Image and Vision Computing 32 (2014) 465–475

Contents lists available at ScienceDirect

Image and Vision Computing

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

Editor's Choice Article

# Robust visual tracking via augmented kernel SVM ☆

Yancheng Bai, Ming Tang *

*National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, China*

ARTICLE INFO

*Article history:*
Received 5 March 2013
Received in revised form 2 March 2014
Accepted 12 April 2014
Available online 22 April 2014

*Keywords:*
Feature representation
Appearance model
Augmented Kernel Matrix (AKM)

ABSTRACT

Most current tracking approaches utilize only one type of feature to represent the target and learn the appearance model of the target just by using the current frame or a few recent ones. The limited representation of one single type of feature might not represent the target well. What's more, the appearance model learning from the current frame or a few recent ones is intolerant of abrupt appearance changes in short time intervals. These two factors might cause the track's failure. To overcome these two limitations, in this paper, we apply the Augmented Kernel Matrix (AKM) classification to combine two complementary features, pixel intensity and LBP (Local Binary Pattern) features, to enrich the target's representation. Meanwhile, we employ the AKM clustering to group the tracking results into a few aspects. And then, the representative patches are selected and added into the training set to learn the appearance model. This makes the appearance model cover more aspects of the target appearance and more robust to abrupt appearance changes. Experiments compared with several state-of-the-art methods on challenging sequences demonstrate the effectiveness and robustness of the proposed algorithm.



## 1. Introduction

Object tracking is an important problem in computer vision and has many practical applications, especially for vehicle navigation, augmented reality (AR) and human computer interface (HCI). Although many tracking algorithms have been proposed [1–8], it is still a great challenging task to design a robust tracker which can cope with different kinds of objects under various situations. A very common difficulty is to deal with the visual appearance changes of the target over time due to pose changes, sudden illumination changes, partial occlusion and background clutter.

To deal with the changes of the object appearance, a common strategy is to adopt an adaptive appearance model [2,3,5], which updates during the tracking process with the appearance changes of the target. To construct the adaptive model, it often uses the current or the latest few frames and represents the target by one type of the feature. Comaniciu [1] only used the current frame and represented the target as the weighted feature histogram to construct the appearance model. Ross et al. [3] employed the incremental learning scheme to learn a low-dimensional subspace representation, which was based on the pixel intensity. Babenko et al. [5] represented the target by using the generalized Haar-like features [9] and applied Multiple Instance Learning (MIL) [10] to select many discriminative weak classifiers from a feature pool. The weak classifiers were updated online by means of a forgetting factor and this implicitly meant that the appearance model was built only on the latest few frames. Kwon and Lee [4] utilized the first initial and latest four frames and sparse principal component analysis (SPCA) to select a set of complementary feature templates to model the target. Bai and Tang [7] proposed the online Laplacian ranking support vector tracker (LRSVT) to incorporate the initial and the latest frames and the weakly labeled information in the next frame to model the appearance. In the LRSVT system, the target was represented by using Haar-like features.

Many of the aforementioned approaches could not deal with all various appearance changes simultaneously either due to the limited representation of one single type of feature or the only usage of the recent target samples [11]. To cope with the representation limitation, one strategy was that we could design a strong feature which was robust to any change. However, it was not an easy task [12,13], especially for the model-free tracking problem in which no prior knowledge about the target is known except for the object location at the beginning of tracking. Another strategy was that we could propose an efficient scheme that combined different complementary features (e.g. image features based on pixel intensity, edge and texture information) [12,13]. One type of feature captured one channel of information of the target and compensated for others' representation limitation.

To make the tracker be intolerant of abrupt appearance changes in short time intervals, we could add the tracking result into the training set [11]. Nevertheless, if we brutally added all the tracking results, it would need much time to learn the appearance model and even degrade the tracker's performance. Instead, we could mine a few representative examples from the tracking results to add into the training set.

In this paper, we applied the Augmented Kernel Matrix (AKM) classification [14–16] to combine two complementary features, the pixel

☆ Editor's Choice Articles are invited and handled by a select rotating 12 member Editorial Board committee. This paper has been recommended for acceptance by Vladimir Pavlovic.

* Corresponding author at: Room 1404, Intelligent Building, 95 Zhongguan-cun East Road, 100190, Beijing, China.
*E-mail addresses:* ycbai@nlpr.ia.ac.cn (Y. Bai), tangm@nlpr.ia.ac.cn (M. Tang).

http://dx.doi.org/10.1016/j.imavis.2014.04.008
0262-8856/

intensity and LBP [17] features to model the target's appearance. The AKM classification assigns different weights to the information channel per example rather than per kernel. As a result, this could make the AKM learn a much more representative appearance model to cover the target appearance changes. Meanwhile, to make usage of the tracking result, we proposed the AKM clustering algorithm to group the tracking result into a few aspects and selected a few examples from each aspect. The AKM clustering has two advantages as follows. Firstly, it could decide the number of the cluster automatically, which was very important to the tracking problem. We could not give a predefined number of clusters due to the arbitrary appearance changes during the tracking process. Secondly, it could also handle outliers by employing a soft margin constant. Therefore, it might discard a few inaccurate tracking results which were unavoidable in the tracking process. And this contributed to our system to be more robust to abrupt appearance changes in short time intervals.

The rest of this paper is organized as follows. In Section 2, we briefly overview the related work. The novel tracking algorithm, the Augmented Kernel Matrix tracker and the implementation details are given in Section 3. Experimental results and comparison with other state-of-the-art methods are presented in Section 4. Section 5 summarizes our work.

## 2. Related work

To combine different features, Bach et al. [18] formulate the multiple kernel learning (MKL) as a second-order cone programming (SCOP) problem, which can be solved efficiently only for medium-scale problems. Therefore, Rakotomamonjy et al. [19] propose solving by subgradient descent approach, which converges rapidly and is favorably efficient compared to other MKL algorithms. Vishwanathan et al. [20] develop a very efficient approach, called SMO–MKL, which uses sequential minimal optimization (SMO) [21] algorithm to solve MKL problem directly and is easily scaled to the large problem.

In computer vision, MKL has been applied to object detection [12,22] and classification [13]. Varma et al. [12] apply MKL to learn a combination of base kernels and show impressive results on varied object classification tasks. To make the system more efficient, Vedaldi et al. [22] extend Varma's work and learn a three-stage classifier which combines linear and non-linear kernels. Gehler and Nowozin [13] compare different feature combination strategies for object classification.

In [14], Yan et al. propose that the AKM learning for a single training example may have different importance in different feature spaces, in contrast to MKL that assigns the same weight to all examples in one feature space. Due to the large augmented matrix which requires a large memory, Awais et al. [15] develop a novel two-stage Augmented Kernel Matrix. Kernels are grouped automatically by kernel alignment and the most influential training examples are identified within each group and used to construct the AKM matrix. The AKM learning has been applied to object recognition [14,15] and shows improvement over MKL and other feature combination schemes.

In the tracking problem, Kim et al. [23] cluster the tracking results into a few aspects and learn cluster-specific appearance models for object tracking. Different from Kim's work, the purpose of the clustering in ours is to mine the most representative tracking results and Kim's work requires an off-line setup to construct the cluster priors.

In [11], Park et al. propose an online clustering algorithm to cluster the historical information into a few clusters and learn a few cluster-specific appearance models as Kim did. Due to the online scheme, the tracker might learn a cluster-specific appearance modeling the background, which causes tracking failures.

## 3. The Augmented Kernel Matrix Tracker

In this section, we begin with an overview of multiple kernel learning, which is a common scheme for feature combination and has been widely used in object detection [12,22] and classification [13]. Next, we present the AKM learning [14] for classification and clustering. Finally, we give our tracking algorithm and the implementation details.

### 3.1. The multiple kernel learning

MKL is to learn a linear combination of different kernels during the training phase. And the feature combination in MKL can be given in the following formulation:

$$\begin{aligned} &K\left(x_i,x_j\right)=\sum_{m=1}^{M}d_mK_m\left(f_m(x_i),f_m\left(x_j\right)\right)\\ &\sum_{m=1}^{M}d_m=1,d_m\geq 0,\forall m=1\cdots M; \end{aligned} \tag{1}$$

where $f_m(x_i)$ is the $m$th feature of image patch $x_i$, the weight $d_m$ stands for the importance of the $m$th kernel $K_m(f_m(x_i),f_m(x_j))$ and $M$ is the total number of kernels. In our work, one feature only corresponds to one kernel. Therefore, the kernel and feature have the same meaning if there's no other explanation.

Among the many existing MKL learning algorithms, simpleMKL [19] is known for its simplicity and efficiency. The optimization problem of simpleMKL is given as follows:

$$\begin{aligned} \min_{d,w,b,\eta}\quad & \frac{1}{2}\sum_{m=1}^{M}\frac{1}{d_m}\|w_m\|^2+C\sum_{i=1}^{N}\eta_i\\ \text{s.t.}\quad & y_i\left(\sum_{m=1}^{M}w_m^T\Phi(f_m(x_i))+b\right)\geq 1-\eta_i,\forall i=1\cdots N;\\ & \sum_{m=1}^{M}d_m=1,d_m\geq 0,\forall m=1\cdots M; \end{aligned} \tag{2}$$

where $y_i$ is the label of the image patch $x_i$, $\Phi(f_m(x_i))$ is the implicit mapping of $f_m(x_i)$ imposed by the $m$th kernel, $C$ is the penalty parameter, $\eta_i$ is the slack factor, $b$ is the bias and $N$ is the total number of training examples. Problem (2) can be solved efficiently by alternating between determining the SVM model parameters via a standard SVM solver and determining the kernel combination importance by using a projected gradient descent method. Details can be found in [19].

The main problem in MKL is that all samples in one feature space are assigned the same weight, however, the individual samples may have different importance. Another problem is that we find that simpleMKL often assigns a major weight to one feature due to the sparsity at kernel level. This would not achieve the goal of feature combination to resist the appearance changes. Therefore, we use the AKM learning to construct the appearance model.

### 3.2. Feature combination via AKM classification

The key idea of the Augmented Kernel Matrix Learning is that the positive and negative training examples in each feature space are classified as positive or negative respectively. This allows to assign different weights to the information channel per example rather than per kernel which exploit information from individual samples deeply. Therefore, the primal SVM of AKM scheme is given:

$$\begin{aligned} \min_{w,b,\eta}\quad & \frac{1}{2}\sum_{m=1}^{M}\|w_m\|^2+C\sum_{m=1}^{M}\sum_{i=1}^{N}\eta_{mi}\\ \text{s.t.}\quad & y_i\left(w_m^T\Phi(f_m(x_i))+b\right)\geq 1-\eta_{mi},\\ & \eta_{mi}\geq 0,\forall m=1\cdots M,i=1\cdots N; \end{aligned} \tag{3}$$