D., Silverman, H.: A class of algorithms for digital image registration. IEEE Trans. on Computers C-21(2), 179–186 (1972)
4. Li, W., Salari, E.: Successive elimination algorithm for motion estimation. IEEE Trans. on Image Processing 4(1), 105–107 (1995)
5. Wang, H., Mersereau, R.: Fast algorithms for the estimation of motion vectors. IEEE Trans. on Image Processing 8(3), 435–439 (1999)
6. Mc Donnel, M.: Box-filtering techniques. Computer Graphics and Image Processing 17, 65–70 (1981)