Unmanned Aerial Vehicle Video-based Target Tracking Algorithm Using Sparse Representation

Minjie Wan, Guohua Gu*, Weixian Qian, Kan Ren, Xavier Maldague, Senior Member IEEE, Qian Chen*

Abstract—Target tracking based on unmanned aerial vehicle (UAV) video is a significant technique in intelligent urban surveillance systems for smart city applications, such as smart transportation, road traffic monitoring, inspection of stolen vehicle, etc. In this paper, a vision-based target tracking algorithm aiming at locating UAV-captured targets, like pedestrian and vehicle, is proposed using sparse representation theory. First of all, each target candidate is sparsely represented in the subspace spanned by a joint dictionary. Then, the sparse representation coefficient is further constrained by an L_2 regularization based on the temporal consistency. To cope with the partial occlusion appearing in UAV videos, a Markov Random Field (MRF)-based binary support vector with contiguous occlusion constraint is introduced to our sparse representation model. For long-term tracking, the particle filter framework along with a dynamic template update scheme is designed. Both qualitative and quantitative experiments implemented on visible (Vis) and infrared (IR) UAV videos prove that the presented tracker can achieve better performances in terms of precision rate and success rate when compared with other state-of-the-art trackers.

Index Terms—Target tracking algorithm, UAV, sparse representation, temporal consistency, contiguous occlusion constraint, smart city.

I. Introduction

Intelligent urban surveillance and smart transportation systems are significant Internet of Things (IoT) applications for smart cities [1-2]. In these applications, UAV has become one of the most commonly used platforms where imaging devices (e.g., Vis and IR cameras) are installed to capture and locate objects of interest, e.g., vehicle or pedestrian, with the aid of real-time digital signal processors (DSPs). Thus, it is a burgeoning topic to develop high-accuracy object location algorithms for urban surveillance systems. Noticeably, visual tracking for UAV-captured target has attracted much attention during the past decades and has been extensively applied to road traffic data monitoring, inspection of stolen vehicle, security control of restricted area and suspicious human movement tracking [3-5]. In this paper, we focus on developing an intelligent target tracking algorithm to help locate UAV-captured objects, which is robust to the typical interferences, like partial occlusion, illumination variation, pose change, etc., in UAV scenes.

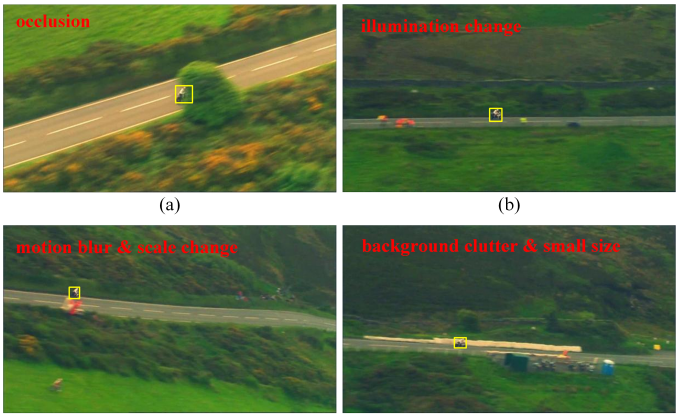

Fig.1. Frames from racer video. The ground truth of the racer is marked in yellow. Due to the influences of partial occlusion, illumination change, motion blur, scale change, background clutter and small target size, target tracking in UAV videos is still a hard task.

Target tracking based on UAV video analysis provides a more intuitive sense and a more convenient operation when compared with other radio frequency-based, wireless-based and microwave-based methods, thereby making it more and more popular in IoT applications. Given the initial state of target in the first frame by hand or using automatic object detection algorithm, the purpose of target tracking algorithm is to predict the following target states in the successive frames automatically and precisely. It should be noticed that UAV video based-target tracking do suffer from occlusion, illumination variation, scale change, motion blur, background clutter, noise disturbance, low resolution and small target size, which may easily cause the drift problem [6] (see an example

This work was supported in part by the National Natural Science Foundation of China under Grant 6167509 and Grant 61701233. (*Corresponding author: chenq@njust.edu.cn, gghnjust@mail.njust.edu.cn)

Minjie Wan, Guohua Gu, Weixian Qian, Kan Ren and Qian Chen are with the School of Electronic and Optical Engineering, Nanjing University of Science and Technology, Nanjing 210094, China. (minjiewan1992@njust.edu.cn, gghnjust@mail.njust.edu.cn, weixianqian_njust@yahoo.com, k.ren@njust.edu.cn).

Xavier Maldague is with the Department of Electrical and Computer Engineering, Computer Vision and Systems Laboratory, Laval University, 1065 av. de la Médecine, Quebec City, QC G1V 0A6, Canada. (Xavier.Maldague@gel.ulaval.ca)

shown in Fig.1). As a result, it is of great necessity for us to further investigate a robust target tracking algorithm to overcome the afore-discussed adverse factors.

During the past decades, sparse representation (also called sparse coding) derived from compressive sampling has shown striking advantages in artificial intelligence, e.g., face or speech recognition [7], image denoising [8], image super-resolution [9], motion segmentation [10] and target detection [11], etc. The main problem that sparse representation concerns is to exploit the sparsity of a signal with a sampling frequency that is lower than the Shannon-Nyquist rate so as to compress the storage memory to a great extent. Based on this theoretical basis, the tracking problem is converted into constructing a sparse representation of target candidate using templates with online update. Previous work made by Zhang et al. [12] has verified that using sparse coding is able to facilitate tracking robustness to image corruptions caused by illumination variation and partial occlusion. Mei et al. [13] proposed an L_1 tracker and it is the first study applying sparse representation scheme to target tracking problem. In their method, each target candidate is sparsely coded in a linear subspace composed of target and trivial templates. The solution of coding coefficient is obtained by solving an L_1-regularized least square problems and the candidate with the smallest reconstruction error is selected as the tracking result. Large numbers of experiments demonstrate that L_1 tracker can do well in videos with obvious occlusions, but the computing process of numerical optimization is computationally expensive, which makes it difficult to run L_1 tracker in real time. To alleviate this problem, a bounded particle resampling (BPR)- L_1 tracker using minimum error bound and occlusion detection was proposed by Mei et al. [14] later. This improved L_1 tracker aims at decreasing the running time by virtue of the minimum error bound computed from a linear least square equation. In addition, Bao et al. [15] improved the tracking precision of L_1 tracker by imposing an L_2 norm regularization on the coefficients associated with trivial templates and greatly increased the running speed by introducing an accelerated proximal gradient (APG) approach to solve the resulting L_1 norm related minimization problem with guaranteed quadratic convergence. What is more, Zhang et al. [16] proposed a multi-task sparse (MTT) learning tracker that regularizes the representation problem to enforce joint sparsity and learn the particle representation together via sparsity-inducing $L_{p,q}$ mixed norm. Zhang et al. further pointed out that L_1 tracker is a special case of the MTT formulation, and the overall computational complexity of MTT is generally attractive owing to the joint learning of particle representations.

Despite the great advantages achieved by the previous studies related to sparse representation-based trackers, there are still some important issues to be further investigated. On the one hand, target templates and trivial templates are mixed up to compose a template dictionary in these trackers. As pointed out by Bao et al. [15], it is highly possible for the sparse linear combination of trivial templates to include parts of the object. What is more, occlusion is simply modeled by an arbitrary sparse coefficient vector, but other useful properties of partial occlusion are ignored by most of the existing studies. Also, the solvers of the L_1 minimizations presented in different trackers should be especially considered. Otherwise, the tracking results will converge to local optima and the tracking drift problem will thus occur.

Enlightened by the afore-discussed problems, we propose a new sparse representation-based visual tracking method for UAV videos in this study. Firstly, each target candidate is linearly represented by a template dictionary. Different from the original L_1 tracker, the template dictionary utilized is made up of positive templates (i.e., target templates) and negative templates (i.e., background templates) in order to enhance the sparsity of representation coefficient. Considering the temporal consistency of object appearance, an L_2 regularization is imposed on the representation coefficient to reflect the similarity of appearance between the current frame and the previous frame. Next, a binary support vector is proposed to model the partial occlusion that may appear in UAV videos. On account of the prior knowledge that partial occlusion only occupies a small number of pixels and its shape is always continuous, an Ising model based on Markov Random Field (MRF) is employed to model the occlusion. To get the optimizations of the target representation coefficient and the binary support vector, an alternating algorithm involving APG method and graph cuts is developed. For long-term tracking, the tracking method is implemented under the framework of particle filter, and the template dictionary is also dynamically updated according to a metric called average cosine similarity.

The main contributions of this paper can be summarized as the following aspects:

① Target appearance is modeled by the sparse representation over a joint template dictionary;

② A temporal consistency-based L_2 regularization is proposed to constrain the representation coefficient of target;

③ An MRF-based binary support vector is exploited to model the partial occlusion by considering its sparsity and continuity;

④ The template dictionary is dynamically updated to overcome the appearance change of target.

The remainder of this paper is organized as follows: Section II brief reviews the related work about the research results of target tracking in recent years; in Section III, we introduce the theories of our study at great length; both qualitative and quantitative comparisons with the state-of-the-art tracking methods are implemented in Section IV; besides, a discussion about some key details in our algorithm is made in Section V; lastly, a conclusion as well as the future work is summarized in Section VI.

II. Related work

The recent decades have witnessed much progress in visual tracking [17]. In this section, we focus on discussing the appearance representation problem and the mainstream trackers relevant to this topic.

In general, target tracking algorithms can be categorized as either generative or discriminative based on their appearance models. Actually, there are various representation methods [18], the most typical representatives of which can be listed as: raw pixel representation, histogram representation, active contour representation, corner feature representation, etc. However, their robustness to the environmental interferences in UAV scenes is quite weak. Fortunately, dictionary-based sparse representation is found as a good choice owing to the following

facts: (1) it is proved to be robust to corruptions caused by illumination variation and occlusion; (2) the data-storage burden and computational burden of UAV system can be reduced since only few entries of the representation coefficient need to be recorded; (3) the templates utilized in this method are all extracted in the test images themselves without any offline trainings. Based on this consideration, dictionary-based sparse representation is employed in this paper.

Generative trackers learn a model to describe the target object and select the most similar image patch with minimal reconstruction error as the tracking result. Ho et al. [19] designed a tracking algorithm that uses a set of learned subspace model to address appearance change in unstable tracking environment. Instead of utilizing pre-trained subspace, an incremental visual tracking (IVT) algorithm was proposed by Ross et al. [20] by learning adaptive appearance model online. Leveraging the spatio-temporal relationship between target and its surrounding background, Zhang et al. [21] proposed a dense spatio-temporal context (STC) learning- based tracker to overcome the interference of occlusion. For the same purpose, Wang et al. [22] introduced a probability continuous outlier model (PCOM) to depict the continuous outliers existing in the linear representation model. To cope with partial occlusion and pose change during object motion, Kong et al. [23] presented a low-rank representation (LRR)-based tracker by means of decomposing the object observation matrix into a low-rank target appearance matrix and an occlusion matrix. Integrating low-rank and sparse representation theory, Zhang et al. [12] presented a consistent low-rank sparse (CLRS) tracker that jointly learns the particle representation to exploit the motion and appearance consistency of target. For the purpose of dealing with rigid and nonrigid deformations, Oron et al. [24] designed a joint model involving appearance and spatial configuration of pixels so as to predict the amount of local distortion of target [25]. Zhang et al. [26] developed a two-layer CNN without training (CNT) that directly exploits local structural and inner geometric layout information from data without manual tweaking.

Discriminative trackers regard tracking problem as a binary classification task that aims to find the decision boundary that best distinguishes the target from the background. Grabner et al. [27] proposed an online AdaBoost (OAB) classifier-based tracker integrating Haar feature, oriented histogram feature and local binary pattern feature. To alleviate cumulative tracking error, Babenko et al. [28] formulated online tracking under the multiple instance learning (MIL) framework where samples are considered in positive and negative sets. Considering sample importance, a weighted multiple instance learning (WMIL) tracker was further proposed by Zhang et al. [29]. Later, Zhang et al. [30] presented an online discriminative feature selection (ODFS) algorithm to exploit significant prior information of instance labels and the most correct positive instance using a novel formulation which is much simpler than MIL tracker. Owing to the rapid development of compressive sensing, a compressive tracking (CT) algorithm [25,31] was developed by training the naive Bayes classifier via features extracted from the multi-scale image feature space with data-independent basis. To mitigate the adverse effect of wrong labeling samples, Hare et al. [32] proposed an online structured output support vector machine (SVM) for robust tracking (Struck). Furthermore, Henriques et al. [33] discussed the circulant structure of the kernel matrix in SVM using fast Fourier transform algorithm. To sum up, the afore-discussed related work is summarized in Table I.

Table I. Summary of the related trackers.

Type	Name	Characteristic
Generative trackers	Ho's method	Learned subspace model
	IVT	Online low-dimensional subspace learning
	PCOM	Probability continuous outlier model
	STC	Dense spatio-temporal context learning
	LRR	Low-rank representation
	CLRST	Joint sparse representation with low-rank constraint
	Oran et al.	Joint model of appearance and spatial configuration of pixels
	CNT	Two-layer convolutional neural networks without offline trainings
Discriminative trackers	OAB	Online AdaBoost feature selection
	MIL	Online multiple instance learning
	WMIL	Bag probability function combining the weighted instance probability
	ODFS	Online discriminative feature selection
	CT	Multi scale-based compressive feature
	Struck	online structured output SVM
	Henriques's method	Circulant structure of SVM

III. THEORY

In this section, we focus on introducing the detailed theories of our algorithm, including target appearance model, solving method of optimization, particle filter framework as well as template update.

A. Target Appearance Model

In light of the theory of image sparse coding, a target is linearly spanned in the feature sub-space made up of template dictionary. Since occlusion is a fundamental factor in UAV tracking problem, an occlusion term is added to the target appearance model in this work. Thus, the appearance model proposed can be simply formulated as

$$Y=D\alpha+S+\varepsilon \quad (1)$$

where, Y stands for the target; D and α represent the template dictionary and representation coefficient, respectively; S is a binary support vector that denotes the occlusion; ε means the reconstruction error. For a more intuitive presentation, the core idea of our target appearance mode can be summarized using Fig.2. In the following sections, we would like to introduce the constraints imposed on α and S which make the reconstruction of target appearance more accurate.

Fig.2. Schematic diagram of the proposed appearance model using sparse representation.

1) Sparse Representation

In light of the sparse representation theory, a target image patch $Y_p \in R^q$ (q denotes the total pixel number of this patch) that is stacked into a column vector can be estimated by the homogenous positive templates and their corresponding representation coefficients [34] with a non-negativity constraint as

$$Y_p \cong \alpha_1 d_p^1+\alpha_2 d_p^2+\ldots+\alpha_{N_p} d_p^{N_p} = \left(d_p^1,d_p^2,\ldots,d_p^{N_p}\right)\left(\alpha_1,\alpha_2,\ldots,\alpha_{N_p}\right)^T = D_p\alpha_p, \quad \text{s.t., } \alpha_p \geq 0 \tag{2}$$

where, $D_p=\left(d_p^1,d_p^2,\ldots,d_p^{N_p}\right)\in R^{q\times N_p}$ and $\alpha_p=\left(\alpha_1,\alpha_2,\ldots,\alpha_{N_p}\right)^T \in R^{N_p}$ refer to the target dictionary and its representation coefficient vector; N_p means the number of positive templates in D_p. Ideally, the coefficient vector α_p is sparse, indicating that there are few non-zero entries in α_p. As for the non-negativity constraint, a real target can almost always be represented by the positive templates dominated non-negativity coefficients as the templates that are most similar to the tracking target are positively related to the target [13].

Suppose that the object of interest is manually selected in the first frame, the positive templates are initialized by randomly cropping image patches in a circle region around the object center, and the sampling region Ω_p is described as

$$\Omega_p=\left\{\left(x_p,y_p\right)\,|\,0<\sqrt{\left(x_p-x_0\right)^2+\left(y_p-y_0\right)^2}<r_1\right\} \tag{3}$$

where, $\left(x_p,y_p\right)$ is the center of target template; $\left(x_0,y_0\right)$ is the center of object cropped in the first frame; r_1 is the radius of sampling region.

Similarly, a background image patch $Y_n\in R^q$ can also be sparsely coded by the linear combination of negative templates and the corresponding representation coefficient as

$$Y_n \cong \alpha_1 d_n^1+\alpha_2 d_n^2+\ldots+\alpha_{N_n} d_n^{N_n} = \left(d_n^1,d_n^2,\ldots,d_n^{N_n}\right)\left(\alpha_1,\alpha_2,\ldots,\alpha_{N_n}\right)^T = D_n\alpha_n \quad \text{s.t., } \alpha_n \geq 0 \tag{4}$$

where, $D_n=\left(d_n^1,d_n^2,\ldots,d_n^{N_n}\right)\in R^{q\times N_n}$ and $\alpha_n=\left(\alpha_1,\alpha_2,\ldots,\alpha_{N_n}\right)^T \in R^{N_n}$ refer to the background dictionary and its representation coefficient vector; N_n is the number of negative templates in D_n.

Similar to the positive templates, the negative templates are initialized by randomly drawing samples within a doughnut-shaped image region Ω_n away from the object center, whose inner radius and external radius are denoted as r_2 and r_3:

$$\Omega_n=\left\{\left(x_n,y_n\right)|r_2<\sqrt{\left(x_n-x_0\right)^2+\left(y_n-y_0\right)^2}<r_3\right\} \tag{5}$$

where, $\left(x_n,y_n\right)$ is the center of background template.

By integrating both the target and background dictionaries, a randomly sampled target candidate $Y\in R^q$ can be sparsely represented by a joint template dictionary D as

$$Y \cong D_p\alpha_p+D_n\alpha_n=\left(D_p,D_n\right)\left(\alpha_p;\alpha_n\right)=D\alpha, \text{ s.t. } \alpha\geq 0 \tag{6}$$

where, $D=\left(D_p,D_n\right)\in R^{q\times N}$ and $\alpha=\left(\alpha_p;\alpha_n\right)\in R^N$ stand for the joint template dictionary and the joint representation coefficient, respectively; $N=N_p+N_n$ means the total number of templates.

In our algorithm, once a target candidate is sampled from the current frame, it is directly coded using the joint dictionary D, rather than coded by D_p and D_n respectively. We argue that the joint representation coefficient can be sparser when this joint dictionary is employed. Specifically speaking, if an image patch is the target, it can only be sparsely represented by D_p, i.e., there are few non-zero entries in α_p with α_n being a zero-vector, vice versa. Fig. 3 gives an intuitive illustration of the above-discussed sparse representation model.

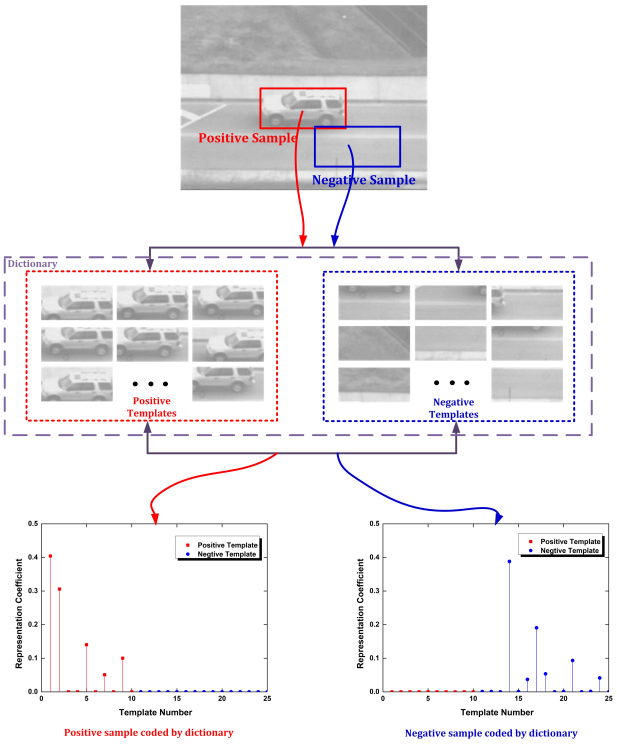

Fig.3. Schematic diagram of the sparse representation model.

Now, the preliminary appearance model using sparse representation can be formulated as follows

$$Y=D\alpha+\varepsilon, \text{ s.t. } \alpha\geq 0 \tag{7}$$

Equivalently, the sparse representation of target candidate Y can be written as the minimum error reconstruction error with a regularized L_1 minimization function as

$$\min_{\alpha}\frac{1}{2}\|Y-D\alpha\|_2^2+\lambda\|\alpha\|_1, \text{ s.t. } \alpha\geq 0 \tag{8}$$

where, $\|\cdot\|_1$ denotes L_1 norm; $\|\cdot\|_2$ denotes L_2 norm; λ is a constant that controls the sparsity of α.

2) Temporal Consistency

To further constrain the optimization function proposed in Eq.(8), temporal consistency is taken into consideration in object representation. As mentioned in the previous work made by Zhang et al. [12], object appearance will not change obviously during the two adjacent frames. Motivated by this, we thus point out that the sparse representation coefficients calculated in the current frame and the previous frame should also be similar to each other. Here, we compare the representation coefficient of target candidate α with that of the tracking result in the previous frame α_0 via L_2 norm, i.e., $\Delta\alpha=\|\alpha-\alpha_0\|_2$.

To embed the temporal consistency proposed above to the optimization function, we modify Eq.(8) as follows

$$\min_{\alpha}\frac{1}{2}\|Y-D\alpha\|_2^2+\lambda\|\alpha\|_1+\gamma\|\alpha-\alpha_0\|_2^2, \text{ s.t. } \alpha\geq 0 \tag{9}$$

where, γ is a constant that controls the value of $\Delta\alpha$.

It is worth noting that there are two special cases where the temporal consistency term should be set to invalid (λ=0):

① We start to compute Δα from the third frame, so this term is invalid in the second frame;
② The update of positive dictionary will cause significant difference of the representation coefficients between adjacent frames, so this term is set to be invalid once the update occurs. Since negative templates almost have no influence on the coding coefficient of target, which will be demonstrated in Section V.A.2, we do not make additional settings for λ when negative templates are updated..

3) Occlusion Model

In UVA videos, the object of interest (usually, vehicles or pedestrians) is easily corrupted by partial or full occlusions generated by shadow or plant, and this often leads to unpredictable tracking errors. In this paper, we try to address the partial occlusion problem.

Let us define a binary support vector $S\in\{0,1\}^q$, in which the non-zero entries stand for the occlusion pixels.

We regard those non-zero entries as outliers in the image patch of target candidate that cannot be fitted into the linear representation of dictionary templates, so Eq.(9) needs to be improved using a projection operator $\Phi_S(\cdot)$ in related to the proposed binary support vector S as

$$\min_{\alpha}\frac{1}{2}\|\Phi_S(Y\text{-}D\alpha)\|_2^2+\lambda\|\alpha\|_1+\gamma\|\alpha\text{-}\alpha_0\|_2^2,\ \text{s.t. }\alpha\geq 0 \tag{10}$$

where, the projection operator is defined as

$$\Phi_S(x_i)=\begin{cases}x_i, & \text{if } S_i=0\\ 0, & \text{others}\end{cases} \tag{11}$$

For UAV videos, we have two pieces of fundamental prior knowledge of the partial occlusion: ① it occupies a small number of pixels; ② it should be continuously distributed in the image patch of target. Motivated by MRF [35], we can model the partial occlusion via Ising model. First, the binary support vector S is mapped back to its matrix form $S^*\in\{0,1\}^{m\times n}$ (m and n are the width and height of each sample image). Then, we consider a graph $G_{S^*}=(V,E)$, where V is the set of vertices representing all the $q=m\times n$ pixels in S^* and E denotes the set of edges connecting spatially or temporally neighboring pixels. Based on the assumptions made above, an energy function of G_{S^*} can be formulated as

$$\Psi(G_{S^*})=\sum_{it\in V}\mu_{it}(S_{it}^*)+\sum_{(it,kl)\in E}\upsilon_{it,kl}|S_{it}^*-S_{kl}^*| \tag{12}$$

where, $\mu_{it}\in\{0,1\}$ denotes the unary potential of vertex S_{it}^*; $\upsilon_{it,kl}>0$ denotes the degree of dependency between S_{it}^* and S_{kl}^*. Since we treat $S_{it}^*=1$ as the occlusion pixel in the image patch of target candidate, the unary potential μ_{it} is defined as

$$\mu_{it}(S_{it}^*)=\begin{cases}\upsilon_{it}, & \text{if } S_{it}^*=1\\ 0, & \text{others}\end{cases} \tag{13}$$

where, $\upsilon_{it}>0$ penalizes $S_{it}^*=1$. In this study, both $\upsilon_{it,kl}$ and υ_{it} are set as constants because the vertices and edges in S^* can be approximately seen as isotropic. As a result, the energy function in Eq.(12) can be simply rewritten as

$$\Psi(G_{S^*})=\eta\sum_{it\in V}S_{it}^*+\varsigma\sum_{(it,kl)\in E}|S_{it}^*-S_{kl}^*| \tag{14}$$

where, η and ς are two positive constants that represent $\upsilon_{it,kl}$ and υ_{it}, respectively.

To make our occlusion model simpler and clearer, we further rewrite Eq.(14) as [36]

$$\Psi(G_{S^*})=\eta\|S\|_1+\varsigma\|\Theta S\|_1 \tag{15}$$

where, S is the column vector form of S^* and Θ is the node-edge incidence matrix of G_{S^*}.

4) Optimization Function

Based on the models proposed from Section III.A.1- Section III.A.3, we can obtain the optimization function of our appearance model that comprehensively considers the sparse representation, temporal consistency as well as partial occlusion. And, its final mathematical expression is given as

$$\begin{aligned}\min_{\alpha,S\in\{0,1\}}&\frac{1}{2}\|\Phi_S(Y\text{-}D\alpha)\|_2^2+\lambda\|\alpha\|_1+\\ &\gamma\|\alpha\text{-}\alpha_0\|_2^2+\eta\|S\|_1+\varsigma\|\Theta S\|_1,\ \text{s.t. }\alpha\geq 0\end{aligned} \tag{16}$$

As can be seen from Eq.(16), the following task for us is to estimate the optimal solution of α and S. We just present the optimization function of model here, and the detailed algorithm that solves Eq.(16) will be discussed in the next section.

B. Solution of Optimization Model

Through observing the optimization function formulated in Eq.(16), we find that it is nonconvex and there are both discrete and continuous variables in it. Since it is hard to solve α and S at the same time, we choose to adopt an alternating algorithm that separates Eq.(16) over α and S into two independent steps.

1) Estimation of Sparse Representation Coefficient

Given an estimate of the binary support vector $\hat{S}$, the optimization problem in Eq.(16) turns to be an L_1-regularized problem over α:

$$\min_{\alpha}\frac{1}{2}\|\Phi_{\hat{S}}(Y\text{-}D\alpha)\|_2^2+\lambda\|\alpha\|_1+\gamma\|\alpha\text{-}\alpha_0\|_2^2,\ \text{s.t. }\alpha\geq 0 \tag{17}$$

Considering the non-negativity constraint is inconvenient for solution, we use an additional term $1_{R^+}(\alpha)$ that is defined as Eq.(18) to replace it.

$$1_{R^+}=\begin{cases}0, & \text{if } \alpha\geq 0\\ +\infty, & \text{others}\end{cases} \tag{18}$$

Here, we further decompose Eq.(17) as the following two functions:

$$f(\alpha)=\frac{1}{2}\|\Phi_{\hat{S}}(Y\text{-}D\alpha)\|_2^2+\gamma\|\alpha\text{-}\alpha_0\|_2^2 \tag{19}$$

$$g(\alpha)=\lambda\|\alpha\|_1+1_{R^+}(\alpha) \tag{20}$$

As can be observed from Eqs.(19-20), Eq.(17) is divided into two parts: 1) the first part f(α) is composed of two L_2 norms, and they are both differentiable; 2) the second part g(α) is an L_1 constraint term, and it is a non-smooth but convex function. Wang et al. [37] stated that this kind of minimization problem can be solved by APG method with quadratic convergence.

For f(α), its gradient function ∇f(α) is computed as

$$\nabla f(\alpha)=M^T(M\alpha-C)+2\gamma\alpha \tag{21}$$

where, $M=\Phi_{\hat{S}}(D)$ and $C=\Phi_{\hat{S}}(y)$. Then, we prove that the gradient of f(α) is Lipchitz continuous, since for any $\alpha_1,\alpha_2\in R^N$.

$$\begin{aligned}&\|\nabla f(\alpha_1)-\nabla f(\alpha_2)\|\\ &=\left\|\left(M^T(M\alpha_1-C)+2\gamma\alpha_1\right)\right.\\ &\left.-\left(M^T(M\alpha_2-C)+2\gamma\alpha_2\right)\right\|\\ &=\left\|M^TM(\alpha_1-\alpha_2)+2\gamma(\alpha_1-\alpha_2)\right\|\\ &\leq\left\|M^TM(\alpha_1-\alpha_2)\right\|+\|2\gamma(\alpha_1-\alpha_2)\|\end{aligned} \tag{22}$$

$$\leq \left(\left\| M^T M \right\|_2 + 2\gamma \right) \left\| \alpha_1 - \alpha_2 \right\|$$

Thus, the Lipchitz constant of $f(\alpha)$ is $L=\left\| M^T M \right\|_2+2\gamma$.

Lastly, the optimal solution of α can be worked out by the APG framework [38], and the detailed steps are listed in Algorithm 1.

Algorithm 1. Steps of computing the optimal solution of α using APG

1. **Initialize** $a_0=a_{-1}=0 \in R^N$, $t_0=t_{-1}=1$;
2. **For** i=0,1,2,…, **iterate** until convergence

(1) $b_{i+1}=a_i+\frac{t_{i-1}-1}{t_i}(a_i-a_{i-1})$;

(2) $a_{i+1}=\arg\min_{\alpha} \frac{L}{2}\left\| \alpha - b_{i+1}+\frac{\nabla f(b_{i+1})}{L} \right\|_2^2 + G(\alpha) = \max\left\{0, b_{i+1}-\frac{\nabla f(b_{i+1})+\lambda}{L}\right\}$;

(3) $t_{i+1}=\frac{1+\sqrt{1+4t_i^2}}{2}$.

2) Estimation of Binary Support Matrix

Using the methods introduced in Section III.B.1, the sparse representation coefficient $\hat{\alpha}$ can be regarded as a fixed constant. Hence, we rewrite the optimization function in Eq.(16) as the following form:

$$\begin{aligned} &\min_{S\in\{0,1\}} \frac{1}{2}\|\Phi_S(Y-D\hat{\alpha})\|_2^2+\eta\|S\|_1+\varsigma\|\Theta S\|_1 \\ &= \min_{S\in\{0,1\}} \frac{1}{2}\sum_i (Y-D\hat{\alpha})_i^2(1-S_i)+\eta\sum_i S_i+\varsigma\|\Theta S\|_1 \\ &= \min_{S\in\{0,1\}} \sum_i (\eta-\frac{1}{2}(y-D\hat{\alpha})_i^2)S_i+\varsigma\|\Theta S\|_1+\zeta \end{aligned} \quad (23)$$

where, $\xi=\frac{1}{2}\sum_i (y-D\hat{\alpha})_i^2$ is a constant in the case that $\hat{\alpha}$ is fixed. Referring to [22], Eq.(23) is now the standard form of the first-order MRF with binary labels so that it can be well solved by maxflow/mincut method in graph cuts [39-40].

We can easily infer that S_i tends to be 1 if $\eta<\frac{1}{2}(y-D\hat{\alpha})_i^2$, indicating that the value of η highly depends on the linear span residual over D. As a matter of fact, η cannot be too small since the partial occlusion usually occupies a relatively small number of pixels in the UAV image. That is to say, a too small η will lead the occlusion to be filled with the image patch, thus resulting in a low-energy optimization function. To get a balance, we empirically update η as $\eta=\max\{\eta,4.5\sigma_r^2\}$ in each frame, where σ_r^2 represents the variance of the residual $(y-D\hat{\alpha})_i^2$.

C. Implementation of long-term tracking

In Section III.A and Section III.B, we have built up the appearance model of our tracker as well as the solution algorithm. Now, we concentrate on constructing a long-term tracking framework based on particle filter and dynamic template update.

1) Framework of Particle Filter

Particle filter is a Bayesian sequential importance sampling technique aiming for state estimation in a dynamic system, and it is widely used in non-linear and non-Gaussian tracking problems [23].

Let us define the object state in the k-th frame as a multi-dimensional state vector X_k. Assume that $Y_{1:k-1}=\{Y_1,Y_2,\ldots,Y_{k-1}\}$ is the available observations of target from the first frame to the (k-1)-th frame, $p(X_k|Y_{1:k-1})$ denoting the predicting distribution can be thus recursively computed as

$$p(X_k|Y_{1:k-1})=\int p(X_k|X_{k-1})p(X_{k-1}|Y_{1:k-1})dX_{k-1} \quad (24)$$

where, $p(X_k|X_{k-1})$ represents the state transition model.

In the k-th frame, the observation Y_k, which is the target candidate in Section III, is available and the posterior estimation of target state can be updated according to the Bayesian theorem as

$$p(X_k|Y_{1:k})=\frac{p(Y_k|X_k)p(X_k|Y_{1:k-1})}{p(Y_k|Y_{k-1})} \quad (25)$$

where, $p(Y_k|X_k)$ is the observation likelihood that measures the similarity between the candidate patch and the templates. In our study, the observation likelihood of a particle, which can be understood as the sampled candidate target mentioned in Section 3.1, is defined as the joint reconstruction error:

$$p(Y_k|X_k)=e^{-\beta(\varepsilon_p-\kappa\varepsilon_n)} \quad (26)$$

where, $\varepsilon_p=\left\| Y_k-\alpha_p D_p \right\|_2$ and $\varepsilon_n=\|Y_k-D_n\alpha_n\|_2$ represent the positive and negative reconstruction errors, respectively; β and κ are constants which are set empirically.

Suppose that P particles are generated in each frame according to the transition model $p(X_k|X_{k-1})$, and the particle state with the maximal approximate posterior probability is selected as the optimal tracking result:

$$X_k^*=\arg\max_{X_k} p(X_k|Y_{1:k}) \quad (27)$$

2) Template Update

The update of templates, especially the positive templates, is an important task in visual tracking. We state that a time-invariant template dictionary will easily result in drifts, as the object appearance only remains constant in a short period and always undergoes remarkable changes due to the influence of shape and illumination variations in UAV videos [41]. On the contrary, if we update the positive templates too often, minor tracking errors will be accumulated, which will eventually cause serious tracking failures. Therefore, the entries in the dictionary need to dynamically update with a suitable strategy. Note that we focus on the update scheme of positive templates in this work. As for the negative templates, we simply remove the templates in the previous and add the new templates from the last frame [42].

Before introducing the scheme of template update, we propose an index called average cosine similarity to judge whether the target appearance changes. Given the newly selected candidate target $\widehat{Y_k}$ in the current frame and the current positive dictionary $D_p=\left\{d_p^1,d_p^2,..,d_p^{N_p}\right\}$, the average cosine similarity $\overline{\Delta_k}$ is defined as

$$\overline{\Delta_k}=\frac{1}{N_p}\sum_{i=1}^{N_p} \arccos\langle \widehat{Y_k}, d_p^i \rangle \quad (28)$$

where, $\arccos\langle\cdot\rangle$ denotes calculating the arc-cosine value of the inner product of two vectors. If $\overline{\Delta_k}\geq\tau$ (τ is a pre-defined threshold), we judge that the target appearance has changed and we need to update the positive templates in this frame.

Another problem needs to be handled is to allocate different weights to the templates so that we can decide which template to be replaced when $\overline{\Delta_k}\geq\tau$. Directly, the sparse coefficient of positive templates α_p can be applied as the template weight. Considering the sparsity of α_p may lead to a degenerate

updating effect, an iterative approach as $\zeta_i \leftarrow \zeta_i * \exp(\alpha_{p_i})$ is designed, where the weight of each template is initialized as $\zeta_i=1/P$. One more point should be noted is that the update of template cannot be implemented when large numbers of occlusion pixels exist. That is to say, the update mechanism can be started only in the case that ones(S)≤δ, where ones(·) means counting the non-zero entries in a vector and δ=round(ξ·q) (ξ is a pre-defined threshold). For fear of the case that the newly added template plays a dominate role in the dictionary, we assign the median value of the whole weights to the new template. In addition, the maximum weight is set to be a constant ω to prevent skewing. To sum up, the complete procedure of template update is listed in Algorithm 2, and the parameter settings of τ, ξ and ω will be discussed in Section IV.E.

Algorithm 2: Steps of template update

Input: the positive template set $D_p=\{d_p^1,d_p^2,..,d_p^{N_p}\}$, the newly selected target $\widetilde{Y_k}$, the representation coefficient of positive template α_p and three pre-defined thresholds τ, ξ and ω.
Initialize the weight of each template as $\zeta_i=1/P$;
1. Update the weights: $\zeta_i=\zeta_i*\exp(\alpha_{p_i})$;
2. Compute the average cosine similarity as $\overline{\Delta_k}=\frac{1}{N_p}\sum_{i=1}^{N_p}\arccos<\widetilde{Y_k},d_p^i>$;
3. **if** $\overline{\Delta_k}\geq\tau$ and ones(S)≤δ
4. $i_0=\arg\min_{1\leq i\leq N_p}(\alpha_{p_i})$;
5. $T_p(:,i_0)\leftarrow\widetilde{Y_k}$;
6. $\zeta_{i_0}\leftarrow$median(ζ);
7. **end if**
8. Normalize ζ=ζ/sum(ζ) so that sum(ζ)=1;
9. Adjust ζ to guarantee max(ζ) =ω in order to prevent skewing;
10. Normalize the new positive dictionary D_p;
Output: the updated template D_p.

D. Summary of the Proposed Algorithm

To sum up the theories presented in Section III.A – III.C, we conclude this sparse representation-based target tracking into three main procedures: state sampling, optimization of energy function and template update. To give a clearer and more detailed explanation, a summary of our algorithm is listed in Algorithm 3.

Algorithm 3: Summary of the proposed tracker

1. **Input**: a manually cropped ground truth of target in the first frame;
2. **Initialize**: a joint dictionary including both positive and negative templates $D=(D_p,D_n)$ and the state variable X_1;
3. **for** k = 2 to the last frame **do**
4. Do the particle sampling in the affine parameter space;
5. **for** i = 1 to the last particle **do**
6. Crop and normalize the image patch of candidate target Y_i;
7. **Initialize**: $\lambda=10^{-4},\gamma=10^{-3},\hat{S}=0$;
8. **Iterate**:
9. Compute $\hat{\alpha}$ using APG: $\hat{\alpha}\leftarrow\arg\min_{\alpha}\frac{1}{2}\|\Phi_{\hat{S}}(Y-D\alpha)\|_2^2+\lambda\|\alpha\|_1+\gamma\|\alpha-\alpha_0\|_2^2$, s.t. α≥0;
10. Update η: $\eta=\max\{\eta,4.5\sigma_r^2\}$;
11. Compute $\hat{S}$ using graph cuts: $\hat{S}\leftarrow\arg\min_{S\in\{0,1\}}\sum_i\left(\eta-\frac{1}{2}(y-D\hat{\alpha})_i^2\right)S_i+\varsigma\|\Theta S\|_1$;
12. **Until convergence**
13. **end for**
14. Resample via the weight $p(X_k|Y_k)$;
15. Update the dictionary templates;
16. **Output**: $X_k^*=\arg\max_{X_k}p(X_k|X_{1:k})$.
17. **end for**

IV. Experimental Results

In this experimental section, we validate the accuracy and effectiveness of our tracker by qualitative and quantitative comparisons with other state-of-the-art tracking algorithms.

A. Datasets

To evaluate the tracking ability of the presented algorithm, 18 groups of challenging UAV videos are compiled to be the datasets in the following experiments. One half of the selected datasets are Vis videos and the other half are IR videos. On the other hand, these videos can be classified with 10 attributes according to different factors existing in UAV videos [26]: illumination variation (IV), scale variation (SV), occlusion (OCC), deformation (DEF), motion blur (MB), fast motion (FM), in-plane rotation (IPR), out-of-plane rotation (OPR), background clutters (BC) and low resolution (LR), which are summarized in Table II. All the video sequences, except the sequence pedestrian captured by our own mid-wave IR detector, can be downloaded from VIVID [43], TB-50 & TB-100 [44] and VOT2015 [45] databases.

Table II. Database categorized with 10 attributes.

		IV	SV	OCC	DEF	MB	FM	IPR	OPR	BC	LR
Vis	Egtest01	√	√				√	√		√	√
	Egtest02		√	√		√			√		
	Egtest03			√		√	√		√		
	Egtest04	√	√						√	√	
	Human			√	√				√	√	
	Racer		√	√	√	√	√		√		√
	Park			√	√						
	Suv			√				√		√	
	Truck		√	√			√	√	√		√
IR	Pktest01			√							
	Pktest02	√		√				√			
	Pktest03	√		√							√
	Pktest04	√		√				√			
	Pktest05			√							
	Horse		√		√				√		
	Rhino		√		√				√	√	√
	Pedestrian		√	√	√						
	Quadrocopter				√		√	√	√		
Total number		5	8	13	7	3	5	6	9	5	5

B. Baselines

We choose 9 state-of-the-art visual tracking algorithms, including IVT [20], L_1 [13], PCOM [22], CT [25,31], MTT [16], WMIL [29], OFDS [30], STC [21] and CNT [26], to compare with our proposed tracker. All these compared algorithms utilized are downloaded from their publicly available source codes provided by the relevant references. For fair comparisons, the tracking initializations are the same for all the algorithms, and the specific parameters of each tracker are set as their default values provided by the public codes.

C. Evaluation Criteria

In our experiments, two extensively used metrics: center location error ε_c and overlap score φ are chosen to evaluate the tracking precision rate and success rate.

ε_c is defined as the Euclidean distance between the central location of tracked object and the manually labeled ground truth [12], which can be formulated as

$$\varepsilon_c=\sqrt{(x_t-x_0)^2+(y_t-y_0)^2} \tag{29}$$

where, (x_t,y_t) and (x_0,y_0) stand for the center coordinates of the tracking result and the ground truth.

Given a tracked bounding box S_t and a corresponding ground truth bounding box S_g, φ which takes both the size and the scale of object into account is defined as

$$\varphi=\frac{|S_t\cap S_g|}{|S_t\cup S_g|}\in[0,1] \tag{30}$$

where, $|\cdot|$ means counting the pixel number in the image region decided by the bounding box; $\cap$ and $\cup$ represent the intersection and union operators.

Apparently, a satisfactory tracking result has a small value of ε_c and a large value of φ at the same time.

D. Implementation Details

In this paper, all the experiments are implemented using Matlab 2012b on a PC with 2.60 GHz INTEL CPU and 4.0 GB installed memory (RAM). All the images are converted to grayscale when executing the codes, and the initial state of the target, i.e., the center coordinate and the size, is given manually in the first frame.

Table III. Summary of parameter settings.

Parameter	Meaning	Value Setting
λ	L_1 regularization parameter	10^{-4}
γ	L_2 regularization parameter	10^{-3}
τ	Threshold of average cosine similarity	35°
ξ	Occlusion threshold	0.1
ω	Weight threshold	0.3
P	Particle number	600
r_1	Radius of positive Sampling region	4
r_2	Inner radius of negative Sampling region	8
r_3	External radius of Negative sampling region	30
N_p	Number of positive templates	10
N_n	Number of negative templates	15
β	Reconstruction error coefficient	50
κ	Reconstruction error coefficient	0.7

All the parameters mentioned in the proposed algorithm as well as their meanings and default values are listed in Table III, and the settings of some key parameters are discussed in Section IV.E.

E. Parameter Setting

As known to all, the selection of key parameters always has a great influence on the final results of target tracking. Thus, it is of great necessity for us to make a parameter analysis before running the proposed algorithm.

Here, we focus on studying the L_1 regularization parameter λ and the L_2 parameter regularization γ, the threshold of average cosine similarity τ, the weight threshold ω and the occlusion threshold ξ. In our experiments, 3168 frames of test images selected from TB50 & TB100 database are used as training data. The targets are all moving vehicles or pedestrian, which match the target types in our study. Average overlap score $\bar{\varphi}$ is employed to evaluate the tracking performances corresponding to different parameter settings.

Since λ and γ are two correlated parameters, we put them together to implement the sensitivity analysis. They are both parameterized by a discrete set $\Lambda_{\lambda,\gamma}=\{10^{-7},5\times10^{-7},10^{-6},5\times10^{-6},...,10^{0}\}$, and the performances of different (λ,η) are evaluated by $\bar{\varphi}$.

For each $\lambda_i\in\Lambda$ with different η, 15 average overlap scores can be obtained; for each $\eta_i\in\Lambda$ with different λ, 15 average overlap scores can be obtained also. That is to say, what we need to do is to find an optimal combination (λ,η) that achieves the highest $\bar{\varphi}$ among all the 15×15 groups of combinations.

To make the statistical result more intuitive, we separately analyze the two parameters as follows:

Given a fixed $\lambda_i\in\Lambda$, we record the highest $\bar{\varphi}$ among the 15 scores with different η. For different λ, we obtain the corresponding results shown in Fig. 4(a). It is clear that $\bar{\varphi}$ gets the highest value when $\lambda=10^{-4}$.

Similarly, given a fixed $\eta_i\in\Lambda$, we record the highest $\bar{\varphi}$ among the 15 scores with different λ, and the corresponding statistical result is presented in Fig. 4(b). As can be seen, $\bar{\varphi}$ reaches top when $\gamma=10^{-3}$.

Using the same way of sensitivity analysis, τ, ω and ξ are parameterized by three discrete sets: $\Lambda_\tau=\{5,10,15,\ldots,85,90\}$, $\Lambda_\omega=\{0.05,0.1,0.15,0.2,...,1\}$ and $\Lambda_\xi=\{0,0.05,0.1,0.15,\ldots,0.9,0.95,1\}$. The related $\bar{\varphi}$ values are recorded and the τ-$\bar{\varphi}$, ω-$\bar{\varphi}$ and ξ-$\bar{\varphi}$ relationship curves are drawn in Figs.(5-7). Observing the global maximum of each curve, $\bar{\varphi}$ reaches the highest point when $\tau=35^\circ$, $\omega=0.3$ and $\xi=0.1$.

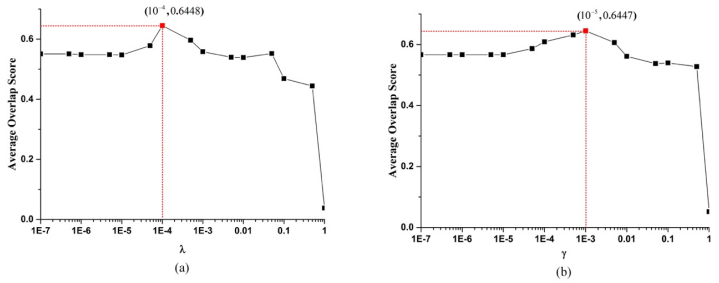

Fig.4. Effects of λ and γ on $\bar{\varphi}$: (a) λ-$\bar{\varphi}$ relationship curve; (b) γ-$\bar{\varphi}$ relationship curve.

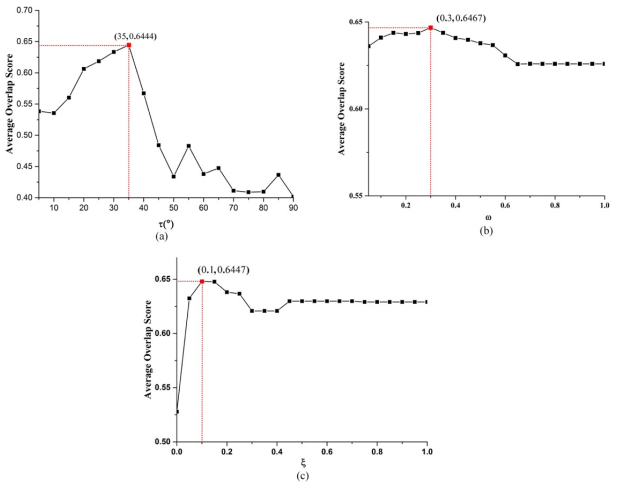

Fig.5. (a) τ-φ̄ relationship curve; (b) ω-φ̄ relationship curve; (c) ξ-φ̄ relationship curve.

F. Qualitative Comparison

To present an intuitive comparison of tracking performance, we show the tracking results of our tracker as well as other 9 state-of-the-art trackers over 9 Vis UAV videos and 9 IR UAV videos in Figs.(6-7), respectively.

In general, our tracker gets the most satisfactory performances in different UAV scenes. We find it does well in dealing with remarkable scale, deformation and rotation changes in Egtest01, Egtest02, Horse and Quadrocopte videos, which demonstrates that the dictionary-based sparse representation is effective in addressing appearance variations. Owing to the contiguous occlusion constraint, the presented tracking algorithm is also robust to various shapes of partial occlusions in Egtest03, Human, Park and Suv sequences. Notably, the dictionary used in our algorithm involves background templates, thereby making it effective in addressing the background clutters in Egtest 04 and Rhino videos. In addition, the challenges generated from fast motion (see Egtest03, Racer and Truck videos) is overcome by adjusting the translation parameters of particle filter framework.

As for other algorithms, CNT and STC also achieve relatively promising results in most sequences, but we notice that the size of STC's bounding box is not so stable (see Suv and Pktest01 videos). IVT and L_1 are generative trackers which focus on setting up appropriate object appearance models, so these two trackers are quite good in coping with videos with appearance changes, like Egtest01 and Human. CT ,WMIL and ODFS regard the tracking problem as a binary classification, but we find that they are easy to drift when remarkable changes of environmental factors and object appearance occur. What is more, the tracking result of PCOM degrades when encountering illumination changes and occlusion interferences. As a whole, MTT gets the worst tracking performance since it loses the target in most of the videos.

G. Quantitative Comparison

In this part, relevant experimental results of one-pass evaluation (OPE) [48] on the basis of precision rate and success rate are reported. Based on the two evaluation criteria discussed in Section IV.C, we further introduce two plots to test the quantitative performances of the 10 trackers:

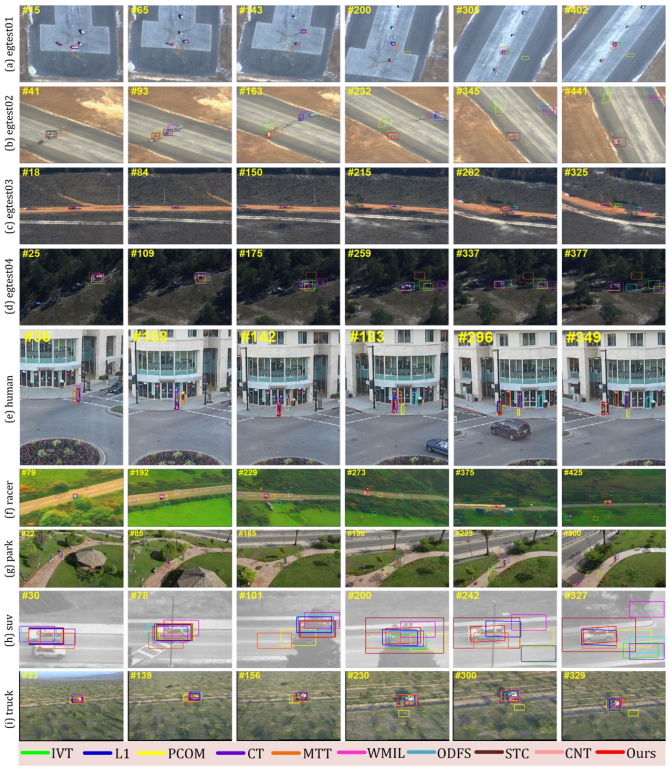

Fig. 6. Tracking results of 10 trackers on 9 Vis UAV videos delineated by different colors. Frame numbers are overlayed in yellow.

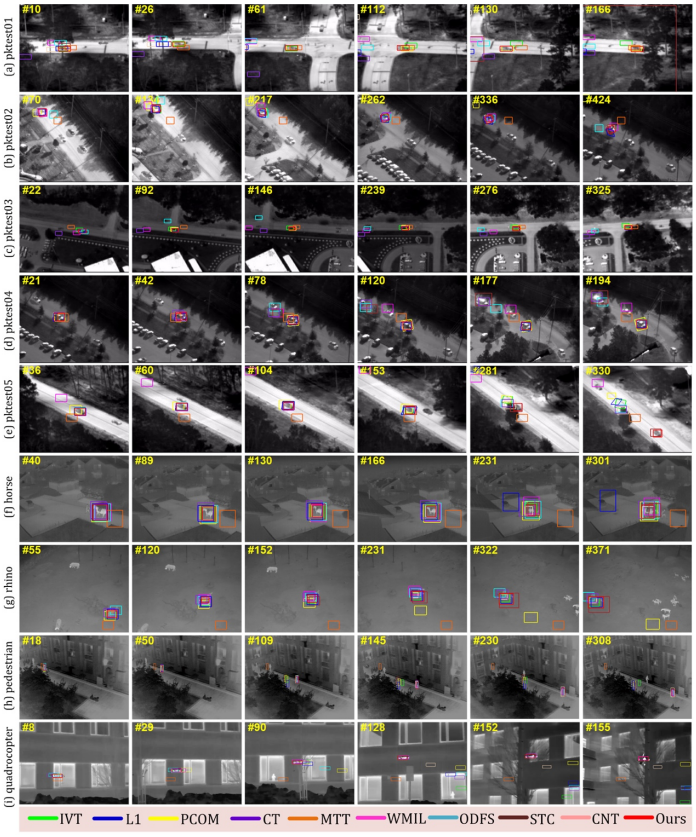

Fig. 7. Tracking results of 10 trackers on 9 IR UAV videos delineated by different colors. Frame numbers are overlayed in yellow.

Table IV. The average location center error and overlap score of the overall performance. The red fonts indicate the best performances; the green fonts indicate the second-best ones and the blue fonts indicate the third-best ones.

	IVT	L_1	PCOM	CT	MTT	WMIL	ODFS	STC	CNT	Ours
$\overline{\varepsilon_c}$	58.807	50.666	66.274	93.638	74.497	79.571	68.537	41.432	22.099	4.813
$\overline{\varphi}$	0.408	0.521	0.264	0.333	0.085	0.265	0.322	0.458	0.609	0.709

① Precision Plot is defined as the percentage (precision rate) of those frames whose center location errors are shorter than a given threshold;
② Success Plot is defined as the percentage (success rate) of those frames whose overlap scores are higher than a given threshold. Usually, we choose the success rate with the threshold set as 0.5 for tracking evaluation.

1) Overall Performance

Fig.8 reports the overall performance of the 10 trackers in terms of precision and success plots over 18 UAV videos. In addition, Table IV lists the average location error $\overline{\varepsilon_c}$ and the average overlap score $\overline{\varphi}$, which are calculated as the average ε_c value and φ value of all the frames, respectively.

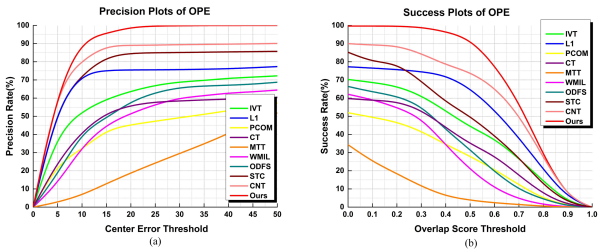

Fig.8. Overall performance over 18 UAV videos: (a) precision plot of OPE; (b) success plot of OPE.

Intuitively, the proposed tracker ranks first in both precision rate and success rate while MTT ranks last. In the precision plot, our designed tracking algorithm achieves the $\overline{\varepsilon_c}$ of 4.813 pixels while the second-best tracker CNT gets the $\overline{\varepsilon_c}$ of 22.099 pixels, which is about 4.5 times greater than our method. CT gets a poor performance in precision rate, which is approximately 18 times behind that of the proposed tracker. As a whole, ODFS, IVT, L_1, PCOM and STC are at the same level (40-60 pixels) whereas the average location center errors of MTT and WMIL are larger than 70 pixels, but smaller than 90 pixels.
In the success plot, our method and CNT method still occupy the top two positions, the average overlap scores of which are 0.709 and 0.609 respectively. However, we notice that MTT falls down to the last and its $\overline{\varphi}$ value is only 12% of our method. The performances of PCOM and WMIL, whose $\overline{\varphi}$ values are between 0.2 and 0.3, are also not satisfactory. The rest trackers, especially ODFS, IVT and L_1, achieve the modest $\overline{\varphi}$ values that match their performances in precision rate.
To sum up, the proposed method and CNT method get the most accurate and robust tracking results while the tracking abilities of CT and MTT are the weakest among all.

2) Attribute-based Performance

To further analyze the effectiveness and robustness in different environments, the precision and success plots are drawn in Fig.9 and Fig.10 according to the 10 attributes introduced in Table II. Besides, statistical results of $\overline{\varepsilon_c}$ and $\overline{\varphi}$ of each attribute are revealed in Table V and Table VI.

Illumination variation: We test 5 videos (Egtest01, Egtest04, Pktest02, Pktest03, Pktest04) characterizing in having obvious illumination change. As a whole, L_1 tracker and our method rank top two in this attribute. L_1 achieves the best precision rate while our tracker achieves the top performance in success rate. Noticeably, CNT only 8% falls behind our tracker in $\overline{\varphi}$, which proves its effectiveness in illumination variation to some degree.

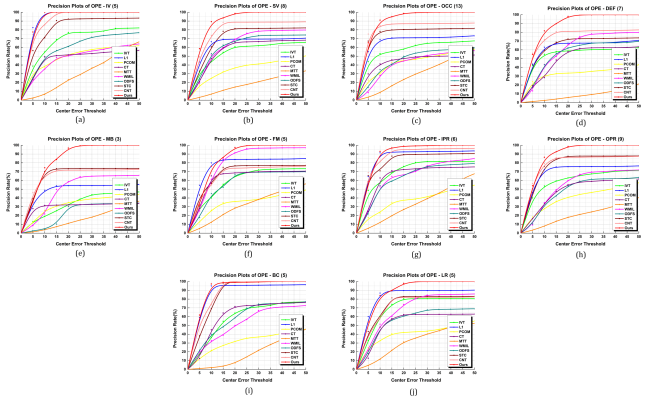

Fig.9. The precision plots of videos with different attributes. The number in the title stands for the number of sequences belonging to the attribute.

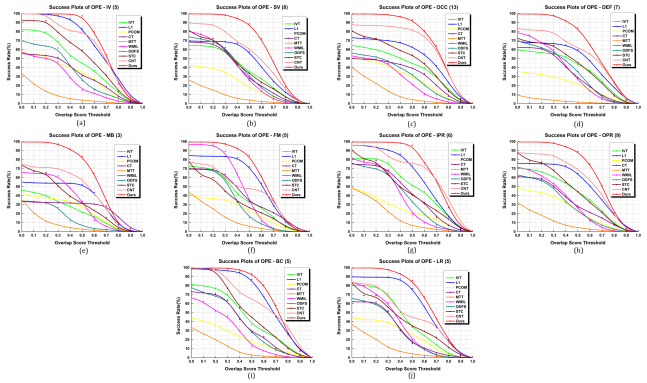

Fig.10. The success plots of videos with different attributes. The number in the title stands for the number of sequences belonging to the attribute.

Scale Variation: Our method and CNT tracker outperform other state-of-the-art methods in both $\overline{\varepsilon_c}$ and $\overline{\varphi}$. It is striking that STC method degrades steeply when addressing sequences with scale variation, which further validates the observation mentioned in Section IV.F that the update of scale factor is easily affected by the surrounding environment. In our opinion, the particle filter framework that takes scale factor into the affine parameter helps overcome the effect of scale change. To this end, discriminative tracking algorithms, like WMIL, CT and ODFS, are comparatively weaker in this aspect.
Occlusion: Similar to SV, the proposed tracker and CNT tracker also occupies the first two rankings in OCC. Despite the inaccuracy in target scale, STC still gets the third place in $\overline{\varepsilon_c}$. This is probably because $\overline{\varepsilon_c}$ measures the shift between the bounding box center and the ground truth center meanwhile the

Table V. Statistical results of $\overline{\varepsilon_c}$. The red fonts indicate the best performance; the green fonts indicate the second-best ones and the blue fonts indicate the third-best ones.

	IVT	L_1	PCOM	CT	MTT	WMIL	ODFS	STC	CNT	Ours
IV	32.545	3.019	55.096	60.136	51.051	58.109	32.762	13.211	4.008	3.321
SV	81.703	76.744	79.653	90.904	101.347	56.989	79.262	60.761	29.404	5.498
OCC	67.075	61.668	63.671	113.793	70.663	94.373	85.687	49.871	26.141	4.799
DEF	87.808	67.254	86.166	130.492	120.493	72.611	82.572	85.551	40.186	6.258
MB	144.346	143.088	87.332	189.596	91.379	94.332	191.893	109.504	65.393	7.981
FM	88.519	56.424	78.175	92.367	71.073	13.556	93.7345	92.909	55.131	7.844
IPR	34.972	13.935	65.912	42.806	45.644	27.803	40.375	22.158	8.967	5.001
OPR	74.321	66.827	69.962	99.938	78.883	69.399	97.998	51.789	30.695	5.771
BC	37.354	6.153	74.267	47.722	71.554	68.807	41.885	6.704	5.326	4.673
LR	67.592	41.943	62.604	88.263	74.639	24.137	81.111	72.578	44.542	5.879
Ave.	71.623	53.705	72.283	95.601	77.672	58.011	82.728	56.503	30.979	5.701

Table VI. Statistical results of $\overline{\varphi}$. The red fonts indicate the best performance; the green fonts indicate the second-best ones and the blue fonts indicate the third-best ones.

	IVT	L_1	PCOM	CT	MTT	WMIL	ODFS	STC	CNT	Ours
IV	0.442	0.671	0.304	0.314	0.074	0.242	0.328	0.531	0.645	0.711
SV	0.352	0.447	0.203	0.334	0.071	0.339	0.346	0.369	0.532	0.668
OCC	0.394	0.508	0.269	0.315	0.101	0.242	0.386	0.289	0.646	0.730
DEF	0.368	0.463	0.192	0.311	0.026	0.332	0.333	0.402	0.539	0.669
MB	0.216	0.372	0.227	0.256	0.064	0.326	0.140	0.328	0.446	0.634
FM	0.357	0.540	0.213	0.385	0.116	0.444	0.297	0.302	0.438	0.641
IPR	0.483	0.598	0.223	0.406	0.148	0.340	0.359	0.426	0.652	0.691
OPR	0.397	0.511	0.234	0.312	0.085	0.305	0.280	0.427	0.568	0.694
BC	0.424	0.674	0.201	0.351	0.087	0.249	0.348	0.494	0.622	0.720
LR	0.397	0.591	0.236	0.272	0.093	0.325	0.273	0.378	0.466	0.680
Ave.	0.383	0.537	0.230	0.325	0.086	0.314	0.309	0.411	0.555	0.683

tracked object of STC does not drift far away from the real target too much. It can be concluded from Fig.9(c) and Fig.10(c) that the target appearance model constructed by generative methods may be more robust to occlusion than that constructed by discriminative method, since the former (e.g., CNT, IVT and L_1) rank ahead of the latter (e.g., WMIL, CT and ODFS) in both precision rate and success rate. As far as we are concerned, generative approaches use more complicated mathematical models that take occlusion inferences into consideration to represent the target appearance whereas the running efficiency is sacrificed. We particularly point out that IVT projects the target pixels onto a PCA orthogonal space and weighs the similarity using Mahalanobis distance, therefore this method is not so sensitive to partial occlusion.

Deformation & Motion Blur: The proposed joint dictionary-based sparse coding method forms a global representation, thereby equipping it to account for DEF. CNT tracker encodes the geometric layout information with multiple simple cell feature maps [26], which also contributes to its robustness to DEF. Thus, the two trackers get the best performances in DEF attribute. On the other hand, our method provides a much more promising performance in MB, owing to its selection of template dictionary involving background templates. By contrast, L_1 and MTT which also use sparse coding, but ignore the influence of background degrade strikingly, even less than half of ours in terms of $\overline{\varphi}$.

Fast Motion & Rotation: The candidate sampling strategy does influence the tracking accuracy in FM attribute. Since the variance of translation parameters of our particle filter are set to be large, our method still achieves satisfactory ranks in FM. Noteworthily, WMIL also achieves high values of $\overline{\varepsilon_c}$ (only falls behind ours by 5.7 pixels on average) and $\overline{\varphi}$ (17% smaller than L_1 tracker) owing to its greedy strategy for updating the tracker location. As can be seen from the plots shown in Fig.9(f-g) and Fig.10(f-g), our method, CNT, L_1 and STC receive the highest four rankings, mainly because of their complex appearance representation models and the effective sampling strategies.

Background clutter: The hybrid template dictionary designed in our method provides useful information to locate the position of target from the background clutters. For CNT, the background context information is updated online and pooled in each frame [26], which is quite beneficial to overcome the adverse influence of clutters. Also, L_1 tracker takes the trivial templates into consideration, so the drift generated by background clutter can be alleviated to a certain degree.

Table VII. Statistical results of $\overline{\varepsilon_c}$ for each model.

	IV	SV	OCC	DEF	MB	FM	IPR	OPR	BC	LR	Ave.
Model A	3.321	5.498	4.799	6.258	7.981	7.844	5.001	5.771	4.673	5.879	5.701
Model B	45.887	75.765	70.968	87.405	108.737	103.91	70.401	78.430	65.158	83.076	78.973
Model C	36.155	57.749	69.423	79.901	103.879	81.193	64.346	63.493	55.914	68.867	68.092
Model D	8.227	13.127	8.431	13.431	19.949	20.126	12.493	14.041	11.082	14.009	13.491

Table VII. Statistical results of $\bar{\varphi}$ for each model.

	IV	SV	OCC	DEF	MB	FM	IPR	OPR	BC	LR	Ave.
Model A	0.711	0.668	0.730	0.669	0.634	0.641	0.691	0.694	0.720	0.680	0.683
Model B	0.330	0.306	0.342	0.294	0.292	0.301	0.322	0.322	0.337	0.317	0.317
Model C	0.476	0.441	0.361	0.345	0.341	0.411	0.429	0.429	0.486	0.478	0.424
Model D	0.659	0.595	0.629	0.580	0.543	0.571	0.593	0.593	0.641	0.597	0.601

PCOM performs poorly in scenes with background clutters, because the background pixels are easily mistaken as the principle component while the pattern on the real target is categorized to be outliers.

Low resolution: Naturally, the low resolution attribute in UAV videos may make it hard to exploit effective hand-crafted features from the target, thereby decreasing the tracking accuracy greatly. As for the precision rate, the $\bar{\varepsilon}_c$ value of our method outperforms the second place WMIL and the third place L_1 by 75.1% and 85.7%, respectively. In regard to the success rate, the superiority of WMIL is replaced by CNT, but WMIL still gets a relatively promising result (32.5% on average).

H. Ablation Experiment

As described in Section III.A.4, the proposed tracker can be regarded as the following optimization problem, which is called as Model A in this section.

Model A: Complete model

$$\min_{\alpha,S\in\{0,1\}} \frac{1}{2}\|\Phi_S(Y-D\alpha)\|_2^2+\lambda\|\alpha\|_1+\gamma\|\alpha-\alpha_0\|_2^2+\eta\|S\|_1+\varsigma\|\Theta S\|_1 \quad \text{s.t. } \alpha\geq 0 \tag{31}$$

Based on the function of each part in Eq.(31), Model A can be further divided into three sub-parts:

Model B: Basic sparse representation mode

$$\min_{\alpha} \frac{1}{2}\|Y-D\alpha\|_2^2+\lambda\|\alpha\|_1, \text{ s.t. } \alpha\geq 0 \tag{32}$$

Model C: Temporal consistency constraint-based sparse representation model

$$\min_{\alpha} \frac{1}{2}\|Y-D\alpha\|_2^2+\lambda\|\alpha\|_1+\gamma\|\alpha-\alpha_0\|_2^2, \text{ s.t. } \alpha\geq 0 \tag{33}$$

Model D: Continuous occlusion constraint-based sparse representation model

$$\min_{\alpha,S\in\{0,1\}} \frac{1}{2}\|\Phi_S(Y-D\alpha)\|_2^2+\lambda\|\alpha\|_1+\eta\|S\|_1+\varsigma\|\Theta S\|_1, \quad \text{s.t. } \alpha\geq 0 \tag{34}$$

To verify the impact of each part on the proposed tracker, the above four models are utilized to implement the tracker on the 18 test videos, respectively. To be fair, Model B and Model C are solved by APG method and Model D is solved by The alternating algorithm involving APG method and graph cuts. Besides, all the relevant parameters are set as the default values listed in Table II.

In Table VII and Table VIII, we make a statistic in terms of average center location error $\bar{\varepsilon}_c$ and average overlap score $\bar{\varphi}$ to show the attribute-based performance for each model.

As for Model A, it is quite clear that it is far superior than other models in all types of attributes, which demonstrates the superiority of combination of sparse representation, temporal consistency and contiguous occlusion constraint. By contrast , Model B is weak at dealing with all the attributes because the constraint in this basic model is the simplest. However, the tracking precision improves to some degree when the temporal consistency constraint is added to Model B. For Model C, its $\bar{\varepsilon}_c$ decreases about 13.7% and its $\bar{\varphi}$ increases about 33.7% on average when compared with Model B, but the improvement in OCC, DEF and MB is still not so remarkable. This is because the test videos belonging to these three attributes all suffer from partial occlusions, but Model C lacks occlusion related constraint. On the other hand, Model D with sparsity constraint and contiguous constraint does achieves much significant improvement in tracking accuracy. As can be seen, its average $\bar{\varepsilon}_c$ is approximately 2.36 times larger than Model A, but is only 17.0% and 19.8% of Model B and Model C. The condition of $\bar{\varphi}$ is also satisfactory, which is almost the same with Model A, but 89.5% and 41.7% greater than Model B and Model C. This fact demonstrates the effectiveness of integrating sparsity and contiguous occlusion constraints together, especially in treating OCC.

To sum up, we can get the following conclusions from this ablation experiment:

① The basic sparse representation model is far insufficient to get accurate tracking results in all types of UAV scenes;

② The temporal consistency model is able to improve the tracking precision to a certain degree when the target appearance changes, but it is almost useless in addressing UAV videos with partial occlusions;

③ The contiguous occlusion constraint model is effective in overcoming the influence caused by OCC. When it is combined with the basic sparse representation model, it can achieve satisfactory tracking precision in most cases;

④ It is the combination of the above three models that contributes to the success of the complete tracking model to get satisfactory tracking performance in UAV videos.

I. Execution Efficiency

We aim to analyze the execution efficiency of the proposed algorithm by both theoretical analysis and comparative test. Firstly, the time complexity of our tracker is given; then, the running speed of each tracker is tested in a quantitative way.

Table IX Average execution time and frame frequency of each tracker over the 18 UAV videos utilized in Section IV.F. The red font indicates the best performance; the green font indicates the second best one and the blue font indicates the third best one.

	IVT	L1	PCOM	CT	MTT	WMIL	ODFS	STC	CNT	Ours
T/s	0.161	0.164	0.012	0.101	0.159	0.100	0.052	0.066	3.030	1.538
Fr/fps	6.21	6.09	83.33	9.90	6.28	10.00	19.23	15.15	0.33	0.65

1) Time Complexity

The most time-consuming part of our algorithm is the iterations implemented in Algorithm 1. For α step, the iteration of APG stops when $\|\alpha_i-\alpha_*\|<\epsilon$, where α_* is one minimizer of Eq.(16) and ϵ is a convergence threshold. According to Bao et al.'s analysis [15], this α step achieves ϵ- optimality within $K=O\left(\sqrt{L/\epsilon}\right)$ iterations. For S step, the binary optimization problem is solved by graph cut based maxflow/mincut strategy using Boykov-Kolmogorov (BK) algorithm [46]. Based on the previous study made Boykov et al. [47], the trivial upper bound on the number of augmentations for BK algorithm is the cost of the minimum cut $|C|$ ($C=\{S,T\}$, where S and T are two disjoint subsets of graph), which results in the worst case complexity $O\left(N_n N_e^2|C|\right)$ (N_n and N_e are the numbers of nodes and edges in the graph, respectively). Thus, the time complexity of Algorithm 1 can be computed as $O\left(\max\left(K,N_n N_e^2|C|\right)\right)$.

With respect to Algorithm 2, it is obvious that its time complexity is O(1) because there are no iterations in it.

Algorithm 3 is the whole procedures of our tracking algorithm with the total frame number and the total particle number set as N and P, respectively. This means the total iteration number of Algorithm 1 and Algorithm 2 is NP. As a result, the time complexity of the proposed algorithm can be denoted as $O\left(NP\cdot\left(\max\left(K,N_n N_e^2|C|\right)\right)\right)$.

2) Execution Time

To fairly test the execution efficiency of each tracker, we record the average execution time T for running a frame on average and compute the corresponding average frame frequency Fr in Table IX.

As we can see, discriminative approaches, e.g., PCOM, ODFS, WMIL and CT, all achieve satisfactory running speeds. It is worth noting that PCOM gets the most promising average running time (0.012s) and frame frequency (83.33 fps) among all the 10 algorithms. On the other hand, generative approaches, especially those particle filter-based trackers, are comparatively much weaker in this aspect. Generally, the average frame frequency of IVT, L_1 and MTT is approximately 6fps, which is only 7% of PCOM. We argue that this is because the iterative computations of particle filter that hugely increase the running time. STC gets the second place in this comparison, mainly owing to its fast convolution operation implemented in Fourier domain. Through the statistical results presented in Table IX, we must admit that our method does suffer from the poor running efficiency which only 0.65 fps on average. According to our analysis, not only the iterations in particle filter, but also the iterations in APG algorithm, that improve the computational cost. CNT also depends on the convolution operation, but it is slowest one among all, the average running time of which is about 2 times longer than ours. This is because the convolution in spatial domain is time-consuming and this high computational load is also amplified manyfold by particle filter. Although the execution efficiency of our method is not so superior, we still believe that it can be alleviated and improved by the following two aspects that is of expectation in the future: ① introduce the mechanism of multi-task sparse learning [16] to compute the solutions of particles all at once; ② exploit other more efficient solution algorithms to replace APG approach; 3) transplant the Matlab code to GPU platform which can address each particle in parallel.

We argue that the real-time running of the proposed method is promising, since the frame frequency may be increased by 600 times, if the procedure of particle filter can be implemented in parallel by GPU.

V. Discussion

To further demonstrate the reasonability of the details designed in our proposed algorithm, we make a discussion about the construction of joint dictionary and the penalties.

A. Discussion about Joint Dictionary

In this part, we focus on proving two detailed conclusion mentioned in Section III: 1) Using joint dictionary representation can improve the sparsity of coefficient; 2) Negative templates will not affect the coding of positive image patch. In our discussion, the 18 UAV videos in Section IV as well as their ground truth image patches are employed as test data.

1) Advantage of Joint Dictionary

The ground truth Y is coded by a joint dictionary $D_1=[D_p,D_n]$ containing both positive and negative templates and the single dictionary $D_2=D_p$ containing only positive templates, respectively, which can be expressed as

$$\min_{\alpha_1}\frac{1}{2}\|Y-D_1\alpha_1\|_2^2 \tag{35}$$

$$\min_{\alpha_2}\frac{1}{2}\|Y-D_2\alpha_2\|_2^2 \tag{36}$$

where, D_p and D_n are sampled around Y in each frame, and the two coefficients α_1 and α_2 can be computed using APG method. Since L_1 norm is an extensively used metric to evaluate the sparsity of vector, we define the change ratio of sparsity γ_s as

$$\gamma_s=\frac{\|\alpha_2\|_1-\|\alpha_1\|_1}{\|\alpha_1\|_1}\times 100\% \tag{37}$$

Obviously, $\gamma_s>0$ if α_1 is sparser than α_2. Here, the γ_s value is calculated for each frame and we record the average value of γ_s of each video, which is written as $\overline{\gamma_s}$. Table X reports the statistical result of $\overline{\gamma_s}$.

It is clear that the $\overline{\gamma_s}$ values of all the videos are positive, which directly demonstrates that the representation coefficient using joint dictionary is sparser than the one using single dictionary. Moreover, the average value of $\overline{\gamma_s}$ is approximately 31.6%, meaning that the sparsity is improved 31.6% on average by means of this strategy.

2) Effect of Negative Templates

When the positive templates are fixed and the negative ones are changed, we aim to demonstrate the following two facts: ① the coding coefficient corresponding to positive templates will not have obvious difference; ② the coding coefficient corresponding to negative templates always remains $\mathbf{0}$.

As introduced in Section III, the positive and negative templates are randomly sampled in region Ω_p with a sampling radius r_1 and Ω_n with an inner radius r_2 and an external radius r_3. For changing the negative dictionary, we fix r_1 to a get a group of

Table X. Statistical result of $\overline{\gamma_s}$ (%).

	Egtest01	Egtest02	Egtest03	Egtest04	Human	Racer	Park	Suv	Truck
$\overline{\gamma_s}$	24.19	13.87	5.73	49.48	9.89	31.24	22.13	18.91	15.68
	Pktest01	**Pktest02**	**Pktest03**	**Pktest04**	**Pktest05**	**Horse**	**Rhino**	**Pedestrian**	**Quadrocopter**
$\overline{\gamma_s}$	25.86	27.96	28.88	41.51	62.19	26.58	71.84	64.98	28.05
Ave.								31.60	

Table XI. Statistical result of $\overline{\Delta_p}$(°).

	Egtest01	Egtest02	Egtest03	Egtest04	Human	Racer	Park	Suv	Truck
$\overline{\Delta_p}$	1.4943	1.2121	2.9484	1.3436	2.0517	1.3029	1.6128	1.0560	2.5627
	Pktest01	**Pktest02**	**Pktest03**	**Pktest04**	**Pktest05**	**Horse**	**Rhino**	**Pedestrian**	**Quadrocopter**
$\overline{\Delta_p}$	2.4062	2.1683	2.0843	1.7628	1.8568	1.5953	2.3025	2.4062	2.1683
Ave.								1.9075	

Table XII. Statistical results of $\overline{\|\alpha_{n1}\|_1}$ and $\overline{\|\alpha_{n2}\|_1}$.

	Egtest01	Egtest02	Egtest03	Egtest04	Human	Racer	Park	Suv	Truck
$\overline{\|\alpha_{n1}\|_1}$	0.0835	0.0442	0.0483	0.0445	0.0342	0.0872	0.0853	0.0604	0.0501
$\overline{\|\alpha_{n2}\|_1}$	0.0821	0.0458	0.0511	0.0461	0.0343	0.0861	0.0855	0.0609	0.0489
	Pktest01	**Pktest02**	**Pktest03**	**Pktest04**	**Pktest05**	**Horse**	**Rhino**	**Pedestrian**	**Quadrocopter**
$\overline{\|\alpha_{n1}\|_1}$	0.0457	0.0185	0.0735	0.0039	0.0470	0.0873	0.0094	0.0305	0.0897
$\overline{\|\alpha_{n2}\|_1}$	0.0433	0.0192	0.0729	0.0031	0.0471	0.0867	0.0102	0.0308	0.0914
Ave.				$\overline{\|\alpha_{n1}\|_1}$.				0.0524	
				$\overline{\|\alpha_{n2}\|_1}$.				0.0525	

positive templates and change r_2 and r_3 to get two groups of different negative templates for each frame. By this means, a positive dictionary D_p with r_1=4 and two negative dictionaries with different sampling radii: D_{n_1} (r_2=8, r_3=30) and D_{n_2} (r_2=30, r_3=52) are the generated.

Then, the ground truth of target Y is coded by $D_1=[D_p,D_{n_1}]$ and $D_2=[D_p,D_{n_2}]$ using Eqs.(35-36), and two corresponding coding coefficients $\alpha_1=\begin{pmatrix}\alpha_{p_1}\\ \alpha_{n_1}\end{pmatrix}$ and $\alpha_2=\begin{pmatrix}\alpha_{p_2}\\ \alpha_{n_2}\end{pmatrix}$ are obtained.

Here, two statistical experiments are implemented below:

① the cosine similarity Δ_p between α_{p_1} and α_{p_2} is computed to judge whether the coding coefficient corresponding to positive templates have obvious difference. For each frame, Δ_p is computed as

$$\Delta_p=\arccos\langle\alpha_{p_1},\alpha_{p_2}\rangle \tag{38}$$

Here, the average value of Δ_p of each video, denoted as $\overline{\Delta_p}$, is counted and the statistical result is reported in Table XI. As can be seen clearly, the $\overline{\Delta_p}$ values of the 18 videos are all smaller than 3°, and the average value of $\overline{\Delta_p}$ is only 1.9075°. We argue that this cosine similarity is quite small when compared with the threshold $\tau=35^\circ$ in Algorithm 1 which represents remarkable appearance change. Based on this consideration, it is convincing that the coding coefficient corresponding to positive templates almost keep unchanged when D_n varies.

② Based on the discussion above, the two coding coefficients corresponding to negative templates are denoted as α_{n1} and α_{n2}. It should be noted that α_{n1} and α_{n2} cannot be a perfect zero vector due to the inevitable coding error. In other words, α_{n1} and α_{n2} should be very sparse vectors if the conclusion is true. Therefore, L_1 norm is employed to evaluate the sparsity again. To be specific, $\|\alpha_{n1}\|_1$ and $\|\alpha_{n2}\|_1$ are calculated for each frame and the average values $\overline{\|\alpha_{n1}\|_1}$ and $\overline{\|\alpha_{n2}\|_1}$ are calculated for each test video. Table XII shows the related statistical results. From Table XII, we can get two points: firstly, the L_1 norms of coding coefficients corresponding to negative templates are quite small (10^{-3}~10^{-2}), indicating that they tend to be zero-vectors; secondly, for a real target, the change does not affect the coding coefficients corresponding to negative templates, since $\overline{\|\alpha_{n1}\|_1}$ and $\overline{\|\alpha_{n2}\|_1}$ are almost the same.

B. Discussion about Penalty

According to the experimental results presented in Section IV.G, our tracker achieves outstanding improvement in tracking precision when compared with other state-of-the-art ones. However, we do admit that there are some penalties in other aspects:

1) Time cost: As can be seen from Table IV, our tracker and CNT achieve the top two performances in terms of overall $\overline{\varepsilon_c}$ and $\overline{\varphi}$. By contrast, their frame frequencies are the lowest. This phenomenon indicates that high tracking precision always leads to high time cost. As for our algorithm, the most time-consuming part is the $K=O(\sqrt{L/\epsilon})$ iterations in Algorithm 1. Moreover, this procedure is repeated P times due to the particle filter scheme.

2) Hardware cost: Our final aim is to realize this tracking algorithm on UAV system, so we need to transplant it to hardware platform. Typically, FPGA+GPU or FPGA+DSP architecture is applied in UAV surveillance system, in which

FPGA is good at implementing convolution operations while GPU/DSP is utilized for float-point calculations. Actually, most of the procedures of our algorithm can only be implemented in GPU or DSP, and only template sampling can be done in FPGA. To this end, the resource of FPGA is wasted, thus a more advanced DSP or GPU is needed to get real-time running. That is to say, the core frequency of DSP should be higher and there should be more parallel cores in DSP or GPU. It results in two further penalties: ① the price of system will improve; ② the power consumption will be higher.

VI. Conclusion

This paper proposes a sparse representation-based target tracking algorithm aiming at locating the target of interest, e.g., pedestrian or vehicle, in UAV videos. First of all, the target candidate is linearly represented by the sub-space spanned by a joint dictionary and the coefficient is constrained by an L_1 regularization. Considering the temporal consistency, an L_2 regularization is imposed on the representation coefficient further. To overcome the adverse effect of partial occlusion extensively existing in UAV videos, a contiguous occlusion constraint with MRF-based binary support vector is developed to model the occlusion. Through our sparse representation model, the tracking problem is cast as an optimization problem which can be solved by an alternating algorithm involving APG and graph cuts. Finally, we implement the tracking algorithm under the framework of particle filter and select the particle with the maximal approximate posterior probability as the tracking result. Experimental results show that the proposed algorithm performs well in terms of accuracy and robustness. Our future work will focus on optimizing the codes and transplanting our algorithm to hardware devices, like FPGA and GPU, to make the real-time running come true.

References

[1] Hu, L., & Ni, Q. (2018). IoT-driven automated object detection algorithm for urban surveillance systems in Smart Cities. IEEE Internet of Things Journal, 5(2), 747-754.
[2] Yao, Y., Sun, Y., Phillips, C., & Cao, Y. (2018). Movement-Aware Relay Selection for Delay-Tolerant Information Dissemination in Wildlife Tracking and Monitoring Applications. IEEE Internet of Things Journal.
[3] Lu, H., Li, Y., Mu, S., Wang, D., Kim, H., & Serikawa, S. (2017). Motor anomaly detection for unmanned aerial vehicles using reinforcement learning. IEEE Internet of Things Journal.
[4] Cao, Y., Wang, G., Yan, D., & Zhao, Z. (2015). Two algorithms for the detection and tracking of moving vehicle targets in aerial infrared image sequences. Remote Sensing, 8(1), 28.
[5] Yuan, X., Kong, L., Feng, D., & Wei, Z. (2017). Automatic feature point detection and tracking of human actions in time-of-flight videos. IEEE/CAA Journal of Automatica Sinica, 4(4), 677-685.
[6] Zhang, S. (2005, March). Object tracking in unmanned aerial vehicle (uav) videos using a combined approach. In Acoustics, Speech, and Signal Processing, 2005. Proceedings. (ICASSP'05). IEEE International Conference on (Vol. 2, pp. ii-681). IEEE.
[7] Liu, X., Lu, L., Shen, Z., & Lu, K. (2018). A novel face recognition algorithm via weighted kernel sparse representation. Future Generation Computer Systems, 80, 653-663.
[8] Geng, L., Ji, Z., Yuan, Y., & Yin, Y. (2018). Fractional-order sparse representation for image denoising. IEEE/CAA Journal of Automatica Sinica, 5(2), 555-563.
[9] Yang, J., Wright, J., Huang, T. S., & Ma, Y. (2010). Image super-resolution via sparse representation. IEEE transactions on image processing, 19(11), 2861-2873.
[10] Lai, T., Wang, H., Yan, Y., Chin, T. J., & Zhao, W. L. (2017). Motion segmentation via a sparsity constraint. IEEE Transactions on Intelligent Transportation Systems, 18(4), 973-983.
[11] Yuan, Z., Lu, T., & Tan, C. L. (2017). Learning discriminated and correlated patches for multi-view object detection using sparse coding. Pattern Recognition, 69, 26-38.
[12] Zhang, T., Liu, S., Ahuja, N., Yang, M. H., & Ghanem, B. (2015). Robust visual tracking via consistent low-rank sparse learning. International Journal of Computer Vision, 111(2), 171-190.
[13] Mei, X., & Ling, H. (2009, September). Robust visual tracking using ℓ 1 minimization. In Computer Vision, 2009 IEEE 12th International Conference on (pp. 1436-1443). IEEE.
[14] Mei, X., Ling, H., Wu, Y., Blasch, E., & Bai, L. (2011, June). Minimum error bounded efficient ℓ 1 tracker with occlusion detection. In Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on (pp. 1257-1264). IEEE.
[15] Bao, C., Wu, Y., Ling, H., & Ji, H. (2012, June). Real time robust l1 tracker using accelerated proximal gradient approach. In Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on (pp. 1830-1837). IEEE.
[16] Zhang, T., Ghanem, B., Liu, S., & Ahuja, N. (2012, June). Robust visual tracking via multi-task sparse learning. In Computer vision and pattern recognition (CVPR), 2012 IEEE conference on (pp. 2042-2049). IEEE.
[17] Salti, S., Cavallaro, A., & Di Stefano, L. (2012). Adaptive appearance modeling for video tracking: Survey and evaluation. IEEE Transactions on Image Processing, 21(10), 4334-4348.
[18] Li X, Hu W, Shen C, et al. A survey of appearance models in visual object tracking[J]. ACM transactions on Intelligent Systems and Technology (TIST), 2013, 4(4): 58.
[19] Ho, J., Lee, K. C., Yang, M. H., & Kriegman, D. (2004, June). Visual tracking using learned linear subspaces. In Computer Vision and Pattern Recognition (CVPR), 2004 IEEE Conference on (pp. 782-789). IEEE.
[20] Ross, D. A., Lim, J., Lin, R. S., & Yang, M. H. (2008). Incremental learning for robust visual tracking. International journal of computer vision, 77(1-3), 125-141.
[21] Zhang, K., Zhang, L., Liu, Q., Zhang, D., & Yang, M. H. (2014, September). Fast visual tracking via dense spatio-temporal context learning. In European Conference on Computer Vision (pp. 127-141). Springer, Cham.
[22] Wang, D., & Lu, H. (2014). Visual tracking via probability continuous outlier model. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 3478-3485).
[23] Kong, X., Chen, Q., Xu, F., Gu, G., Ren, K., & Qian, W. (2015). Motion object tracking based on the low-rank matrix representation. Optical Review, 22(5), 786-801.
[24] Oron, S., Bar-Hillel, A., Levi, D., & Avidan, S. (2015). Locally orderless tracking. International Journal of Computer Vision, 111(2), 213-228.
[25] Zhang, K., Zhang, L., & Yang, M. H. (2014). Fast compressive tracking. IEEE Transactions on Pattern Analysis & Machine Intelligence, (1), 1-1.
[26] Zhang, K., Liu, Q., Wu, Y., & Yang, M. H. (2016). Robust visual tracking via convolutional networks without training. IEEE Transactions on Image Processing, 25(4), 1779-1792.
[27] Grabner, H., & Bischof, H. (2006, June). On-line boosting and vision. In Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on (Vol. 1, pp. 260-267). IEEE.
[28] Babenko, B., Yang, M. H., & Belongie, S. (2011). Robust object tracking with online multiple instance learning. IEEE transactions on pattern analysis and machine intelligence, 33(8), 1619-1632.
[29] Zhang, K., & Song, H. (2013). Real-time visual tracking via online weighted multiple instance learning. Pattern Recognition, 46(1), 397-411.
[30] Zhang, K., Zhang, L., & Yang, M. H. (2013). Real-time object tracking via online discriminative feature selection. IEEE Transactions on Image Processing, 22(12), 4664-4677.
[31] Zhang, K., Zhang, L., & Yang, M. H. (2012, October). Real-time compressive tracking. In European conference on computer vision (pp. 864-877). Springer, Berlin, Heidelberg.
[32] Hare, S., Golodetz, S., Saffari, A., Vineet, V., Cheng, M. M., Hicks, S. L., & Torr, P. H. (2015). Struck: Structured output tracking with kernels. IEEE transactions on pattern analysis and machine intelligence, 38(10), 2096-2109.
[33] Henriques, J. F., Caseiro, R., Martins, P., & Batista, J. (2012, October). Exploiting the circulant structure of tracking-by-detection with kernels. In European conference on computer vision (pp. 702-715). Springer, Berlin, Heidelberg.
[34] Li, Z., Li, J., Ge, F., Shao, W., Liu, B., & Jin, G. (2016). Dim moving target tracking algorithm based on particle discriminative sparse representation. Infrared Physics & Technology, 75, 100-106.

[35] Li, S. Z. (2009). Markov random field modeling in image analysis. Springer Science & Business Media.
[36] Zhou, X., Yang, C., & Yu, W. (2013). Moving object detection by detecting contiguous outliers in the low-rank representation. IEEE Transactions on Pattern Analysis and Machine Intelligence, 35(3), 597-610.
[37] Wang, P., Chen, Q., & Shao, N. (2016, September). Moving object detection via low-rank total variation regularization. In Applications of Digital Image Processing XXXIX (Vol. 9971, p. 997132). International Society for Optics and Photonics.
[38] Wang, B. X., Zhao, B. J., Tang, L. B., Wang, S. G., & Wu, J. H. (2014). Robust visual tracking algorithm based on bidirectional sparse representation.
[39] Kolmogorov, V., & Zabih, R. (2002, May). What energy functions can be minimized via graph cuts?. In European conference on computer vision (pp. 65-81). Springer, Berlin, Heidelberg.
[40] Boykov, Y., Veksler, O., & Zabih, R. (2001). Fast approximate energy minimization via graph cuts. IEEE Transactions on pattern analysis and machine intelligence, 23(11), 1222-1239.
[41] Li, Y., Li, P., & Shen, Q. (2014). Real-time infrared target tracking based on ℓ 1 minimization and compressive features. Applied optics, 53(28), 6518-6526.
[42] Du, D., Lu, H., Zhang, L., & Li, F. (2015, September). Visual tracking via guided filter. In Image Processing (ICIP), 2015 IEEE International Conference on (pp. 1781-1785). IEEE.
[43] http://vision.cse.psu.edu/data/vividEval/datasets/datasets.html
[44] http://cvlab.hanyang.ac.kr/tracker_benchmark/datasets.html
[45] http://www.votchallenge.net/vot2015/dataset.html
[46] https://vision.cs.uwaterloo.ca/code/
[47] Boykov Y, Kolmogorov V. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision[J]. IEEE Transactions on Pattern Analysis & Machine Intelligence, 2004 (9): 1124-1137.
[48] Wu, Y., Lim, J., & Yang, M. H. (2013). Online object tracking: A benchmark. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 2411-2418).