J. Vis. Commun. Image R. 39 (2016) 253–265

Contents lists available at ScienceDirect

J. Vis. Commun. Image R.

journal homepage: www.elsevier.com/locate/jvci

Tracking with dynamic weighted compressive model

Ting Chen [a,b,*], Yanning Zhang [a], Tao Yang [a], Hichem Sahli [a,b,c]

[a] School of Computer Science and Engineering, Northwestern Polytechnical University, Xi'an 710072, PR China
[b] Dept. Electronics and Informatics (ETRO), AVSP Lab, Vrije Universiteit Brussel (VUB), Brussels, Belgium
[c] Interuniversity Microelectronics Center (IMEC), Leuven, Belgium

ARTICLE INFO

Article history:
Received 11 August 2015
Revised 12 April 2016
Accepted 3 June 2016
Available online 4 June 2016

Keywords:
Compressive tracking
Dynamic weighted compressive model
Random matrix

ABSTRACT

Fast compressive tracking utilizes a very sparse measurement matrix to capture the appearance model of targets. Such model performs well when the tracked targets are well defined. However, when the targets are low-grain, low-resolution, or small, a single fixed size sparse measurement matrix is not sufficient enough to preserve the image structure of the target. In this work, we propose a multi-sparse measurement matrices scheme along with a weight map to select the best measurement matrix that preserves the image structure of the targets during tracking. The weight map combines a contrast weight and a feature weight to efficiently characterize the target appearance and location. Moreover, a dispersion function is used for the online update of the target template, allowing tracking both the location and scale of the target. Extensive experimental results have demonstrated that the proposed DWCM tracking algorithm outperforms several state-of-the-art tracking algorithms as well as compressive tracker.

1. Introduction

Visual tracking is an important topic in computer vision, and have a wide range of practical applications, such as automatic video monitoring, and video indexing [1,2]. Visual tracking is challenging due to significant target appearance variations caused by illumination change, occlusion, fast or abrupt target motion, and also cluttered background [3–14].

To design robust trackers, much attention has been made in devising effective appearance models, being the key factor for the performances of a tracking algorithm [15,16]. Based on the used appearance model, tracking algorithms can be formulated as *discriminative* tracking or *generative* tracking. Discriminative tracking, treats visual tracking as a binary classification problem to define the boundary between a target and the background. Among such approaches, Avidan [17], integrated the SVM classifier and the optical flow tracker in a Support Vector Tracking (SVT) mechanism. In [18], Collins et al., developed an on-line feature ranking mechanism which selects the top-ranked distribution features to separate the target from the background. Babenko et al. [19], utilized a multiple instance learning (MIL) framework to update the appearance model with a set of image patches. Dinh et al. [20], proposed a tracking method which automatically explores the context information with the semantic terms of distracters and supporters. In [21], Grabner et al. utilized an on-line semi-supervised boosting method which combined decision of a given prior and an on-line classifier to solve the drift problem in tracking applications. In [22], Avidan used the boosting method to train a strong classifier and then the mean-shift method [23] to find the location of the target. Kalal et al. [24], proposed an on-line P–N learning approach with structural constraints to train a binary classifier from labeled and unlabeled examples in an iterative process. The learning process is guided by positive (P) and negative (N) constraints which restrict the labeling of the unlabeled set. P–N learning has been applied to solve the problem of on-line learning of target detector during tracking. Finally, we can mention the work of Yao et al. [25], who proposed a framework for weighted online learning, using weighted reservoir sampling for tracking.

Generative tracking methods learn a model to represent the appearance of a target. Tracking is then expressed as finding the most similar target appearance to the model. In this category of methods we can refer to the work of Adam et al. [26], who utilized multiple fragments or patches to characterize the template target. This approach is robust to partial occlusions. In [27], Ross et al. proposed an on-line incremental subspace learning algorithm to reflect the appearance changes during tacking. In [28], Kwon et al. used a visual tracking decomposition scheme to address the tracking of an target whose motion and appearance change drastically.

Sparse representation has recently been introduced for tracking, in which a tracking candidate is sparsely represented as a linear

* Corresponding author.
E-mail address: chentingnwpu@163.com (T. Chen).

http://dx.doi.org/10.1016/j.jvcir.2016.06.001
1047-3203/

combination of target templates and trivial templates. In [29], Mei et al. proposed a robust visual tracking method by casting tracking as a sparse approximation problem in a particle filter framework. The sparsity has been achieved by solving an l_1 regularized least squares problem. The limitation of Mei's et al. tracker is that its computational complexity is rather high. To alleviate this limitation, Li et al. [30], introduced a customized Orthogonal Matching Pursuit (OMP).

Compressive sampling is a new concept in signal processing and information theory where one measures a small number of non-adaptive linear combinations of the signal. These measurements are usually much smaller than the number of samples that define the signal. From these small number of measurements, the signal is then reconstructed by a non-linear procedure [31,32]. Based on the compressive sensing ideas, Zhang et al. [33] proposed a compressive tracking (CT) method. They employed a fixed sparse random measurement matrix to extract low-dimensional features from a multi-scale image feature space, being a linear combination of generalized Haar-like features. Then, the features in the compressed domain, are classified via a naive Bayes classifier with online update. This method achieved good results in several scenarios. Fast compressive tracking (FCT) [34] is an extension to CT that adopts a coarse-to-fine search strategy to reduce the computational complexity in the detection procedure.

Both CT and FCT use a fixed random measurement matrix, which could cause the target to be lost when its appearance or scale changes, thus suffering from the drifting problem. To alleviate this problem, Teng and Liu [35] proposed a multi-scale tracking method via random projections (MSRP), where rapid fern-based features, instead of the Haar-like features of CT and FCT, have been used. Moreover, they employed a feedback mechanism to reduce tracking drift and perform multi-scale tracking via a point tracking process to correct the final location of the target. In [36], they employed two random measurement matrices to extract two complementary features, along with an adaptive weighting scheme favoring the best performing features to update the classifier. In [37], the authors proposed a semi-supervised compressive coding algorithm (SCC) for online sample labeling. An adaptive compressive sensing for appearance modeling using a weighted random projection and the Fisher discrimination criterion are used to evaluate the discrimination capability of each random feature. With the purpose of multi-scale tracking and adaptive classification, Wu et al. [38] proposed a multi-scale tracking based on compressive sensing (MSCT) which combines the random projection-based appearance model with the bootstrap filter framework.

Motivated by the above discussion on Compressive Tracking, in this work an effective and efficient CT-based tracking algorithm is proposed. The algorithm is based on the original CT tracker of Zhang et al. [33], for the use of sparse random measurement matrix, the Haar-like multi-scale image features and the naive Bayes classifier with online update. However, to reduce tracking drift, we follow the ideas of (i) the adaptive random measurement matrix of [38,36], and (ii) the feedback mechanism of [35,38]. We propose a Dynamic Weighted Compressive Model (DWCM) for tracking. The DWCM algorithm adopts multi-sparse measurement matrices along with an adaptive weighting scheme favoring the best performing sparse measurement matrix. To deal with the scale, we apply several sparse measurement matrices with different dimensions. We apply a weighting scheme, combining contrast weight (to emphasize the appearance) and a feature weight (to emphasize the texture) to preserve the target's appearance. To further reduce tracking drift, we employ a feedback mechanism. A dispersion function, resembling the motion models used in [36,38], is defined for the online update of the target location and template. The remainder of this paper is organized as follows, Section 2 details the proposed Dynamic Weighted Compressive tracking (DWTM) method. In Section 3, we carry out an extensive quantitative evaluation which shows a significant improvement over state-of-the-art approaches. Finally, Section 4 presents concluding remarks.

2. Proposed DWCM algorithm

This paper is an extension of our previous work in [39], where we proposed a dynamic compressive tracking algorithm, augmenting the real-time compressive tracking algorithm of Zhang et al. [33]. In [39] we used several sparse measurement matrices with different dimensions and a dynamic importance ranking weight. The approach allowed selecting the best random matrix (i.e. dimension) to well locate the target. In this paper, we follow the same idea of using multi-sparse measurement matrices with different dimensions, however, different from [39], the proposed DWCM algorithm (i) uses a new formulation of the weight map (see Section 2.4), and (ii) introduce a feedback mechanism, through a dispersion function, for the online update of the target location and template (see Section 2.5).

2.1. DWCM overview

The main components of the proposed algorithm are shown in Fig. 1. At the first frame, we locate manually the target that we intend to track in the video (or by using an automated detector). Then, for each frame I_t we follow the two stages algorithm of the original compressive tracker of Zhang et al. [33]: *tracking* and *updating*. In the *tracking* stage, candidate image patches or samples of the target at frame I_t are sampled around the tracking result at I_{t-1} (Fig. 1A). Then, integral vectors of these patches are calculated by accumulation (see Section 2.2). Based on the compressive sensing [33], the high-dimensional integral vectors of samples are compressed to extract low dimensional feature vectors using a static measurement matrix $\boldsymbol{R}$. This process is denoted as:

$$\mathbf{v} = R\mathbf{x} \tag{1}$$

with $R \in \mathbb{R}^{n\times m}$ being the random projection matrix, $\mathbf{x} \in \mathbb{R}^m$ denotes the integral vectors, and $\mathbf{v} \in \mathbb{R}^n$ indicates the compressed feature vectors with dimensions $n \ll m$. $R \in \mathbb{R}^{n\times m}$ is a very sparse random matrix, the entries of which are defined as:

$$r_{ij} = \sqrt{s} \times \begin{cases} 1 & with \quad probability \quad \frac{1}{2s} \\ 0 & with \quad probability \quad 1-\frac{1}{s} \\ -1 & with \quad probability \quad \frac{1}{2s} \end{cases} \tag{2}$$

with $s = 1$ or 3. In [33], the parameter n has been set as $n = o(m) = m/(a\log_{10}(m)) = m/(10a) \sim m/(6a)$, with the constant $a = 0.4$, which makes $\boldsymbol{R}$ a very sparse matrix. $\boldsymbol{R}$ is also easy to compute, as it requires only a uniform random generator. In our case, we use multiple random matrices, $R_r \in \mathbb{R}^{n_r\times m}$; $r = 1,\ldots,N_R$ ($N_R \geqslant 2$), with different dimensions ($n_r \times m$) and random values. After the process of compression, the low-dimensional feature vectors, $\mathbf{v}_r \in \mathbb{R}^{n_r}$, $r = 1,\ldots,N_R$, are entered into N_R online learning naive Bayes classifiers, the output of which is a pool of candidates tracking locations (see Section 2.3). The optimal target location (Fig. 1B) is then estimated as the location optimizing an importance weight function (see Section 2.4).

In the *updating* stage (Fig. 1D), first a feedback mechanism (Fig. 1C) is applied to force all the candidates tracking locations to the optimal location (see Section 2.5). The optimal location at frame I_t is used for selecting training samples, of the target and background, to update the parameters of the naive Bayes classifiers, which will be used in the $(t+1)$-th frame.