J. Vis. Commun. Image R. 55 (2018) 688–697

Contents lists available at ScienceDirect

J. Vis. Commun. Image R.

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

Visual tracking via Graph Regularized Kernel Correlation Filer and Multi-Memory Voting ☆

Weiwei Zheng, Huimin Yu *, Wei Huang

College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou, People's Republic of China

ARTICLE INFO

Article history:
Received 6 September 2017
Revised 2 June 2018
Accepted 3 August 2018
Available online 4 August 2018

Keywords:
Visual object tracking
Graph Regularized Kernel Correlation Filer
Multi-Memory Voting
Drift handling

ABSTRACT

Correlation filter based tracking approach has been an important branch of visual tracking. However, most correlation filter based trackers fail to work under occlusion due to their frame-by-frame model update strategy, and the tracking performance can be further enhanced by optimizing the energy equation. The target appearance during tracking is nearly moving on a manifold. So, the classification scores should be similar on the target manifold. K Nearest Neighbor graphs are constructed and the classification scores on the neighborhood are regularized to have similar values. Through the local score propagation on the graph, the learned Graph Regularized Kernel Correlation Filer can represent different appearances of the object. Furthermore, in the proposed Multi-Memory Voting scheme, occlusion problem is addressed by voting from multiple target snapshots in the memory pool. An extensive evaluation on two recent benchmarks shows that the proposed tracker achieves competitive performance compared to nine other state-of-the-art trackers.

1. Introduction

Visual object tracking has been a popular research direction in computer vision for years. It can be used in public applications such as video surveillance, traffic monitoring, and anomaly detection. While it remains an open problem as there are many challenges such as occlusion, deformation, illumination variation, and out of view [1].

The goal of visual tracking is to predict the states of a target in subsequent video frames, given the initial bounding box of the target. Since the first use of the tracking-by-detection scheme in SVT [2], it has been widely applied in different tracking approaches. The tracking-by-detection scheme converts a tracking problem into a detection problem, and a classifier is learnt online to find the target location with maximum classification score. A tracking-by-detection based tracker has three main parts, including target model, motion model and model update. There are various target models proposed over the pase decade. SVM model [3–6] is often used to discriminate target objects from background. Usually, few samples are employed for training due to the limitation of model complexity, which weakens the discrimination of SVM based trackers. In boosting models [7–9], several weak classifiers are updated online and linearly combined to form a strong classifier. The performance of the ensemble classifier is highly dependent on the design of weak classifiers. Some local linear and nonlinear subspaces are learned to represent various appearances of the target in the subspace model [10]. In sparse representation models [11–18], each candidate patch is sparsely reconstructed by the pre-learned dictionaries and low reconstruction error indicates high target probability. This kind of trackers is time-consuming due to the $L1$ regularization. With the success in the field of object classification, deep learning methods have been transferred into tracking problem. MDNet [19] is one of the best deep learning based trackers. The network structure is composed of shared pre-trained CNN layers and multiple branches of domain-specific layers. Each branch is responsible for one target and updated during the tracking process. Despite of excellent performance, MDNet is time-consuming with nearly 1 fps with the help of GPU. While in SiamFC [20], a novel fully-convolutional Siamese network is proposed for target searching. This method achieves good performance and operates beyond real-time. Meanwhile, some efforts are made by CFNet [21] and ECO [18] to combine the Correlation filter with deep learning techniques.

Correlation filter is widely used in the field of signal processing. In image processing, cross-correlation is a measure of similarity between two images as a function of the displacement. So, correlation filters can be used for object recognition and location. In addition to the tracking field, correlation filters have been used for face

☆ This paper has been recommended for acceptance by Zicheng Liu.

* Corresponding author.
E-mail address: yhm2005@zju.edu.cn (H. Yu).

https://doi.org/10.1016/j.jvcir.2018.08.004
1047-3203/

recognition in [22–24] and road sign identification in [25]. More applications of correlation filter in image processing can be found in [26].

Correlation filter based methods [27–32,17,18] have received much attention as they can solve two common problems in the tracking-by-detection scheme. One is that most trackers have to apply dense search within the predetermined region around the current target. The other is that few samples are used to update the tracking model due to real-time requirements. As we know, the lack of training samples makes a classifier less discriminant in a machine learning problem. In correlation filter based methods, thousands of samples are obtained by cyclic shifts of a base sample, and the classification scores for all these samples can be obtained by the convolution between the base sample and the filter coefficients. Ridge Regression is often employed as their target model. However, these methods take all training samples as independent ones, and do not reveal the relevance among these samples. In the tracking problem, all samples have some intrinsic relevance in space and time. If the relevance among samples could be revealed to constraint the learning model, the tracking performance will be improved. Moreover, most existing correlation filter based trackers update their model parameters frame by frame with a constant learning rate. As time goes on, the target model gradually forgets its previous appearance. This simple model update strategy is likely to result in the drift problem under the condition of occlusion.

In order to address the above-mentioned issues, we propose a Graph Regularized Kernel Correlation Filer (GRKCF) to reveal the relevance among training samples, and a Multi-Memory Voting (MMV) scheme to handle the drift problem under occlusion. In the graph model, similar pairs of training samples are linked by edges on the K Nearest Neighbor (KNN) graph. The relevant information between two training samples are propagated through neighbor edges rather than the direct connection between them. It is because the feature space of the target is generally locally Euclidean, based on the manifold assumption. Most existing trackers fail to work when their target models get corrupted by inaccurate prediction or occlusion. To address this common problem, a memory pool is maintained to store multiple historical snapshots of the target. The memory pool can adaptively update and eliminate the corrupted snapshots. Meanwhile, the proposed voting scheme helps to get the tracker out of the trap on the background occlusion. Fig. 1 shows the main procedure of the proposed tracking method.

2. Proposed algorithm

In this section, we present our tracking approach. In Section 2.1, a Graph Regularized Kernel Correlation Filer is proposed for the tracking problem. Sections 2.2 and 2.3 present the optimization algorithm to solve the learning problem. In Section 2.4, a Multi-Memory Voting scheme is designed to handle the drift problem under occlusion. Finally in Section 2.5, we introduce the scale estimation strategy.

2.1. Graph Regularized Kernel Correlation Filer

As we know, most existing correlation filter based trackers employed supervised learning method. Semi-supervised learning methods make use of both labeled and unlabeled data to learn a classifier. So, one advantage of semi-supervised learning is that it reveals the relevance information among the labeled samples and the unlabeled samples to enhance the learning model. The graph model is used to build information propagating paths among the labeled and unlabeled samples. On the graph, the classification scores are propagated among neighbor samples. It is called Local Score Propagation, which ensures local consistency (or smoothness) on the neighborhood.

Here, the tracking problem is taken as a binary classification problem. The labeled samples are image patches extracted from historical frames and the unlabeled samples are image patches extracted from the current frame. Every sample obtains a classification score, ranging from 0 to 1. The probability of a sample belonging to the target is proportional to its classification score. We construct two kNN graphs for information propagation, and the two graph regularizers are added to the Ridge Regression equation. The two graph introduces similarity relevance of the feature space and spatial location relevance into the energy equation (2).

When learning our Graph Regularized Kernel Correlation Filer, the first thing to do is to construct a kNN graph from the training samples which consist of labeled samples $\{(\boldsymbol{x_i}, \boldsymbol{y_i})\}_{i=1}^{l}$ and unlabeled samples $\{\boldsymbol{x_j}\}_{j=l+1}^{l+u}$. Here $\boldsymbol{x_i}$ is the feature of the ith sample, and y_i is the associated label. Every vertex in the graph corresponds to one training sample, and is connected to k nearest neighbor vertices. The edge weight of two vertices represents the similarity between them. For the first graph, the weight function used here is the Gaussian kernel

$$\omega_{ij}^{g1} = exp\left(-\frac{\|\boldsymbol{x_i} - \boldsymbol{x_j}\|^2}{\sigma_\omega^2}\right), \tag{1}$$

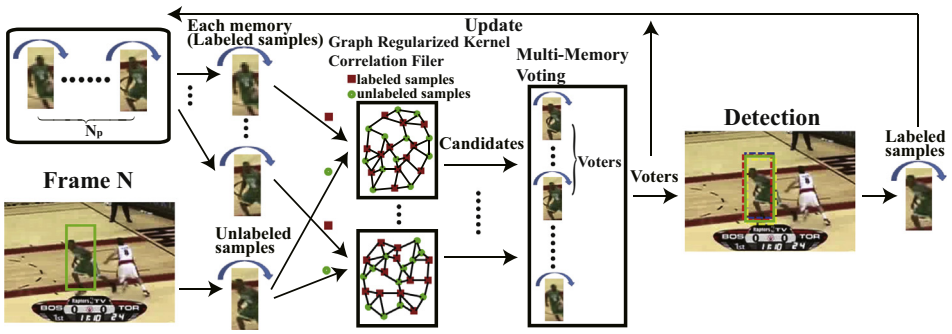

Fig. 1. Main procedure of the proposed algorithm. Suppose we want to track the target in the subsequent Frame *N*. First, we extract unlabeled samples around the current location of the target and map the samples into the feature space. Then, kNN graphs are constructed from the unlabeled samples and every memory in the memory pool. A classifier is obtained by learning the proposed Graph Regularized Kernel Correlation Filer. All unlabeled samples get the corresponding classification scores and the one scored highest will be selected as a candidate target. Finally, the top 3 candidates ranked by the classification scores will vote for the final location and scale of the target object. After these steps, the memory pool is updated by the new labeled samples.