International Journal of Computer Applications (0975 - 8887)
Volume 175 - No.10, August 2020

Visual Object Tracking using Sparse Representation and Interest Points in a Double Step Approach

Mohamad Hosein Davoodabadi Farahani
Department of Electrical Engineering
South Tehran Branch
Islamic Azad University
Tehran, Iran

Mohsen Khan Mohamadi
Department of Electrical Engineering
South Tehran Branch
Islamic Azad University
Tehran, Iran

Mojtaba Lotfizad
Department of Electrical Engineering
Tarbiat Modares University
Tehran, Iran

ABSTRACT

Nowadays, various approaches have been proposed for visual target tracking, amongst which the sparse representation-based approaches have shown efficiency. In this paper, a two-stage approach for visual target tracking is proposed. In the first stage, the approximate target position is determined based on the corner points and sparse representation. In the following, the appearance model memory of the target will be used to determine the exact location of the target to perform the target localization accurately. Experimental results demonstrate that the proposed approach can effectively handle challenges such as abrupt illumination variation, occlusion, and blurriness. Furthermore, based on the evaluations of the qualitative and quantitative results, the proposed algorithm is comparable in performance with other state-of-the-art algorithms.

General Terms

Computer Vision, Image processing

Keywords

Visual Tracking, Sparse Representation, Interest Point, Target Template, Memory Model

1. INTRODUCTION

The visual tracking problem is defined as the estimation of location or motion parameters of one or more targets in a video sequence and is known as a high-level task in computer vision science [1]. The drastic increase in computational power of processors in the last decade and the availability of high-resolution cameras along with the ever-growing need for automatic video analysis have led to the development of new algorithms in the visual tracking domain and made this topic a research hotspot in recent years [2]. The applications of visual tracking are broad and vary from automatic surveillance and motion analysis to human-computer interaction, augmented reality, etc. [3]. Although the development of new algorithms has addressed some of the challenges in the visual tracking problem, yet general target tracking is still a challenge to overcome. The challenges in visual tracking can be broadly classified into two groups: 1) the similarity between the appearance of target and its surrounding. 2) the changes in the appearance of a target itself, which can be due to abrupt motion, state changes, illumination changes, pose variation, and other destructive factors [1]. A visual tracking algorithm is generally comprised of three components:
1) target representation 2) dynamic model 3) search scheme. Nevertheless, it is possible to combine these parts [4]. Target representation can be thought of as the most crucial component of the visual tracking algorithms since it is directly responsible for overcoming challenges, i.e., choosing the best candidate in the presence of distractive factors. Besides, the objective function used for tracking is specified based on the target representation in use [3, 4, 5]. The dynamic model is usually used for the estimation of the possible target's state in order to decrease the computational complexity and reducing the area of the search space [6]. These models are trained either prior to or during the tracking process. Based on the search scheme, tracking algorithms are classified into deterministic and stochastic classes [7]. Having represented the target in a feature space, the tracking problem is reduced to a search process and can be solved using an optimization problem, i.e., maximizing or minimizing objective function based on the dissimilarity or similarity measure. Stochastic approaches usually optimize the objective function by considering the target observations in multiple frames in a Bayesian framework which, was first introduced for machine vision applications in the condensation algorithm [3]. In recent years, researchers have been using the convolutional neural network, biologically inspired methods, and meta-heuristic optimization techniques to track moving objects [8, 9, 10]. In this paper, a visual tracking algorithm using corner points and sparse representation is presented in which an accurate target localization approach based on the updating target's template using sparse representation is proposed. In addition, to overcome tracking challenges, such as

occlusion which has the highest degradation effect on the quality of tracking, the template is updated in a way that encapsulates the appearance changes of the target. The main contributions of this paper can be summarized as: 1) Use of memory template for handling tracking challenges. 2) Use of multiple templates with different learning rates to achieve higher flexibility. Experiments on the proposed algorithm prove its robustness and accuracy compared to its rivals, which will be discussed in the conclusion section. The paper is organized as follows: in the next section, the proposed algorithm is discussed by introducing each different part of the tracker. Then the proposed algorithm is evaluated with the presented measures, and in the following section is the conclusion of the paper. The rest of the paper is organized as follows: Section 2 describes related work on object tracking. The two-stage algorithm for object tracking is proposed in Section 3. Experimental results are reported and analyzed in Section 4. Finally, Section 5 presents the conclusion of the whole paper.

2. RELATED WORK

Designing a visual tracking algorithm that is simultaneously accurate and robust is a challenging task and becomes even more challenging with the presence of destructive factors such as scale and illumination variation, and rotation changes. Recent algorithms proposed in the literature use generative or discriminative approaches for target representation. In the generative approaches, only the target is modeled, while in the discriminative approaches, both the target and its background are modeled. Generative approaches formulate the tracking problem as searching for the region most similar to the target's model in the feature space, and generally, they do not require an extensive database for the training phase. These models are based on either templates or subspace models [3]. Kumar et al. [11] proposed a visual tracking algorithm based on the $\ell 1$ framework in which dictionaries consisting of templates of overlapping target segments have been used. Candidate's segments are sparsely represented in the dictionary space by solving the regularized squared minimization problem. The dictionary is updated based on the target similarity map, and as a result, the target's motion is estimated by combining the obtained information. Another approach is proposed in [12], which is based on a robust approach. In this paper, the challenge of partial occlusion is handled by modeling the target with its components. The weighted component-based approach is used in which the weights are calculated by the difference between the components' colors and the background. Discriminative approaches treat visual tracking as a classification problem, where the goal is to distinguish the target from the background [13, 14]. Hence, the information from target and background are extracted for training a discriminative classifier. These approaches generally require a massive database in order to achieve acceptable performance. The database can be either obtained through the tracking process (online) or offline [15, 16, 17, 18, 19]. Yang et al. [16] proposed a discriminative model for the target's appearance based on the super-pixel algorithm for distinguishing the target from its surrounding background. In the proposed approach, tracking is done by calculating a confidence map and finding the best candidate using a maximum a posteriori probability (MAP). Zhang et al. [17] proposed a discriminative feature selection algorithm in which the trained classifier directly relates its scores with each samples' importance. Then the trained classifier is used for discriminating between the target and the background. Furthermore, in order to make the algorithm robust against target loss in the tracking process, a two-stage algorithm for using the target's information in the first frame and the acquired online information is proposed. In [18] an algorithm with an appearance model based on features extracted from the multi-scale image feature space was proposed. A sparse measurement matrix for feature extraction was used, and the tracking was done with a binary classification utilizing a naive Bayes classifier which is updated online. In [20], a tracking method based on sparse representation in a particle filter framework is presented. Discriminating the target from its background is achieved by activating the target templates or the background templates in a linear system in a competitive manner, and the target's appearance variations are directly modeled as the representation error. Qi et al. [21] proposed a structure-aware local sparse coding algorithm that encodes a target candidate using templates with both global and local sparsity constraints. For robust tracking, they showed local regions of a candidate region should be encoded only with the corresponding local regions of the target templates that are the most similar from the global view. Thus, a more precise and discriminative sparse representation is obtained which accounts for appearance changes.

3. PROPOSED ALGORITHM

3.1 Key Points

The feature extraction and the applications based on these features are indispensable parts of computer vision and image processing applications such as image-stitching and object recognition as well as visual tracking. Features used for such applications are usually extracted from particular regions of the image known as interest points or key points. SURF [22] and SIFT [23] are among feature detectors that are accompanied by their specific descriptors. Features detected by these algorithms are scale-invariant, and their corresponding descriptor makes them robust to rotation as well. Despite the advantages of these algorithms, the computations required for detecting SIFT features is demanding, and henceforth these algorithms cannot be used in tracking applications effectively. The computational burden for detecting the SURF features is less demanding than the SIFT features, but it is still not applicable to fields with time-critical conditions such as online tracking. Key points specify salient regions in an image and can be obtained with less computational complexity. Despite SIFT and SURF, corner detectors only detect points and their corresponding positions. A corner can be defined as a crossing of two edges or as a point with two edges in its neighborhood with different directions. One of the characteristics which are essential for corner detectors is the ability to detect the same corner in many similar images under different conditions such as illumination changes, pose, and rotation variations. A corner detection procedure used in two well-known corner detectors [24] and [25] (Harris and KLT) is as follows:

(1) The gradient operator is applied in two directions x and y in order to obtain I_x and I_y (i.e. using [-1 0 1] and [-1; 0; 1] filters).

(2) Calculation of matrix *A* with the Gaussian mask is as follows:

$$A = \Sigma_u \Sigma_v w(u,v) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} = \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} \quad (1)$$

where $w(u,v)$ is as follows:

$$w(u,v) = \exp -\frac{(u^2+v^2)}{2\sigma^2} \quad (2)$$

Fig. 1: Matching corner points (left: target corner points in frame t, middle: candidate corner points in frame $t+1$, right: matched corner points in frame $t+1$)

In order to calculate the corners, matrix A must have two eigenvalues. If both eigenvalues are large and positive, then the pixel under consideration is a corner.

$$A = M_c = \lambda_1\lambda_2 - k(\lambda_1 + \lambda_2)^2 = \det(A) - ktrace^2(A) \quad (3)$$

Where k is the sensitivity factor and by solving equation Eq.3 the position of the feature point for Harris detector will be equivalent to the local maximums. For the KLT corner detector, the position of the corner is obtained by $min(\lambda_1, \lambda_2)$[25]

3.2 Sparse Representation

Sparse representation has shown its efficiency in many computer vision applications, including image reconstruction, denoising, pattern recognition, as well as tracking [26, 27, 28, 29]. By assuming $T = t_i$ as a series of patches extracted from the target centered at the detected key points which are represented as l_2 normalized vectors. The candidate key points y which are obtained by searching the current frame can be represented as a sparse linear combination of the dictionary's element of the target:

$$y \approx Ta = a_1t_1 + a_2t_2 + ... + a_nt_n \quad (4)$$

Where the target coefficient vector is defined as $a = (a_1, a_2, \cdots, a_n)^T$. Because of the destructive nature of noise and occlusion, using $y = Ta + \varepsilon$ will model these destructive factors where ε is called the error vector, and its nonzero elements specify any irregularity in the appearance caused by occlusion or illumination variations. Therefore with the partial template $I = [i_1,\ i_2,\ \ldots,\ i_d] \in \mathbb{R}^{d\times d}$, the noise is removed and will result in:

$$y = [T, I]\begin{bmatrix} a \\ e \end{bmatrix} \triangleq Bc \quad s.t. \quad c \geq 0 \quad (5)$$

The partial template I is equal to the identity matrix and $e = (e_1, e_2, \cdots, e_d)^T \in \mathbb{R}^d$ is the partial coefficient vector. The above equation does not have a unique solution for c and with the assumption that in every two consecutive frames, the changes in the appearance of the object only affect a limited number of pixels in the appearance model; therefore, limited nonzero e^+ and e^- would exist, and a sparse solution is required for c. Therefore, for solving this problem, the l_1 regularized squared minimization problem is used [29]:

$$\min_{c_i} \|y - B\,c_i\|_2^2 \quad s.t. \quad \|c_i\| \leq \lambda \quad (6)$$

Where $\|.\|_1$ and $\|.\|_2$ are ℓ_1 and ℓ_2 norms, respectively. In the proposed tracker, the object under consideration will be tracked by matching between the target feature points and the candidate feature points. Candidate feature points that are crossly matched with the target dictionary would be used for location estimation of the object. Thus if n is assumed to be the number of feature points in the current frame, only k of them would match the dictionary's atoms $(k < n)$ in such a way that each target's key point is represented as a linear combination of the candidate key points:

$$t_j = Ya_j + \varepsilon = a_{j1}y_1 + \cdots + a_{jn}y_n + c_{j1}i_1 + \cdots + c_{jn}i_n \quad (7)$$

Therefore, the correct candidates can be obtained by finding the maximum of the candidate dictionary's coefficients a_j. If a_{ji} coefficient is matched with the candidate dictionary's element as the largest value, then t_j key points of the target would be chosen as the corresponding matched points (Fig. 1). In the tracker proposed in reference [28] for removing wrong and noisy matches, a recursive approach is used, where each candidate key point is represented as a sparse linear combination of the target's points, and then the displacement of the target is calculated using the median displacement vector. This vector is calculated as follows:

$$x = x_0 + median(dx_i), \quad dx_i = x_i^c - x_i^t \quad (8)$$

$$y = y_0 + median(dy_i), \quad dy_i = y_i^c - y_i^t \quad (9)$$

Where (x_0, y_0) is the position of the target in the last frame and (dx_i, dy_i) are displacement vectors in x and y directions. In cases where the number of matched points is not sufficient, the median of the displacement vector does not show the right direction, and since there is no memory model of the target, the tracking would fail. To tackle this problem, an accurate target localization approach will be used.

3.3 Accurate Localization

The displacement vector obtained from matched corners in the first stage of the tracking (without considering the recursive stage) specifies possible locations of the object since the median vector does not formulate the object's motion and therefore is not an appropriate measure for determining the target's location. Using the target displacement vector, the target's template can be represented by the sparse representation of each new candidate target. This process is similar to that used in the previous stage but with the difference that the same target's template is used here to utilize most of the visual information. To do so, the target's templates and candidates are converted to the matrices with the same size as [20 by 20]. Target's dictionary B with some partial templates is like $B = [B\ I_n\ -I_n]$. Where $I \in \mathbb{R}^{n\times n}$is the identity matrix. As

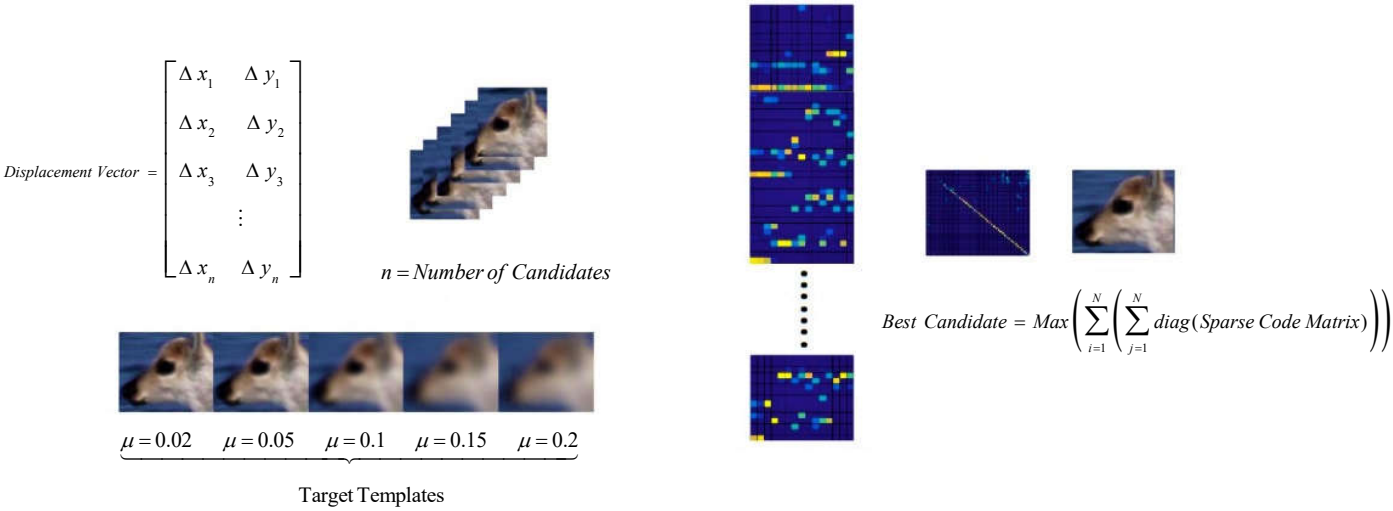

Fig. 2: Updating the target template using different coefficients (top image is the target template, and bottom images are the target templates stored by different learning factors)

mentioned before, this approach is suitable for the removal of noise and wrong information in the sparse representation framework. The candidate dictionary Ycan be obtained in the same similar way. Then just as the previous stage, each image patch's vector in the candidate dictionary Ycan be represented as a sparse linear combination of template dictionary patches B. The performance of the proposed algorithm can be improved by using the memory model. In order to improve the robustness of the algorithm, the proposed algorithm updates its target's template with the changing appearance of the target. Local patches and the whole template is updated in this procedure. In order to tackle the deformation challenge, five target templates were used, each of which is updated with their related coefficient so that in the tracking process the best target's template is used for matching with the object being changed (Fig. 2). Besides, the proposed update method enables the proposed tracker to tackle challenges such as illumination changes and changes in the target's appearance and state. The target's template in the proposed algorithm is updated as follows:

$$\rho_{t+1} = \rho_t(1-\mu) + \psi(\mu) \tag{10}$$

Where ρ_{t+1} is the acquired template model as the target in the frame $t+1$and μ is the update coefficient (learning rate) for the target's template and ψ is the target's template acquired from the current frame. In the final stage, after the calculation of sparse codes related to each area bounded by interest point, now the time for choosing the best candidate has come which is done by comparing the value of M in the following formula:

$$M = \sum_{i=1}^{N}\sum_{i=1}^{N} diag(Sparse\ Code\ Matrix) \tag{11}$$

4. EXPERIMENTAL RESULTS

In this section, the performance of the proposed algorithm, along with five other algorithms will be evaluated. For evaluation, the dataset [30] designed for this aim is utilized, which has been referenced by many pieces of research. The mentioned dataset contains many challenges such as illumination variation, pose variation, sudden movement of the target, and blurriness, which makes it proper for trackers' performance assessment. In total, sixteen sequences namely, *Subway, Fish, Man, Deer, David2, Coupon, Crowds, Crossing, Blurcar4, Blurbody, Blurowl, Dudek, Jumping, Skating1, Faceocc1*, and *Boy* have been used for the evaluation of the trackers' performance which cover a wide range of challenges. The default settings of each tracker were used to have a fair comparison. Visual tracker IPT [28] which was introduced previously as well as the tracker in [31] STC and [32] also known as FCT and IVT [33] and finally [34] which is briefly called VTD are compared with the proposed algorithm. Qualitative and quantitative comparisons have been carried out in order to illustrate the performance of the trackers. It is worth mentioning that in order to remove any distrust in the correctness of the simulation process of these algorithms the results of the VTD algorithm are obtained from reference [34] and the other algorithms are obtained from the respective author(s)' personal website. To solve the ℓ_1 minimization problem, a package called SPAMS [35] which is implemented in MATLAB is used with the regularization parameter set to $\lambda = 0.15$ for all experiments. As it was stated before, target's templates of size [20 20] and patches of size [5 5] are used for sparse representation leading to 16 feature vectors for each candidate. The proposed algorithm has been implemented in Matlab on an Intel 2.4 GHz Corei7 with 6 GB memory machine, which runs at 7.9 frames/sec. In order to compare and assess the performance of the proposed algorithm, qualitative and quantitative evaluations will be utilized.

4.1 Qualitative Evaluations

In the qualitative evaluation, visual comparisons of the tracked region (i.e., the bounded area for target representation) amongst different algorithms are made. The closer the tracked area to the reference target, the more robust will be the tracker under consideration. Fig. 3 shows qualitative comparisons of the proposed tracker along with five trackers. Sequences should be categorized based on the type of destructive factors that are associated with them to evaluate the weakness and strength of the trackers. For this task, nine different attributes have been

Fig. 3: Qualitative evaluation of trackers (sequences from top to bottom: subway, fish, man, deer, david2, coupon, crowds, crossing, blurcar4)

assigned to the related sequences, and one sequence can have up to six attributes. Table 1 illustrates the challenges associated with each sequence. *Illumination Variation* (IV) and *Scale Variation* (SV) indicate that the illumination and the scale of the desired

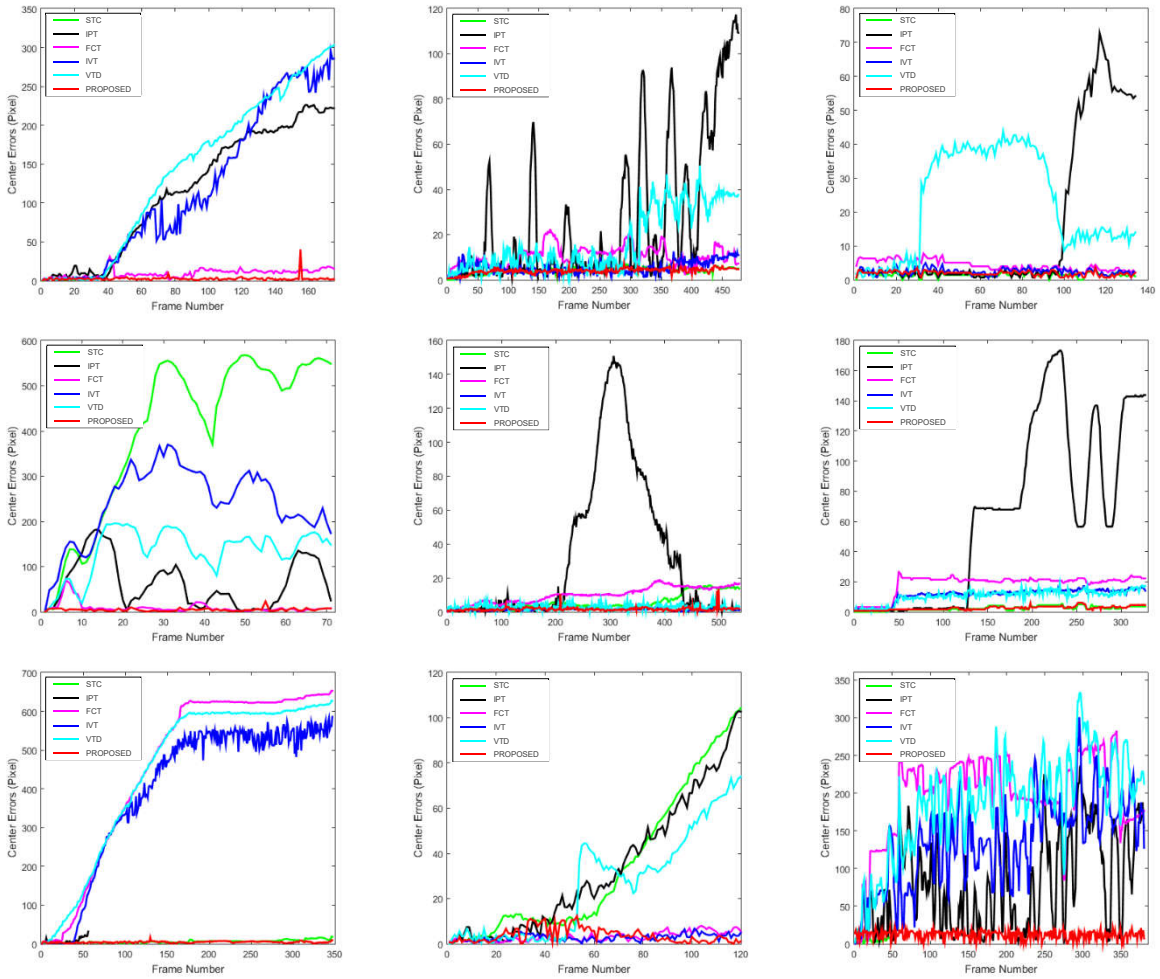

Fig. 4: Quantitative evaluation of trackers using Center Error Pixels (from left to right: 1st row: Subway, Fish, Man. 2nd row: Deer, david2, Coupon. 3rd row: Crowds, Crossing, Blurcar4.)

object in the target region that is shown with a bounding box are changed significantly. For instance, *Fish* is designed especially for examining trackers when the illumination varies in consecutive frames. In the following, *Occlusion* (OCC) and *Deformation* (DEF) are the most challenging destructive factors when the target is partially or fully occluded and the non-rigid object has undergone deformation respectively. *Fast Motion* (FM) and *Motion Blur* (MB) are prevalent phenomena in the dataset and are linked together in a way that the direct result of the abrupt motion of the object is blurriness. *In-Plane-Rotation* (IPR) and *Out-of-Plane-Rotation* (OPR) can also have a damaging effect on the sequences and occur when the target rotates in the image plane and out of the image plane respectively. *Background Clutters* (BC) is the last attribute and occurs when the background near the target has a similar color or texture as the target. The coupon is a perfect example that tests the trackers when the duplicate of the same object exists.

As Fig. 3 illustrates the Proposed algorithm exhibits a fine performance in comparison with other tracking algorithms, especially in challenging scenes such as *Deer* and *Bluecar4* which

Table 1. : Attributes of the Sequences [30]

Factor	Sequences
IV	Fish, Man, Crowds, Skating1
SV	Crossing, Blurowl, Dudek, Skating1
OCC	Subway, Coupon, Dudek, Skating1, FaceOcc1
DEF	Subway, Crowds, Crossing, BlurBody, Dudek, Skating1
MB	Deer, BlurCar4, BlurBody, Blurowl, Jumping, Boy
FM	Deer, BlurCar4, BlurBody, Blurowl, Dudek, Jumping, Boy
IPR	Deer, David2, BlurBody, Blurowl, Dudek, Boy
OPR	David2, Dudek, Skating1, Boy
BC	Subway, Deer, Coupon, Crowds, Crossing, Dudek, Skating1

are known for the fast motion and motion blur. It should be noted that one of the advantages of the proposed approach is to use the different target templates to handle various shape deformation of the target in the aforementioned sequences. Moreover, *Subway* which undergoes drastic occlusion is also a challenging case for

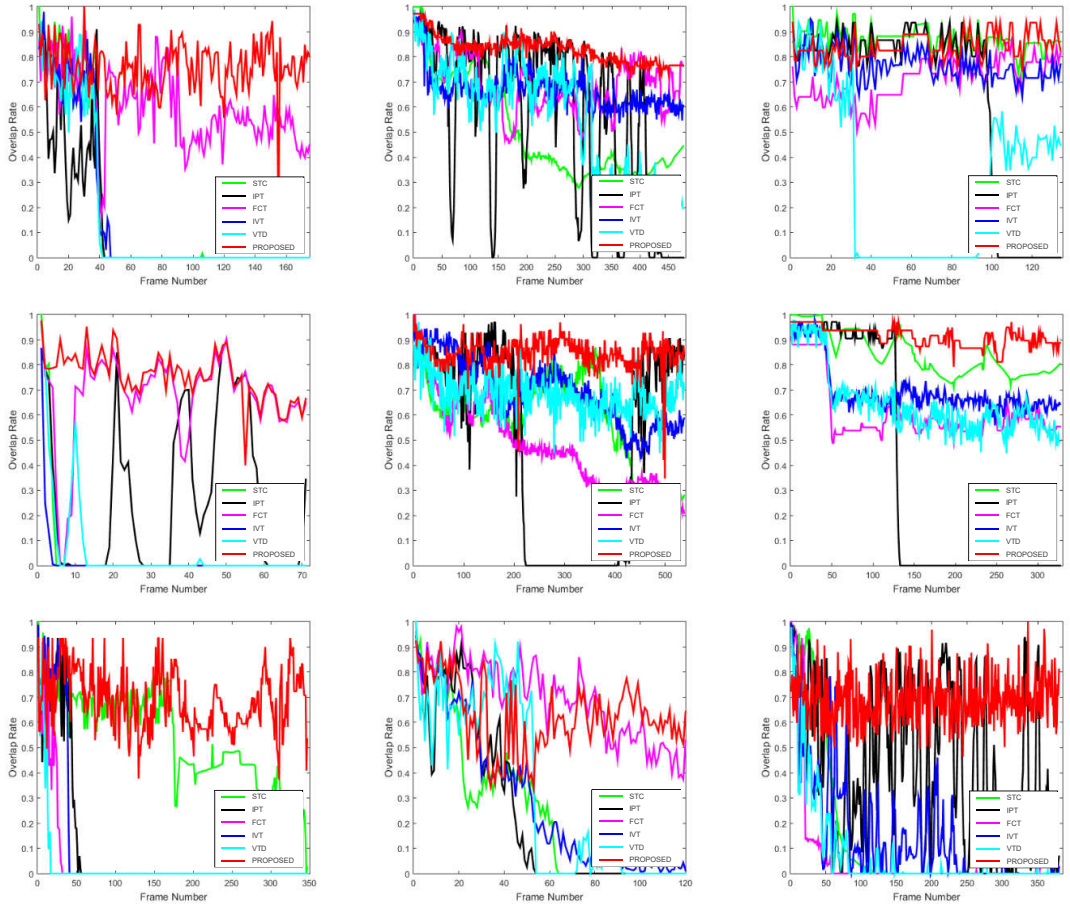

Fig. 5: Quantitative evaluation of trackers using overlap ratio rate (from left to right: 1st row: Subway, Fish, Man. 2nd row: Deer, david2, Coupon. 3rd row: Crowds, Crossing, Blurcar4.)

all of the trackers, and the performance of the proposed tracker has shown favorable results in this regard which outperforms its counterparts.

4.2 Quantitative Evaluations

In quantitative evaluation, several measures can be used which one of the common measures is Center Error. In fact, the distance between the center of the tracked region in each frame and center of the target's region in the given dataset will be determined. The less this distance, the better performance of the tracker. Another measure that is used is the overlap ratio of the tracker which calculates the intersection between the tracked region and target of interest provided by the dataset. This measure will be presented by a diagram whose vertical axis illustrates this value, which is between 0 and 1, and the horizontal values correspond to the tracked frames. If r_a equals to the tracked region by the proposed algorithm and r_b equals to the target region provided by dataset, then $G = \frac{|r_a \bigcap r_b|}{|r_a \bigcup r_b|}$ which $\cap$ and $\cup$ are intersection and union of the regions respectively, and $|.|$ equals the number of pixels. G is equal to the overlap rate in each frame. Fig. 4 and Fig. 5 illustrate diagrams of the center error pixels and overlap rate. Also, the average values of the center error and overlap rate for more and diverse sequences are given in Table 2 and Table 3 for which the best result is shown by red and the second-best value is shown by blue. As the Figure. 4 and Fig. 5 suggest, IPT has the weakest performance among other trackers and the reason lies in the fact that the mentioned algorithm lacks a proper target model leading to losing the object during the process, and this is also reflected in both tables, where the amount of center error and overlap rate is measured. In fact, the proposed tracker relies on the authentic target template approach which takes occlusion, motion blur, and deformation and or any destructive factor into account by using a memory model and a large search space that makes the algorithm resistant to losing the target in comparison with its most similar tracker (i.e., IPT).

5. CONCLUSION

In this paper, a visual tracking algorithm based on corner feature points and sparse representation is proposed. The proposed algorithm comprises a two-stage tracking procedure, which in the first stage, the approximate location of the target, and in the second stage, the exact location of the target will be determined. As is

Table 2. : Average values of the center error

Sequence	Proposed	IPT	STC	FCT	IVT	VTD
Subway	7.596	116.0	1244	9.562	119.5	141.3
Fish	4.819	26.55	3.977	10.47	5.384	16.79
Man	5.650	15.25	1.495	4.321	2.502	22.48
Deer	8.328	64.78	401.9	9.507	241.1	134.8
David2	6.763	32.67	5.577	10.08	1.426	2.855
Coupon	5.545	63.2	2.29	18.72	11.39	10.65
Crowds	5.816	1.170	6.14	459.1	396.8	447.3
Crossing	5.737	33.3	34.0	3.155	2.838	26.12
Blurcar4	6.286	79.1	475	196.2	134.1	185.2
Blurbody	12.432	36.927	148.033	35.095	162.792	146.9
Blurowl	29.681	81.568	50.092	117.666	112.609	252.297
Dudek	14.631	19.812	25.403	33.2	13.7315	10.296
Jumping	27.674	103.846	67.288	39.2085	12.031	41.387
Skating1	18.583	61.973	63.498	152.468	145.973	9.347
Faceocc1	27.264	63.612	31.985	35.591	16.810	20.202
Boy	9.145	32.888	25.920	7.2329	45.376	7.573

Table 3. : Average values of overlap rate

Sequence	Proposed	IPT	STC	FCT	IVT	VTD
Subway	0.759	0.116	0.242	0.602	0.163	0.156
Fish	0.825	0.558	0.510	0.692	0.669	0.556
Man	0.846	0.632	0.871	0.709	0.765	0.302
Deer	0.756	0.272	0.040	0.675	0.017	0.057
David2	0.847	0.481	0.588	0.474	0.696	0.685
Coupon	0.914	0.368	0.841	0.609	0.696	0.644
Crowds	0.687	0.612	0.523	0.043	0.089	0.024
Crossing	0.626	0.240	0.248	0.693	0.293	0.316
Blurcar4	0.745	0.410	0.134	0.055	0.144	0.072
Blurbody	0.791	0.560	0.1613	0.4716	0.0845	0.2362
Blurowl	0.523	0.328	0.1383	0.1363	0.0531	0.0482
Dudek	0.725	0.684	0.5874	0.5959	0.7478	0.799
Jumping	0.356	0.215	0.0693	0.1927	0.4351	0.1232
Skating1	0.8012	0.380	0.3513	0.1282	0.0799	0.525
Faceocc1	0.648	0.284	0.584	0.547	0.753	0.683
Boy	0.584	0.317	0.543	0.6138	0.2706	0.6257

said, the first stage contains the determination of the approximate location of the target using corner points and sparse representation, i.e., matching corner points in the current frame and achieving the displacement vector. By using the displacement vector obtained from the previous stage of tracking, some candidate regions will be extracted and then by using the sparse representation of the target template which is updated by different learning factors as a dictionary, the exact location of the target will be determined by considering the most similar candidate to the target template. As mentioned, a memory's model for the target's template is utilized in order to cope with challenges like illumination changes, occlusion, motion blur, etc. Experimental results indicate that the proposed algorithm has favorable results in comparison with other trackers.

6. REFERENCES

[1] Emilio Maggio and Andrea Cavallaro. *Video tracking: theory and practice*. 2011.

[2] Alper Yilmaz, Omar Javed, and Mubarak Shah. Object tracking: A survey. *Acm computing surveys (CSUR)*, 38(4):13, 2006.

[3] Qing Wang, Feng Chen, Wenli Xu, and Ming-Hsuan Yang. An experimental comparison of online object-tracking algorithms. In *Wavelets and Sparsity XIV*, volume 8138, page 81381A. International Society for Optics and Photonics.

[4] Qiang Guo and Chengdong Wu. Fast visual tracking using memory gradient pursuit algorithm. *J. Inf. Sci. Eng.*, 32(1):213–228, 2016.

[5] Hamd Ait Abdelali, Fedwa Essannouni, Leila Essannouni, and Driss Aboutajdine. Fast and robust object tracking via acceptreject color histogram-based method. *Journal of Visual Communication and Image Representation*, 34:219–229, 2016.

[6] Mohamad Hosein Davoodabadi Farahani and Mojtaba Lotfizad. Visual tracking via decision-based particle filtering based on sparse representation. *Journal of Electronic Imaging*, 27(4):043027, 2018.

[7] Wenhan Luo, Junliang Xing, Anton Milan, Xiaoqin Zhang, Wei Liu, Xiaowei Zhao, and Tae-Kyun Kim. Multiple object

tracking: A literature review. *arXiv preprint arXiv:1409.7618*, 2014.

[8] Mingliang Gao, Jin Shen, and Jun Jiang. Visual tracking using improved flower pollination algorithm. *Optik-International Journal for Light and Electron Optics*, 156:522–529, 2018.

[9] Shengping Zhang, Xiangyuan Lan, Hongxun Yao, Huiyu Zhou, Dacheng Tao, and Xuelong Li. A biologically inspired appearance model for robust visual tracking. *IEEE transactions on neural networks and learning systems*, 28(10):2357–2370, 2016.

[10] Yuankai Qi, Shengping Zhang, Lei Qin, Qingming Huang, Hongxun Yao, Jongwoo Lim, and Ming-Hsuan Yang. Hedging deep features for visual tracking. *IEEE transactions on pattern analysis and machine intelligence*, 41(5):1116–1130, 2018.

[11] Naresh Kumar and Priti Parate. Fragment-based real-time object tracking: A sparse representation approach. In *Image Processing (ICIP), 2012 19th IEEE International Conference on*, pages 433–436. IEEE.

[12] Jaideep Jeyakar, R Venkatesh Babu, and KR Ramakrishnan. Robust object tracking with background-weighted local kernels. *Computer Vision and Image Understanding*, 112(3):296–309, 2008.

[13] Jing Yang, Kaihua Zhang, and Qingshan Liu. Robust object tracking by online fisher discrimination boosting feature selection. *Computer Vision and Image Understanding*, 153:100–108, 2016.

[14] Shengping Zhang, Xiangyuan Lan, Yuankai Qi, and Pong C Yuen. Robust visual tracking via basis matching. *IEEE Transactions on Circuits and Systems for Video Technology*, 27(3):421–430, 2016.

[15] Chao Ma, Xiaokang Yang, Chongyang Zhang, and Ming-Hsuan Yang. Long-term correlation tracking. In *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference on*, pages 5388–5396. IEEE.

[16] Fan Yang, Huchuan Lu, and Ming-Hsuan Yang. Robust superpixel tracking. *IEEE Transactions on Image Processing*, 23(4):1639–1651, 2014.

[17] Kaihua Zhang, Lei Zhang, and Ming-Hsuan Yang. Real-time object tracking via online discriminative feature selection. *IEEE Transactions on Image Processing*, 22(12):4664–4677, 2013.

[18] Kaihua Zhang, Lei Zhang, and Ming-Hsuan Yang. Real-time compressive tracking. In *European conference on computer vision*, pages 864–877. Springer.

[19] Shengping Zhang, Huiyu Zhou, Feng Jiang, and Xuelong Li. Robust visual tracking using structurally random projection and weighted least squares. *IEEE Transactions on Circuits and Systems for Video Technology*, 25(11):1749–1760, 2015.

[20] Shengping Zhang, Hongxun Yao, Huiyu Zhou, Xin Sun, and Shaohui Liu. Robust visual tracking based on online learning sparse representation. *Neurocomputing*, 100:31–40, 2013.

[21] Yuankai Qi, Lei Qin, Jian Zhang, Shengping Zhang, Qingming Huang, and Ming-Hsuan Yang. Structure-aware local sparse coding for visual tracking. *IEEE Transactions on Image Processing*, 27(8):3857–3869, 2018.

[22] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359, 2008.

[23] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[24] Chris Harris and Mike Stephens. A combined corner and edge detector. In *Alvey vision conference*, volume 15, page 10.5244. Citeseer.

[25] Jianbo Shi. Good features to track. In *Computer Vision and Pattern Recognition, 1994. Proceedings CVPR'94., 1994 IEEE Computer Society Conference on*, pages 593–600. IEEE.

[26] Julien Mairal, Michael Elad, and Guillermo Sapiro. Sparse representation for color image restoration. *IEEE Transactions on image processing*, 17(1):53–69, 2008.

[27] John Wright, Allen Y Yang, Arvind Ganesh, S Shankar Sastry, and Yi Ma. Robust face recognition via sparse representation. *IEEE transactions on pattern analysis and machine intelligence*, 31(2):210–227, 2009.

[28] R Venkatesh Babu and Priti Parate. Robust tracking with interest points: A sparse representation approach. *Image and Vision Computing*, 33:44–56, 2015.

[29] Xue Mei and Haibin Ling. Robust visual tracking using $l1$ minimization. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 1436–1443. IEEE.

[30] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang. Online object tracking: A benchmark. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2411–2418, 2013.

[31] Kaihua Zhang, Lei Zhang, Qingshan Liu, David Zhang, and Ming-Hsuan Yang. Fast visual tracking via dense spatio-temporal context learning. In *European Conference on Computer Vision*, pages 127–141. Springer.

[32] Kaihua Zhang, Lei Zhang, and Ming-Hsuan Yang. Fast compressive tracking. *IEEE transactions on pattern analysis and machine intelligence*, 36(10):2002–2015, 2014.

[33] Jongwoo Lim, David A Ross, Ruei-Sung Lin, and Ming-Hsuan Yang. Incremental learning for visual tracking. In *Advances in neural information processing systems*, pages 793–800.

[34] Junseok Kwon and Kyoung Mu Lee. Visual tracking decomposition. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1269–1276. IEEE.

[35] Julien Mairal, F Bach, J Ponce, G Sapiro, R Jenatton, and G Obozinski. Spams: A sparse modeling software, v2. 3. *URL http://spams-devel. gforge. inria. fr/downloads. html*, 2012.