FRoTeMa: Fast and Robust Template Matching

Abdullah M. Moussa
Electrical Engineering Department
Faculty of Engineering,
Port-Said University,
Port-Said, Egypt

M. I. Habib
Electrical Engineering Department
Port-Said University, Egypt
Saudi Electronic University (SEU),
Saudi Arabia

Rawya Y. Rizk
Electrical Engineering Department
Faculty of Engineering,
Port-Said University,
Port-Said, Egypt

***Abstract*—Template matching is one of the most basic techniques in computer vision, where the algorithm should search for a template image T in an image to analyze I. This paper considers the rotation, scale, brightness and contrast invariant grayscale template matching problem. The proposed algorithm uses a sufficient condition for distinguishing between candidate matching positions and other positions that cannot provide a better degree of match with respect to the current best candidate. Such condition is used to significantly accelerate the search process by skipping unsuitable search locations without sacrificing exhaustive accuracy. Our proposed algorithm is compared with eight existing state-of-the-art techniques. Theoretical analysis and experiments on eight image datasets show that the proposed simple algorithm can maintain exhaustive accuracy while providing a significant speedup.**

***Keywords*—template matching; pattern matching; brightness-contrast invariance; rotation invariance; scale invariance; sufficient condition**

I. INTRODUCTION

Template matching is the task of seeking a given template in a given image. It is also known as pattern matching [1] and can be considered as one of the most basic operations in computer vision. Template matching is heavily used in signal, image and video processing. A lot of applications are based on template matching such as image denoising [2-4], motion estimation [5], and emotion recognition [6] to name a few. The literature on template matching contains a variety of algorithms. Based on the search accuracy, these algorithms can be broadly divided into two categories: approximate accuracy and exhaustive accuracy algorithms. The first category can achieve fast speedup at the cost of some loss of accuracy and often depends on one or more approximations, while exhaustive accuracy algorithms obtain fast speedup without losing accuracy. This category includes domain transformation techniques using FFT and bound based computation elimination algorithms in which inappropriate search locations are skipped from computations.

In general, regardless of the accuracy required, there are several ways to tackle the problem. Some techniques use the normalized cross-correlation (NCC) to handle brightness/contrast-invariant template matching. This can be made faster using bounded partial correlation [7, 8] or integral images [9]. Some other techniques depend on scale and rotation invariant key points [10], while some rely on previous segmentation/binarization of the image to analyze [11, 12], FFT [13], partial elimination [14], correlation transitivity [15], histogram [16, 17], circular and radial projections [18, 19], or auto-correlation [20]. However, in many cases, these algorithms cannot be used due to some image characteristics such as little grayscale variations and the resistance of the image to be binarized efficiently, or because they can't handle the rotation, scale, brightness and contrast (RSBC) invariant template matching problem, which is critical to several applications, or because they are unacceptably slow for many real world scenarios.

The tremendous amount of time-sensitive applications of template matching always pushes the need for faster techniques. So in this paper, we present a fast algorithm to solve RSBC invariant grayscale template matching problem. Our algorithm, named FRoTeMa (Fast and Robust Template Matching), is also rotation/scale-discriminating within a predefined level of accuracy, i.e., the method determines the scale and rotation angle of the template within the matching process. Although we argue on maintaining exhaustive accuracy only when both the image and the template are equivalent, experiments on eight image datasets show that the proposed simple algorithm can provide exhaustive-like search when some deteriorating conditions such as blurring or JPEG compression are applied on query images. We have compared our algorithm with eight state-of-the-art techniques. Experimental results reveal that, although its simplicity, the proposed algorithm also shows a significant speed-up.

The rest of the paper is organized as follows: Section II introduces the proposed template matching algorithm. Section III presents the complexity analysis of the new algorithm. In Section IV the experimental results are presented. Finally, conclusions are summarized in Section V.

II. THE PROPOSED FROTEMA ALGORITHM

FRoTeMa depends on bound comparisons to achieve fast performance. Instead of checking the similarity between each pixel in the template with a corresponding pixel in each candidate block in the image to analyze, FRoTeMa uses sufficient condition to distinguish between candidate matching positions and other positions that cannot provide a better degree of match with respect to the current best-matching one.

To describe the condition used, Let $\boldsymbol{I}$ be the image to analyze, $\boldsymbol{T}$ the template, $\boldsymbol{I_b}$ a block in $\boldsymbol{I}$ with the same size of $\boldsymbol{T}$, $\boldsymbol{T_{sb}}$ a sub-template block of size $P \times Q$ pixels, T_{sbs} the summation of pixel intensities within $\boldsymbol{T_{sb}}$, $\boldsymbol{I_{sb}}$ the corresponding sub-block in $\boldsymbol{I_b}$ and I_{sbs} the summation of pixel intensities within $\boldsymbol{I_{sb}}$. We consider $\boldsymbol{T}$ and $\boldsymbol{I_b}$ to be equivalent under brightness/contrast variation if there is a contrast

correction factor α > 0 and brightness correction factor β such that:

$$\boldsymbol{T} = \alpha \boldsymbol{I_b} + \beta \boldsymbol{1} \quad (1)$$

where $\boldsymbol{1}$ is the matrix of ones [18]. Similarly, we have:

$$\boldsymbol{T_{sb}} = \alpha \boldsymbol{I_{sb}} + \beta \boldsymbol{1} \quad (2)$$

Using (2) we conclude:

$$T_{sbs} = \alpha I_{sbs} + \beta PQ \quad (3)$$

If we segment $\boldsymbol{T}$ into smaller blocks similar to $\boldsymbol{T_{sb}}$, and form a new vector $\boldsymbol{T_v}$ to store summation of pixel intensities of each one of these blocks and then do the same for $\boldsymbol{I_b}$ to form a similar vector $\boldsymbol{I_v}$, we can use (3) to write:

$$\boldsymbol{T_v} = \alpha \boldsymbol{I_v} + \beta PQ\boldsymbol{1} \quad (4)$$

Eq. (4) represents the relation between $\boldsymbol{T_v}$ and $\boldsymbol{I_v}$ which is similar to the relation between $\boldsymbol{T}$ and $\boldsymbol{I_b}$ in Eq. (1). For a correct match block in $\boldsymbol{I}$, and because α, β, P and Q are constants, the normalized versions of $\boldsymbol{T_v}$ and $\boldsymbol{I_v}$ to zero mean and unit magnitude (denoted as $\boldsymbol{T_{vnor}}$ and $\boldsymbol{I_{vnor}}$ respectively) will be equivalent. Using a threshold ts to establish a minimum degree of similarity required to accept a match, we can write:

$$\boldsymbol{T_{vnor}} - \boldsymbol{I_{vnor}} \leq ts\boldsymbol{1} \quad (5)$$

Inequality (5) provides the condition used in the algorithm. That is, for each block in $\boldsymbol{I}$, if the condition is satisfied, this means that the block is a candidate to be a correct match, while if the condition is not reached; the block will be skipped without a need for further consideration. Using such condition not only can accelerate the search for the correct match, but also the usage of vector normalization will make the condition able to handle the possible variation in brightness/contrast between the template and candidate blocks.

A. Template Manipulation

Many template matching algorithms have to scan the image to analyze several times to be able to support rotation-scale invariance and this procedure may take much time to compute. FRoTeMa, Instead of doing so, relies on the template to handle the task. First, a prespecified range of scales and orientations should be set. The orientation and scale of the original template are changed according to each combination of angles and scales within the predefined ranges.

We have used the OpenCV [21] library for such rotation and scale operations. Then, a squared area in the middle of the resulting template is chosen and split into a grid of equally-spaced non-overlapping squared blocks that have a predefined side length. η_b is used as one of the algorithm parameters to specify such side length. The grid layout is utilized to make use of (5) while making the implementation more simple. This grid can be split into different number of blocks. If the template is exactly the same as a part of the input image, only one block is sufficient to use (5), while under brightness/contrast variation, we have to use more blocks to be able to apply vector normalization in order to remove the brightness/contrast difference. In our experiments, we have used a grid of nine blocks. The centered block in such grid is called as the head block (Fig. 1). The summation of pixel intensities within each block of the grid is calculated, and then these new nine values are normalized to zero mean and unit magnitude. After this procedure ends, a vector of nine normalized values for each angle-scale combination should be calculated. Notice that the η_b value must be the same for all calculated vectors. These vectors will be stored in a new vector $\boldsymbol{Tm_v}$. At the end of this computation, the resulting $\boldsymbol{Tm_v}$ vector is sorted with respect to the calculated values of head blocks.

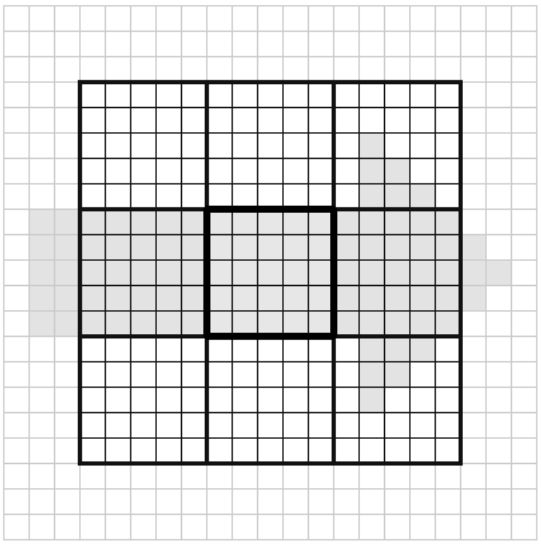

Fig. 1. A sample of 21×21 pixels template. FRoTeMa uses 9 blocks similar to big ones that appear in the middle. In this example, η_b is 5 pixels and head block is the one in the middle with a thicker edge

B. Image Scan

After computing $\boldsymbol{Tm_v}$, the image to analyze $\boldsymbol{I}$ is scanned sequentially in search of the template. Every pixel in $\boldsymbol{I}$ is checked only once by specifying nine squared blocks grid around it, calculating the summation of pixel intensities in such blocks and normalizing the resulting values to zero mean and unit magnitude. A new vector $\boldsymbol{P_v}$ is formed to always store the current pixel values. Similarly, the centered block of each pixel grid will be called as the head one. Notice that η_b value should be fixed for the template and $\boldsymbol{I}$ grids. To accelerate the process, the grid of the left-most pixel in each row of $\boldsymbol{I}$ is computed. Then this grid is used to calculate its neighbor pixel's grid by an overlapping process using a sliding window and such procedure is continued with the same criterion until reaching the right-most pixel.

In the perfect case, when the change between the template and the matching block (if any) is only in brightness, contrast and/or scale-orientation within the prespecified ranges, the $\boldsymbol{P_v}$ vector at one or more pixels will be identical with at least one of the vectors of $\boldsymbol{Tm_v}$. In such case, only pixels which have a $\boldsymbol{P_v}$ vector that is identical to one or more $\boldsymbol{Tm_v}$ vectors will be interesting for further investigation, while in the common case (as seen in many real world scenarios), when some deteriorating conditions such as blurring or JPEG compression are applied on query images, the $\boldsymbol{P_v}$ vector may not be equivalent to any $\boldsymbol{Tm_v}$ vector. So, a thresholding technique is used in this case to handle such possible gap. Two thresholds t_h and t_o are used. t_h sets the limit for the absolute difference between the current pixel's head block value and $\boldsymbol{Tm_v}$ head blocks' values. If one or more $\boldsymbol{Tm_v}$ vectors pass such condition, the remaining eight values in $\boldsymbol{P_v}$ are checked against

the corresponding eight values in the promising vector(s). t_o is used in such case to set the limit for the absolute difference between these values. Head blocks are handled uniquely as they should be the least distorted ones if the template is scaled or rotated with a value that is not covered in the prespecified ranges. While we can merge t_h and t_o values, t_h condition can prune the majority of the $\boldsymbol{Tm_v}$ vectors without a need for further processing. We call the overall of promising $\boldsymbol{Tm_v}$ vectors at all candidate pixels as candidate states. The amount of such states is important as it has a direct relation with speed of computation.

C. Candidate Pixels Handling

Each candidate pixel is in the center of a potential match block $\boldsymbol{b}$ in $\boldsymbol{I}$, and all of them have, by definition, one or more promising $\boldsymbol{Tm_v}$ vectors to be checked. When the current pixel is considered as a candidate one, the following is done with each one of its promising vectors: a new version $\mathbf{adj_t}$ of the original template is formed using the angle and scale that are associated with the vector. Before checking the matching between $\mathbf{adj_t}$ and $\boldsymbol{b}$, the possible variation in brightness and/or contrast between them should be handled. Several techniques may be used to handle this variation such as correlation coefficient and normalization. These techniques are efficient but they have expensive computational complexity. To avoid such drawback, another procedure is used based on (1) to estimate a contrast correction factor α_s and a brightness correction factor β_s. If such estimation relied on a few pixels, it will be sensitive to possible noise. So, a squared area $\boldsymbol{t_a}$ at the center of $\mathbf{adj_t}$ is chosen and intensity of $\boldsymbol{t_a}$ pixels is calculated after subtracting the mean value of them from each pixel to form $\boldsymbol{t_a}$`. A similar process is done for a corresponding squared area $\boldsymbol{I_a}$ in $\boldsymbol{b}$ to form $\boldsymbol{I_a}$` (We found experimentally that $\boldsymbol{t_a}$ and $\boldsymbol{I_a}$ of 11×11 pixels is far enough). This is made to cancel the possible variation in brightness between $\boldsymbol{t_a}$ and $\boldsymbol{I_a}$. To estimate α_s, a new matrix $\boldsymbol{C_m}$ is calculated as = $\boldsymbol{t_a}$` / $\boldsymbol{I_a}$`, and α_s is considered to be the median value of $\boldsymbol{C_m}$. α_s can be used to estimate β_s by using another matrix $\boldsymbol{B_m}$= $\boldsymbol{t_a}$ - ($\alpha_s \boldsymbol{I_a}$). The median value of $\boldsymbol{B_m}$ can be considered as an estimation of β_s.

To check whether $\boldsymbol{b}$ is a correct match block, the sum of absolute difference (SAD) is used between $\mathbf{adj_t}$ and $\boldsymbol{b}$ using the following formula:

$$SAD = \Sigma \left| (\alpha_s \boldsymbol{b} + \beta_s) - \mathbf{adj_t} \right| \quad (6)$$

The SAD value is calculated within the area of the largest circle that can be fit in the middle of $\mathbf{adj_t}$ as such circled area will always be within the template if it is rotated. The ratio between the SAD value and the sum of pixel intensities within the mentioned circled area is calculated for each candidate state in search of the minimum ratio. If the minimum ratio is below some threshold t_s, the template is considered to be detected at the current pixel. The algorithm has been applied successfully on rescaled versions of both of the image to analyze and the template to be a one fourth of their original sizes. i.e., a one level of pyramidal reduction has been used. We didn't have to apply the algorithm on the original scale to get accurate results except in a few cases.

III. Complexity Analysis

Let N be the number of pixels in the image to analyze $\boldsymbol{I}$, M the number of pixels in the template image, C the number of candidate states. G the number of angles and S the number of scales. Practically, we can consider that G and S have a complexity of O($\sqrt{M}$) and η_b is, by definition of O($\sqrt{M}$). As N is usually much larger than M, G, S and η_b, all operations that do not depend on N or C are neglected. At the image scan step, the operations used to prepare each nine-blocks grid with the help of the sliding window have a complexity of O($N\eta_b$) or O($N\sqrt{M}$), while the operations used to calculate $\boldsymbol{P_v}$ have a complexity of O(N). The computation used to match $\boldsymbol{P_v}$ with $\boldsymbol{Tm_v}$ is of O($N\ log\ GS$) or O($N\ log\ M$) as we depend on binary search technique when searching for promising vectors in $\boldsymbol{Tm_v}$. Also, the algorithm uses O(MC) computation to handle the candidate pixels. Thus, the overall complexity of the algorithm is O($N\sqrt{M} + MC$). As will be clarified in the experiments, the proposed algorithm achieves a great reduction in C from millions or hundreds of thousands to just hundreds or tens. Thus, the overall complexity can be reduced to O($N\sqrt{M}$) as opposed to, for example, O(NM) in [19].

IV. Experimental Results and Discussion

A. Experiments

To check FRoTeMa performance, several experiments have been executed using eight datasets. All experiments have been run on a Core i-5 (2.3-GHz PC) with 4 GB of RAM. To check the proposed algorithm robustness to simultaneous variations in rotation, scale, brightness and contrast, a dataset (K1) provided by Kim [22] has been used. It consists of six images and twelve templates. Each one of the six images includes all of the templates. This dataset tests only the rotation invariance. So, for each query, the scale, brightness and contrast of the queried image have been changed randomly within the ranges of 0.7 to 1.4, -25 to +25 and 0.85 to 1.15 of the original values respectively and each randomly changed query has been tested twice.

We have compared FRoTeMa with eight state-of-the-art algorithms ([8, 13-15, 18-20]). Experiments with ZEBC [8], FFT [13], PCE [14], TEA [15], OptA [20] and EGS [20] are performed on satellite image dataset (SI) of a low population density seaport (962x622 pixels) and aerial image dataset (AI) of a densely populated area (1549x2389 pixels) [20] using implementations provided by [23]. For each of the two datasets, templates of 10 different sizes are used (ranges from 31x31 to 121x121 for SI and 39x39 to 129x129 for AI). In each size 10 templates have been tested. The templates are obtained from a different view point at a different time. Therefore both datasets contain projective distortions as well as illumination variations.

Comparison with Forapro [18] and Ciratefi [19] was against five datasets. To test and compare robustness to rotation and scale variations, we have used another dataset (K2) provided by H. Kim [22]. It consists of eight images (400×400 pixels) and nine templates (71×71 pixels). Each one of the eight images includes all of the templates. Fig. 2 shows some examples of the algorithm output against K1 and K2 datasets.

(a) (b)

Fig. 2. (a) An example of FRoTeMa output when queried using the illustrated template from K1. (b) Another example of FRoTeMa output when queried using the illustrated template from K2 (note the discriminated angles and scales)

We have also used four datasets provided by K. Mikolajczyk [24] to check how robust our algorithm is to variations in brightness-contrast (Leuven dataset), focus blur (Bikes and Trees datasets) and JPEG compression (UBC dataset). Each one of these four sets consists of 6 images. The amount of distortion increases as the image number increases from 1 to 6. We have extracted 20 random non-overlapped templates (89×89 pixels) from the first image of each one of the four datasets, and in each set, we have tested FRoTeMa along with Forapro [18] and Ciratefi [19] against all of the 6 images using all of the 20 templates. So, we have evaluated 360 queries for each set. Fig. 3 illustrates some examples of FRoTeMa output against the tested Mikolajczyk datasets. When testing FRoTeMa against K1 and K2, we have checked 36 angles and 6 scales due to datasets challenge, while it was more appropriate when querying the other datasets to check one scale and orientation as they almost don't change. Parameters of the tested algorithms have been used as recommended by [18], [19] and [23]. Table I summarizes the total execution time on AI, SI and K2 datasets of FRoTeMa and the mentioned eight known algorithms.

Table I shows how FRoTeMa can provide an exceptional speed-up over all of the tested existing algorithms. The overall numbers of possible matches when considering the number of pixels, scales and orientations checked for K2, Leuven, Bikes, Trees and UBC datasets are 34560000, 540000, 700000, 700000 and 512000 respectively, while the average number of candidate states tested for each single query when testing FRoTeMa against such datasets was just 432.1, 42.3, 50.6, 283.7 and 31.6 states respectively. This justifies why FRoTeMa provides such great speed which makes a much difference regarding time-sensitive applications.

For K1, K2, Leuven, Bikes, Trees and UBC datasets, the output of the algorithms was considered correct if the overlap error is less than 15%. This is a very strict condition if we realize that Mikolajczyk et al. [25] consider that regions with up to 50% overlap error can still be considered matched successfully using robust descriptors.

TABLE I. TOTAL EXECUTION TIME (SEC) AND SPEED-UP RATIO ON AI, SI AND K2 DATASETS. FROTEMA IS COMPARED WITH EIGHT EXISTING ALGORITHMS (SUR REFERS TO SPEED-UP RATIO)

	FRoTeMa	TEA	OptA	EGS	PCE	FFT	ZEBC
AI	57	237	267	408	668	1675	1680
SUR	1	4.1	4.6	7.1	11.7	29.3	29.4
SI	13	49	58	64	124	153	239
SUR	1	3.7	4.4	4.9	9.5	11.7	18.3
		FRoTeMa		Forapro		Ciratefi	
K2		21		87		282	
SUR		1		4.1		13.4	

For AI and SI datasets, the ground truth is not available, so a match was considered correct if it was within (±4; ±4) pixels of the output location of the OptA [20] algorithm that claims an exhaustive-like accuracy. The performance of algorithms is given in terms of recall: TP/(TP+FN), where TP is True Positive and FN is False Negative. In Table II, performance of FRoTeMa against Forapro [18] and Ciratefi [19] in the tested datasets is illustrated.

TABLE II. PERCENT PERFORMANCE OF THE PROPOSED ALGORITHM AGAINST FORAPRO [18] AND CIRATEFI [19] IN THE TESTED DATASETS. * MEANS THAT THE EXPERIMENT WAS NOT DONE AND NOQ REFERS TO NUMBER OF QUERIES

Dataset	NoQ	FRoTeMa	Forapro	Ciratefi
AI (illumination)	100	100	*	*
SI (illumination)	100	100	*	*
K1 (RSBC)	144	100	*	*
Leuven (camera aperture)	120	100	100	100
Bikes(focus blur)	120	100	100	100
Trees(focus blur)	120	100	97.5	100
UBC (JPEG compression)	120	100	97.5	100
K2 (rotation/scale)	72	100	100	100

Table II shows that besides its superiority in speed, FRoTeMa exhibited 100% accuracy.

Fig. 3. Some examples of FRoTeMa output when tested against Mikolajczyk datasets [24]. For each couple of images, the upper one is the first image in the dataset and the lower one is the sixth image in the dataset when queried against the same template

B. Parameters

FRoTeMa sensitivity to different choices of η_b, t_h, t_o and t_s parameters has been tested. We have examined each one of the parameters while the others are fixed when searching for a template in one image of K2. We have checked 36 angles and 6 scales (from 0.7 to 1.4 of the original template size). In each of the following tables, the fixed parameters can be found in the first line.

TABLE III. SENSITIVITY TO η_b

t_h = 0.03, t_o = 0.06, t_s = 0.09		
η_b	Result (H refers to Hit)	No. of candidate states
3	H	1428
5	H	1997
7	H	2120
9	H	2280
11	H	2040
13	H	2232
15	H	2371
17	H	2449
19	H	2365
21	H	2034
23	H	2088

As we can see in Table III, changing η_b value almost has no effect of the result because no error was detected when varying its value. Also the resulting numbers of candidate states under η_b variation are comparable.

TABLE IV. SENSITIVITY TO t_h

η_b = 9, t_o = 0.04, t_s = 0.04		
t_h	Result (H refers to Hit)	No. of candidate states
0.005	Template Not Found	--
0.007	H	57
0.05	H	370
0.15	H	498
0.2	H	506
0.4	H	506
0.8	H	506
0.99	H	506

As of η_b we can see in Table IV that t_h doesn't affect the correctness of the results after some small value (0.007). Also we can note that the numbers of candidate states are comparable within a wide range of t_h values.

TABLE V. SENSITIVITY TO t_o

η_b = 9, t_h = 0.007, t_s = 0.04		
t_o	Result (H refers to Hit)	No. of candidate states
0.03	Template Not Found	--
0.04	H	57
0.06	H	555
0.08	H	2939
0.1	H	10536
0.12	H	30594
0.2	H	385920
0.4	H	1707086
0.6	H	2192439
0.8	H	2291491
0.99	H	2295690

Table V shows that t_o parameter has a broad range of values that will not affect the correctness (above 0.04). However, it has a big influence on the number of candidate states. So, large values of t_o may make the algorithm slower.

TABLE VI. SENSITIVITY TO t_s

t_h = 0.007, t_o = 0.04, η_b = 9		
t_s	Result (H refers to Hit)	No. of candidate states
0.03	Template Not Found	--
0.04	H	57
0.15	H	57
0.2	H	57
0.4	H	57
0.6	H	57
0.8	H	57
0.99	H	57

Table VI shows that after some value (around 0.04 in this test), the t_s value has no influence on the result. It also doesn't affect the number of candidate states. So, the choice of this value is not critical in terms of its effect on correctness or speed within a wide range of values.

V. CONCLUSIONS

Template matching plays a vital role in image processing and computer vision. It is heavily used in many applications in several diverse areas. In this Paper, We presented FRoTeMa, a new rotation, scale, brightness and contrast invariant template matching algorithm. The proposed method is also rotation/scale-discriminating within a predefined level of accuracy. Algorithm analysis and experimental results using eight datasets and comparisons with eight known methods show that FRoTeMa can provide an exceptional speed which is more suitable for time-sensitive applications while maintaining exhaustive-like search. Future work is aimed at checking whether the proposed method will be able to handle the color template matching problem and deal with color constancy. Also, it will be interesting to investigate the performance of the proposed algorithm after converting it to a parallel one.

REFERENCES

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

[2] Buades A, Coll B, Morel J-M. A Non-Local Algorithm for Image Denoising. IEEE Int. Conf. Comp. Vis. Pattern Recog. 2005; vol. 2: 60-65.

[3] Dabov K, Foi A, Katkovnik V, Egiazarian K. Image Denoising by Sparse 3D Transform-Domain Collaborative Filtering. IEEE Trans. Image Process. 2007; vol. 16: 2080-2095.

[4] Zhang R, Ouyang W, Cham WK. Image Deblocking Using Dual Adaptive Fir Wiener Filter in the DCT Transform Domain. in: Proceedings of IEEE Int'l Conf. Acoustics, Speech, and Signal Processing. 2009; 1181-1184.

[5] Shi X, Diwanji T, Mooney KE, Lin J, Feigenberg S, D'Souza WD, Mistry NN. Evaluation of Template Matching for Tumor Motion Management with Cine-MR Images in Lung Cancer Patients. Medical Physics. 2014; vol. 41.

[6] Mostafa E, Farag A, Shalaby A, Ali A, Gault T, Mahmoud A. Long Term Facial Parts Tracking in Thermal Imaging for Uncooperative Emotion Recognition. Sixth Int. Conf. on Biometrics: Theory, Applications and Systems (BTAS). 2013; 1-6.

[7] Di Stefano L, Mattoccia S, Tombari F. ZNCC-based template matching using bounded partial correlation. Pattern Recognition Letters. 2005; vol. 26: 2129-2134.

[8] Mattoccia S, Tombari F, Di Stefano L. Reliable rejection of mismatching candidates for efficient ZNCC template matching. in ICIP. IEEE. 2008; 849–852.

[9] Lewis JP. Fast template matching. Proc. Vision Interface. 1995; 120-123.

[10] Lowe DG. Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vis. 2004; vol. 60: 91-110.

[11] Torres-Méndez LA, Ruiz-Suárez JC, Sucar LE, Gómez G. Translation, rotation, and scale-invariant object recognition. IEEE Trans. Syst., Man, Cybern. 2000; vol. 30: 125-130.

[12] Kim WY, Yuan P. A practical pattern recognition system for translation, scale and rotation invariance. in: Proceedings of Int. Conf. Comput. Vis. Pattern Recognit. 1994; 391-396.

[13] William P, Saul T, William V, Brian F. Numerical Recipes: The Art of Scientific Computing. 3rd ed. Cambridge University Press; 2007.

[14] Mahmood A, Khan S. Correlation-coefficient-based fast template matching through partial elimination. IEEE Trans on Image Process. 2012; vol. 21, no. 4: 2099 –2108.

[15] Mahmood A, Khan S. Exploiting transitivity of correlation for fast template matching. IEEE Trans on Image Process. 2010; vol. 19, no. 8: 2190 –2200.

[16] Ullah F, Kaneko S. Using orientation codes for rotation-invariant template matching. Pattern Recognition. 2004; vol. 37: 201-209.

[17] Yoo J, Hwang SS, Kim SD, Ki MS, Cha J. Scale-invariant template matching using histogram of dominant gradients. Pattern Recognition. 2014; vol. 47: 3006-3018.

[18] Kim HY. Rotation-Discriminating Template Matching Based on Fourier Coefficients of Radial Projections with Robustness to Scaling and Partial Occlusion. Pattern Recognition. 2010; vol. 43: 859-872.

[19] Kim HY, Araújo SA. Grayscale template-matching invariant to rotation, scale, translation, brightness and contrast. IEEE Pacific-Rim Symp. Image and Video Tech., Lecture Notes in Computer Science. 2007; 100-113

[20] Mahmood A, Mian A, Owens R. Optimizing Auto-correlation for Fast Target Search in Large Search Space. arXiv:1407.3535v2 [cs.CV] 2014.

[21] Open Computer Vision Library, [Internet]. [cited 2015 September] Available from: http://sourceforge.net/projects/opencvlibrary/ .

[22] Kim, HY. [Internet]. [cited 2015 September] Available from: http://www.lps.usp.br/hae/software/

[23] Mahmood, A. [Internet]. [cited 2015 September] Available from: http://www.csse.uwa.edu.au/~arifm/OptA.htm

[24] Visual Geometry Group, Robotics Research Group, Department of Engineering Science, University of Oxford. [Internet]. [cited 2015 September] Available from: http://www.robots.ox.ac.uk/~vgg/data/

[25] Mikolajczyk K, Schmid C. A performance evaluation of local descriptors. IEEE Transactions on Pattern Analysis and Machine Intelligence, Institute of Electrical and Electronics Engineers (IEEE), 2005, 27 (10),pp.1615-1630.