Recurrently Target-Attending Tracking

Zhen Cui[1], Shengtao Xiao[2], Jiashi Feng[2], Shuicheng Yan[3,2]
[1] Research Center for Learning Science, Southeast University, Nanjing 210096, China
[2] Department of Electrical and Computer Engineering, National University of Singapore, Singapore
[3] 360 Artificial Intelligence Institute, Beijing, China
zhen.cui@seu.edu.cn;{xstgavin124,jshfeng}@gmail.com;eleyans@nus.edu.sg

Abstract

Robust visual tracking is a challenging task in computer vision. Due to the accumulation and propagation of estimation error, model drifting often occurs and degrades the tracking performance. To mitigate this problem, in this paper we propose a novel tracking method called Recurrently Target-attending Tracking (RTT). RTT attempts to identify and exploit those reliable parts which are beneficial for the overall tracking process. To bypass occlusion and discover reliable components, multi-directional Recurrent Neural Networks (RNNs) are employed in RTT to capture long-range contextual cues by traversing a candidate spatial region from multiple directions. The produced confidence maps from the RNNs are employed to adaptively regularize the learning of discriminative correlation filters by suppressing clutter background noises while making full use of the information from reliable parts. To solve the weighted correlation filters, we especially derive an efficient closed-form solution with a sharp reduction in computation complexity. Extensive experiments demonstrate that our proposed RTT is more competitive over those correlation filter based methods.

1. Introduction

Visual object tracking is a fundamental research topic in computer vision with a wide range of applications, including video surveillance, traffic monitoring, and augmented reality [45], etc. Despite the great progress that has been made over the past decades, it remains very challenging due to the existing unpredictable appearance variations such as partial occlusion, illumination change, geometric deformation, background clutter, fast motion, etc.

A typical pipeline of visual tracking starts with an initial location (*e.g.*, a rectangle bounding box) of the object in the first video frame, and then the location of the specified target in the subsequent frames is predicted. Among the existing tracking works, recent part-based methods [1, 35, 7, 31] have been studied actively due to their robustness to local appearance variations, particularly partial occlusion. By partitioning the candidate region of the tracking target into several parts, part-based methods attempt to extract some useful cues from those identified reliable parts. For example, Kwon *et al.* [26] used the topology structure of local patches to find those reliable parts. Zhang *et al.* [46] employed a locality-constrained low-rank and sparse prior to establish correspondences between parts across frames. More recently, Liu *et al.* [31] proposed to learn one response function for each part, and integrated response maps of all parts to generate the final tracking confidence. However, those methods might suffer from some difficulties in capturing large-range spatial dependencies between parts, especially for those objects with a large homogenous region.

In addition to the above methods, many holistic tracking methods have also been developed. In particular, correlation filter (CF) based methods [4, 11, 13, 28, 33, 23] arouse increasing interest due to their excellent efficiency and robustness. CF based methods usually learn a group of correlation filters, which produce correlation peaks for the targets in the scene and meanwhile yield low responses to background regions. The correlation filters are trained by scanning the candidate regions using a circular sliding window. As they adopt the holistically convolutional representation, the whole candidate region is identically treated without any identification during training. This might produce inaccurate filters which cause the learnt tracker to drift away from its correct trajectory, especially when the candidate region embodies cluttered background.

To address the above problems, we propose a novel tracking method called Recurrently Target-attending Tracking (RTT), which attempts to identify and exploit those reliable parts throughout the process of model learning. To discover reliable components, RTT employs multi-directional Recurrent Neural Networks (RNNs) to spatially encode all

* This work was performed when Zhen Cui was a Postdoctoral Fellow at National University of Singapore.

parts from four different angles. The multi-directional RNN offer the following advantages for tracking objects robustly. (i) The spatial recurrent models can learn long-range contextual dependencies between parts, and further produce more accurate detection confidence maps associated with the parts. (ii) Encoding from multiple directions can significantly alleviate negative effects of occlusions that occur in one separated direction. (iii) The generated representation of a target is translation-invariant to some extent as the spatial networks are recurrently performed on local parts. (iv) The multi-directional RNNs are very simple and easy to implement compared to those graphic models with complex structures. Benefited from these aforementioned characteristics, multi-directional RNNs are able to provide a reasonable confidence prediction for the target and background region.

The confidence maps produced from multi-directional RNNs are further used to weight correlation filters in order to suppress negative effects of cluttered background and enhance learning from reliable parts. To this end, we reformulate the correlation filter learning into a regularized version by using confidence maps as the weighting factor. To solve the weighted correlation filters, we propose to factorize the high-dimensional space spanned by those concatenated multi-channel features into low-dimensional spaces on the single channel features, and finally derive an efficient analytical solution to the problem of filter learning. The solution sharply reduces the computation complexity of its direct original solution by a factor of d^3, where d is the number of feature channels. Extensive experiments on the public tracking benchmark dataset demonstrate that our proposed RTT outperforms those existing correlation filter based methods.

In summary, our main contributions are three folds: (i) we propose a part-based confidence map learning method to discover reliable target parts and cluttered background regions; (ii) we develop an adaptively weighted correlation filter method to improve tracking performance by using more reliable information during model updating; and (iii) we derive an efficient closed-form solution to the learning of weighted correlation filters with a sharp reduction in computation complexity.

2. Related Work

Our work is related to the object tracking methods, especially those part-based methods and correlation filter based methods, as well as the current popular recurrent neural network technique.

2.1. Object Tracking

Video object tracking has been extensively studied in computer vision over the past decade [2, 21, 23, 1, 35, 7, 31]. Generally, they fall into two categories: generative models and discriminative models. Generative methods [3, 30, 47, 35] search for the most similar region to the tracked target. The target is often represented by a series of templates or spanned as a subspace. Discriminative methods [2, 21, 23] treat the object tracking as a classifier problem, which learns to distinguish the target from background.

The main related work is the part based and correlation filter based methods. Typically, the part based trackers [1, 35, 7, 31] divide the entire target into several parts. Adam *et al.* [1] represented the object by the grid of fragments, and then voted the target position from these fragments. Jia *et al.* [25] used l_1 sparsity to search the closest candidate patches in the next frame. Besides, many methods explore the topology representation of local parts, such as tree structure [27] or graph structure on superpixels [6]. Different from these methods, we use recurrent neural units to model dependencies of parts from multiple directions, which can not only reduce effects of partial occlusions but also build long-range textural dependencies. Besides, RNNs are simpler and easier to be controlled than those models with tree or graph structures.

Correlation filters have made great progress in visual object tracking [8]. Especially after Minimum Output Sum of Squared Error (MOSSE) [4] filter was proposed, numerous correlation filter methods have been developed. Henriques *et al.* [22] introduced the kernel trick strategy, and Danelljan *et al.* [13] used color attributes to better represent the input information. To deal with the scale problem, SAMF [28], DSST [11] and an improved KCF [23] have been proposed subsequently and achieved state-of-the-art performance. With more methods developed [33, 31, 29], correlation filter based trackers have proven their high-efficiency and robustness. Especially, a regularization work of correlation filter [12] was synchronously developed. But different from [12], RTT adaptively learns more reliable filters by using RNNs, and meanwhile employs high-efficient optimization for filter learning.

Besides, some deep learning based tracking methods [32, 43] have been proposed more recently. They usually focus on learning more robust features by training Convolutional Neural Networks (CNN) on external large-scale datasets. In contrast, the focus of this work is on the pure tracking task where one can only use the first frame. It is straightforward to incorporate CNN into our framework, *e.g.*, directly replacing HOG (used below) with CNN feature maps.

2.2. Recurrent Neural Network

Traditional RNNs learn complex temporal dynamics by mapping input sequences to a sequence of hidden states, and

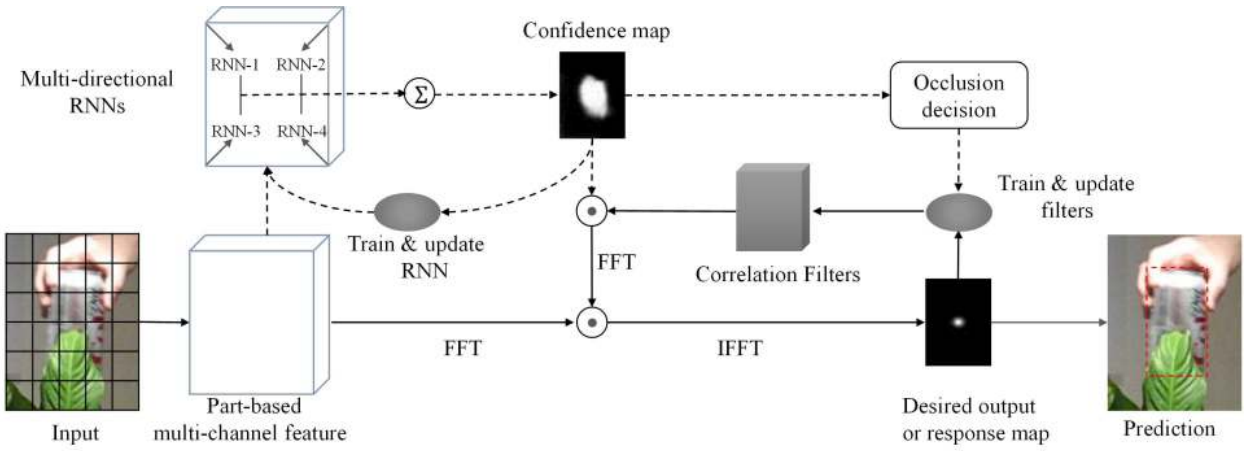


Figure 1. Illustration of our proposed RTT tracker. To identify and exploit those reliable components during tracking, a confidence map is estimated by using multi-directional RNNs, and further used to regularize correlation filters. The dash lines denote the work flow of RNNs. The $\odot$ is an element-wise multiplication operation. More details are described in Section 3.

hidden states to outputs via the recurrence equations:

$$\mathbf{h}_t = \sigma(\mathbf{W}_{xh}\mathbf{x}_t + \mathbf{W}_{hh}\mathbf{h}_{t-1} + \mathbf{b}_h), \quad (1)$$

$$\mathbf{o}_t = \sigma(\mathbf{W}_{ho}\mathbf{h}_t + \mathbf{b}_o), \quad (2)$$

where σ is an element-wise non-linear activation function (*e.g.*, a sigmoid or hyperbolic tangent), $\mathbf{x}_t$ is the t-th input (frame), $\mathbf{h}_t$ is the corresponding hidden state, and $\mathbf{o}_t$ is the predicted output at time t. Given an input sequence $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_T\}$ with length T, the inference is computed sequentially as $\{\mathbf{h}_1, \mathbf{o}_1, \mathbf{h}_2, \mathbf{o}_2, \cdots, \mathbf{h}_T, \mathbf{o}_T\}$ with the initial hidden state $\mathbf{h}_0 = \mathbf{0}$.

Recently, RNNs [36], including the long-term version called Long Short-Term Memory (LSTM) model [24] and Gated Recurrent Unit (GRU) [9], have attracted increasing attention in modeling sequential data. The applications cover multilingual machine translation [41], action recognition [14, 15], scene labeling [39, 34], speech recognition [19], etc. More recently, the conventional RNNs are generalized into more complex structure models, such as 2D RNNs [20, 37], multi-dimensional RNNs [20, 40, 5], tree RNNs [42, 48], etc. Our method is a simple generalization of their methods, and to the best of our knowledge, it is the first attempt to adapt RNNs to modeling the complex long-range dependencies for the tracking task.

3. Recurrently Target-Attending Tracker

In this section, first we give an overview of the proposed RTT tracker. Then we illustrate the generation of confidence maps and the learning process of discriminative filters in details.

3.1. Overview

An overview on the RTT framework is shown in Fig. 1. Given a video frame, we first determine a small candidate region of 2.5 times the size of the bounding box surrounding the localization result in the previous frame, considering that the motion between continuous video frames is usually subtle. For this candidate region, a grid-like partition is used to produce visual parts, and the feature from each part is extracted for the next tracking. In practice, some descriptors pooled on spatial grids such as HOG [17] or high-level features from CNN [38] can be utilized. Thereafter we can obtain the part-based feature $\mathbf{X} \in \mathbb{R}^{h\times w\times d}$ of d channels for each candidate region, where h, w are the height and the width of spatial parts/grids.

RTT attempts to identify those reliable parts and then utilize them for robust tracking. As intimate interactions exist among those spatially adjacent parts and even disjoint parts, between-part relationships can offer valuable contextual information beyond purely relying on a single part. However, the interactions of parts in a 2D space are far more complex than Markov chain structures. Here we employ recurrent neural network to characterize the parts and their complex dependencies, since it is simper and capable of gleaning long-range contextual cues. Moreover, to compensate the inadequacy of a single RNN used in a 2D space, we use several spatial RNNs (*e.g.*, quaddireactional RNNs) to traverse the spatial candidate region from different angles. Such a strategy can effectively alleviate the contamination from partial occlusion or local appearance variation during tracking. The spatial RNNs produces confidence scores for each part, which compose a confidence map for the whole candidate region. The confidence map factually represents the probability of every part being background or target. Thus the confidence map may be used to predict the existence of occlusions and guide the model update. More details about the confidence map generation are described in Section 3.2.

Furthermore, the confidence map can be incorporated in-

to the learning of a discriminative tracker. As the conventional correlation filter trackers usually treat all parts identically, the incremental learning tends to produce results deviating from the expected trajectory due to its sensitiveness to noises from clutter background or occluded regions. Fortunately, the confidence map produced from RNNs can reflect the reliability of candidate region to some extent. Thus the confidence map may be employed to adaptively mask correlation filters to resist those negative effects of cluttered backgrounds or partial occlusions during filter learning. The filter weighting strategy makes RTT more robust to alleviate model drifting due to the use of reliable components during model updating. Similar to those correlation filter based methods, RTT conducts the learning process in the frequency domain. More details about the learning of weighted filters are given in Section 3.3.

As aforementioned, during model training and updating, RTT learns more discriminative correlation filters by adaptively regularizing the filters with the confidence map. In testing, RTT simply employs the learnt discriminative filters to detect the target as discriminative information has adaptively permeated into the tracker.

3.2. Spatial Confidence Map

The recurrent neural network has the memory ability endowed by its repetitive connections. A popular model is the recently proposed long short-term memory (LSTM) [24]. However, LSTM has a high-freedom parameter space. Searching parameters in such a large space suffers from the risk of overfitting specifically for the online tracking task, where training samples are usually scarce. To address this problem, here we choose the conventional RNN unit, which has a few model parameters to be solved.

Concretely, we suppose the candidate parts $\mathbf{X}$ are represented by a graph $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$, where $\mathcal{V} = \{\mathbf{x}_{ij}\}(i = 1, \cdots, h, j = 1, \cdots, w)$ denotes the vertex set of parts indexed by their spatial coordinates, and $\mathcal{E} = \{e_{ij,kl}\}$ represents the edges of spatial neighboring parts. By traversing through $\mathcal{G}$ with a predefined forward evolution sequence, we can define the input state and previous states for a recurrent neural network unit. Here the only requirement for traversing is that one node cannot be processed until the processing of its predecessors is finished. Formally, the adopted multi-directional RNNs in RTT can be written as

$$\mathbf{h}_{ij}^r = \sigma_1(\mathbf{U}^r\mathbf{x}_{ij} + \sum_{(k,l)\in\mathcal{N}_{ij}} \mathbf{W}^r\mathbf{h}_{kl}^r + \mathbf{b}^r), \quad (3)$$

$$\mathbf{o}_{ij} = \sigma_2(\sum_{r\in\mathcal{D}} \mathbf{V}^r\mathbf{h}_{ij}^r + \mathbf{b}), \quad (4)$$

where $\mathbf{x}_{ij}, \mathbf{h}_{ij}^r, \mathbf{o}_{ij}$ respectively denote the representation of input, hidden and output node at the location of (i, j), $\mathcal{N}_{ij}$ is the set of predecessors of the vertex (i, j) in the graph $\mathcal{G}$, and r represents a traversing direction in $\mathcal{D}$. $\mathbf{h}_{ij}^r$ collects information of all the predecessors of the current state (i, j), and the output summarizes the stimulus from all directions $\mathcal{D}$. The learned parameters $\{\mathbf{U}^r, \mathbf{W}^r, \mathbf{b}^r, \mathbf{V}^r, \mathbf{b}\}$ are recurrently utilized in traversing the graph $\mathcal{G}$. Here the non-linear function σ_1 for hidden layers is ReLU [10].

To make the traversing information from traversing processes mutually complementary, we consider four traversing directions starting from four angular points. For example, the directional traversing from the top-left corner is responsible for capturing contextual cues about the top-left areas, with the adjacent predecessor set $\mathcal{N}_{ij} = \{(i, j-1), (i-1, j-1), (i-1, j)\}$. Thus, in a 2D spatial plane, by connecting contiguous parts and traversing these parts respectively from four directions, four directed acyclic chains can be generated to represent the 2D neighborhood system. With successive propagations in chains as formulated in Eqn. (3), the interactions among parts can be achieved.

To obtain a probability map in the output layer, we use the standard softmax function, *i.e.*, $\sigma_2(x) = \frac{\exp^{x_i}}{\sum_k \exp^{x_k}}$. Thus the cross entropy loss can be naturally used as the objective function:

$$E = -\sum_{(i,j)}\sum_{c\in\mathcal{C}} y_{ij}^c \ln Pr(c|\mathbf{x}_{ij}), \quad (5)$$

where $y \in \mathcal{C} = \{0, 1\}$ is the expected binary indicator of being background or target regions, and $Pr(\cdot)$ is the output probability of this model. In model training and updating, we simply assign the label 1 to those parts within the localized bounding box while 0 for outside parts because we do not have accurate labels.

3.3. Weighted Correlation Filter

The correlation filter based methods are to learn a group of filters $\{\mathbf{f}^k\}, k = 1, \cdots, d$, each for one feature channel in $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_d\}$. The learning of a weighted correlation filter can be formally written as minimizing the following loss function:

$$\varsigma(\mathbf{f}) = \|\sum_{k=1}^{d} \mathbf{x}^k * \mathbf{f}^k - \mathbf{y}\|^2 + \sum_{k=1}^{d} \|\mathbf{w} \odot \mathbf{f}^k\|^2, \quad (6)$$

where $*$ denotes a spatial convolution, $\odot$ is an elementwise multiplication operation, $\mathbf{f}^k$ convolves with the k-th channel features, and the weight $\mathbf{w}$ regularizes the correlation filters by using the confidence map produced from the multi-directional RNNs. According to Parseval theorem, the objective function in Eqn. (6) is equivalent to the following loss function in the frequency domain:

$$\varsigma(\widehat{\mathbf{f}}) = \|\sum_{k=1}^{d} \widehat{\mathbf{x}}^k \odot \widehat{\mathbf{f}}^k - \widehat{\mathbf{y}}\|^2 + \lambda \sum_{k=1}^{d} \|\widehat{\mathbf{w}} * \widehat{\mathbf{f}}^k\|^2, \quad (7)$$

where $\widehat{\cdot}$ denotes the FFT of the involved variable, and the constant factor λ is a balance parameter, which is simply set to $\frac{1}{h^2w^2}$ according to the practical implementation of FFT [23].

To vectorize the objective function, we introduce some extra notations. Let $\widehat{\mathbf{X}} = \mathrm{diag}(\widehat{\mathbf{x}})$ denote the diagonal matrix with diagonal elements from the vector $\widehat{\mathbf{x}}$. $\mathcal{W}$ represents a circulant matrix by shifting its basis vector $\widehat{\mathbf{w}}$ in each row, with the first row being set to $\widehat{\mathbf{w}}$. Therefore, the objective function in Eqn. (7) can be written as

$$\varsigma(\widehat{\mathbf{f}}) \quad = \quad \|\sum_{k=1}^{d} \widehat{\mathbf{X}}^k \widehat{\mathbf{f}}^k - \widehat{\mathbf{y}}\|^2 + \lambda \sum_{k=1}^{d} \|\mathcal{W}\widehat{\mathbf{f}}^k\|^2. \quad (8)$$

However, the above equation is defined in a complex domain, where some theories cannot directly apply from the real space. To transform $\varsigma(\widehat{\mathbf{f}})$ into the real domain, we decompose each complex value in the vector/matrix into two real values corresponding to its real and imaginary part respectively. Concretely, the matrix and the vector are respectively expanded in the following formulation:

$$\begin{bmatrix} a_{11}+b_{11}i, \cdots a_{1n}+b_{1n}i \\ \vdots \quad\quad \vdots \quad\quad \vdots \\ a_{n1}+b_{n1}i, \cdots a_{nn}+b_{nn}i \end{bmatrix} \rightarrow \begin{bmatrix} a_{11}, -b_{11}, \cdots a_{1n}, -b_{1n} \\ b_{11}, \ a_{11}, \ \cdots b_{1n}, \ a_{1n} \\ \vdots \quad \vdots \quad \vdots \quad \vdots \quad \vdots \\ a_{n1}, -b_{n1}, \cdots a_{nn}, -b_{nn} \\ b_{n1}, \ a_{n1}, \ \cdots b_{nn}, \ a_{nn} \end{bmatrix},$$

$$\begin{bmatrix} c_{11}+d_{11}i, \cdots c_{1n}+d_{1n}i \end{bmatrix}^\intercal \rightarrow \begin{bmatrix} c_{11}, d_{11}, \cdots c_{1n}, d_{1n} \end{bmatrix}^\intercal.$$

Thus the objective function in Eqn. (8) equates with the following function in the real domain:

$$\varsigma(\widetilde{\mathbf{f}}) \quad = \quad \|\sum_{k=1}^{d} \widetilde{\mathbf{X}}^k \widetilde{\mathbf{f}}^k - \widetilde{\mathbf{y}}\|^2 + \lambda \sum_{k=1}^{d} \|\widetilde{\mathbf{W}}\widetilde{\mathbf{f}}^k\|^2, \quad (9)$$

where $\widetilde{\mathbf{X}}, \widetilde{\mathbf{W}} \in \mathbb{R}^{2n\times 2n}, \widetilde{\mathbf{f}}, \widetilde{\mathbf{y}} \in \mathbb{R}^{2n}$ are real-valued matrices/vectors corresponding to $\widehat{\mathbf{X}}, \mathcal{W}, \widehat{\mathbf{f}}, \widehat{\mathbf{y}}$, and $n = h \times w$.

The loss function in Eqn. (9) can be further simplified by introducing matrix calculation. By defining the concatenation matrix $\mathbf{X} = [\widetilde{\mathbf{X}}^1, \cdots, \widetilde{\mathbf{X}}^d] \in \mathbb{R}^{2n\times 2nd}, \mathbf{f} = [(\widetilde{\mathbf{f}}^1)^\intercal, \cdots, (\widetilde{\mathbf{f}}^d)^\intercal]^\intercal \in \mathbb{R}^{2nd}, \mathbf{W} = \mathrm{diag}([\widetilde{\mathbf{W}}, \widetilde{\mathbf{W}}, \cdots, \widetilde{\mathbf{W}}]) \in \mathbb{R}^{2nd\times 2nd}$, where $\mathbf{W}$ is a block diagonal matrix with the diagonal element being $\widetilde{\mathbf{W}}$, we have a loss function in a more concise form,

$$\varsigma(\mathbf{f}) \quad = \quad \|\mathbf{X}\mathbf{f} - \widetilde{\mathbf{y}}\|^2 + \lambda\|\mathbf{W}\mathbf{f}\|^2. \quad (10)$$

Obviously, this loss function has a closed-form minimum solution obtained by setting its differential to be zero, *i.e.*,

$$\mathbf{f} = (\mathbf{X}^\intercal\mathbf{X} + \lambda\mathbf{W}^\intercal\mathbf{W})^{-1}\mathbf{X}^\intercal\widetilde{\mathbf{y}}. \quad (11)$$

However, we have to calculate an inverse operation on a real matrix of size $2nd \times 2nd$, which dominates the computation cost of the optimization. As multiple channel features are usually employed, the computation complexity on the matrix inverse is usually $O(n^3d^3)$, which is too expensive for real-world applications. Fortunately, we develop an efficient solution which only requires computing the inverse of a matrix of size $2n \times 2n$. It reduces the computation complexity by a factor of d^3, which is quite significant for high-dimensional features. The solution is presented in the following proposition.

Proposition 1. *Suppose* $\mathbf{W}$ *is invertible*[1]. *The optimal solution* $\mathbf{f}$ *to Eqn. (9) is*

$$\mathbf{f} \quad = \quad \frac{1}{\lambda}\mathbf{G}(\mathbf{I} - (\lambda\mathbf{I} + \mathbf{H})^{-1}\mathbf{H})\widetilde{\mathbf{y}}, \quad (12)$$

where $\mathbf{G} = [\widetilde{\mathbf{X}}^1(\widetilde{\mathbf{W}}^\intercal\widetilde{\mathbf{W}})^{-1}, \cdots, \widetilde{\mathbf{X}}^d(\widetilde{\mathbf{W}}^\intercal\widetilde{\mathbf{W}})^{-1}]^\intercal$ *and* $\mathbf{H} = \sum_{k=1}^{d} \widetilde{\mathbf{X}}^k(\widetilde{\mathbf{W}}^\intercal\widetilde{\mathbf{W}})^{-1}(\widetilde{\mathbf{X}}^k)^\intercal$.

Proof. As the matrix $\mathbf{W}$ is invertible, the matrix $\mathbf{P} = \mathbf{W}^\intercal\mathbf{W}$ is symmetrical and positive definite. Performing Singular Value Decomposition (SVD) on $\mathbf{P}$ can be written as $\mathbf{P} = \mathbf{S}\mathbf{V}\mathbf{S}^\intercal$, where $\mathbf{S}$ is an orthonormal matrix and $\mathbf{V}$ is a diagonal matrix with nonnegative elements. Let $\mathbf{Q}$ denote the term $(\mathbf{X}^\intercal\mathbf{X} + \lambda\mathbf{W}^\intercal\mathbf{W})$ in Eqn. (11), according to general matrix algebras we have

$$\begin{aligned} \mathbf{Q}^{-1} &= (\mathbf{X}^\intercal\mathbf{X} + \lambda\mathbf{W}^\intercal\mathbf{W})^{-1} = (\mathbf{X}^\intercal\mathbf{X} + \lambda\mathbf{S}\mathbf{V}\mathbf{V}\mathbf{S}^\intercal)^{-1} \\ &= \mathbf{S}\mathbf{V}^{-1}(\mathbf{V}^{-1}\mathbf{S}^\intercal\mathbf{X}^\intercal\mathbf{X}\mathbf{S}\mathbf{V}^{-1} + \lambda\mathbf{I})^{-1}\mathbf{V}^{-1}\mathbf{S}^\intercal. \quad (13) \end{aligned}$$

Let $\mathbf{U} \triangleq \mathbf{X}\mathbf{S}\mathbf{V}^{-1}$, by using the Woodbury matrix identity, we have

$$\begin{aligned} \mathbf{Q}^{-1} &= \mathbf{S}\mathbf{V}^{-1}(\mathbf{U}^\intercal\mathbf{U} + \lambda\mathbf{I})^{-1}\mathbf{V}^{-1}\mathbf{S}^\intercal \\ &= \mathbf{S}\mathbf{V}^{-1}(\frac{1}{\lambda}(\mathbf{I} - \mathbf{U}^\intercal(\lambda\mathbf{I} + \mathbf{U}\mathbf{U}^\intercal)^{-1}\mathbf{U}))\mathbf{V}^{-1}\mathbf{S}^\intercal, \quad (14) \end{aligned}$$

According to the definition of $\mathbf{W}$ and $\mathbf{X}$, and $(\widetilde{\mathbf{W}}^\intercal\widetilde{\mathbf{W}})^{-1} = \mathbf{S}\mathbf{V}^{-1}\mathbf{V}^{-1}\mathbf{S}^\intercal$, we can derive the solution in Eqn. (12). □

As $\widetilde{\mathbf{X}}^k$ is produced from a diagonal complex matrix according to the above expansion, it is highly sparse. Thus the matrix multiplication with $\widetilde{\mathbf{X}}^k$ can be implemented by simply switching matrix rows/columns and performing an element-wise multiplication operation. Therefore, the overall computation complexity is $O(n^3+dn^2)$, which is significantly smaller than $O(d^3n^3)$ of directly solving Eqn. (11).

4. Implementation

Here we present some implementation details including feature extraction, occlusion decision, scale estimation and model updating.

Feature Extraction. Here we use a variant of HOG [17], which is popular in the tracking task. The HOG features

[1]In practice, a small regularization term $\epsilon\mathbf{I}$ may be used to avoid its singularity.

are sampled from a series of spatial grids with 4×4 pixels and then are quantized into 31 bins. We do not use other features such as color information or even convolutional features, even though more robust features can promote the tracking performance, since our aim is to explore some intrinsic effects to mitigate model drifting problem in the tracking task.

Scale Estimation. Similar to [28], we use the multiple scale searching technique to estimate changes of the target size, where the scaling factors are defined as $\{0.985, 0.99, 0.995, 1.0, 1.005, 1.01, 1.015\}$.

Occlusion Decision. The confidence map produced from spatial RNNs is used to predict the existence of occlusion. When the object is predicted to be occluded with a high probability, the model is not updated. Concretely, we define the entire confidence score as an accumulation of the probability values within the target region. If the current score is less than a certain ratio τ of the average score of previous frames, the current frame is considered to be occluded. In practice, the threshold τ is set to 0.85.

RNN Training and Updating. The standard Back-Propagation Through Time (BPTT) strategy is used for training RNN. In spatial RNNs, the dimension of hidden layers is the same as the number of channels. As the training samples are insufficient, we employ the first five frames to train spatial RNNs with a learning rate of 0.02. The RNNs are updated in the subsequent frames with a fixed interval of five frames. To avoid over-fitting in the current frame, we employ a small learning rate 0.001 and a few iteration times 100 in fine-tuning. The learning momentum is fixed as 0.9.

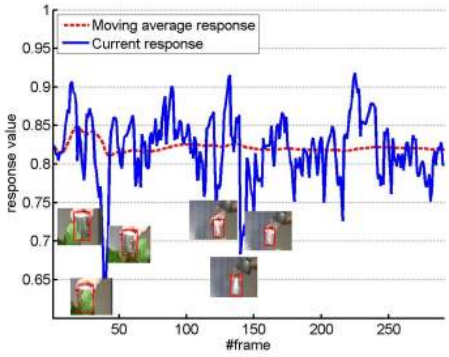
Filter Updating. The update procedure is straightforward except that the first frame is used to initialize the model. Similar to previous correlation filter based methods, we linearly combine the new filter with the old one as below:

$$\bar{\mathbf{f}} = \theta \mathbf{f}_{new} + (1 - \theta)\bar{\mathbf{f}}, \tag{15}$$

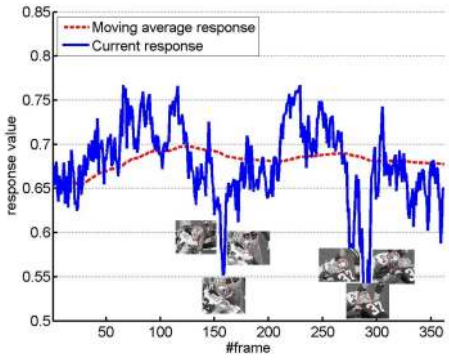
where the learning factor θ is set to 0.025.

5. Experiments

In the following experiments, two evaluation criteria are used. The first one is mean center location error (CLE), *i.e.*, the difference between ground truth and prediction results. A smaller CLE means a more accurate tracking result. The second one is the Pascal VOC Overlap ratio (VOR) [16], which is defined as $VOR = Area(B_T \cap B_G)/Area(B_T \cup B_G)$, where B_G is the bounding box of ground truth, and B_T is the predicted bounding box. A bigger value indicates a more accurate prediction. We employ all the 51 video sequences in the popular benchmark [44] to extensively evaluate our method.



(a) Coke Sequence

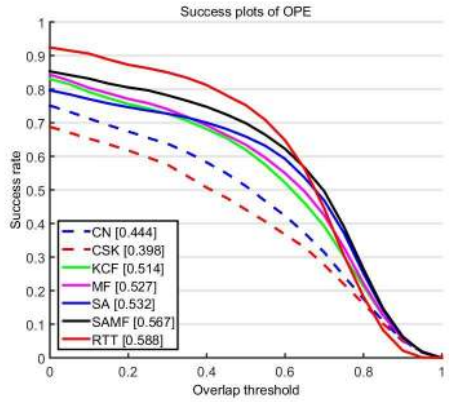


(b) Football Sequence

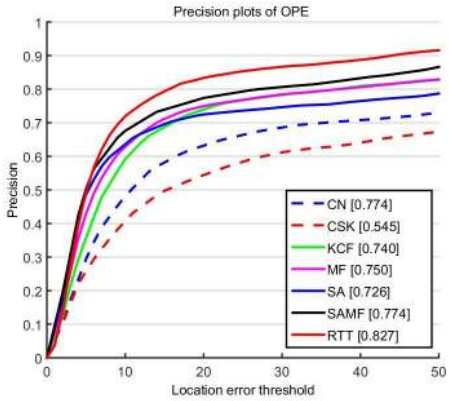
Figure 2. Examples of confidence maps. The blue line represents the probability of the current bounding box, and the red line denotes the average probability of the previous observation. The valleys below the average line means the possible occurrence of dramatic appearance variations such as occlusion, deformation and illumination changes.

5.1. Appearance Variation-Attending Prediction

To investigate the effectiveness of multi-directional RNNs for predicting appearance variations, in Fig. 2 we show two video sequences covering diverse appearance variations including object occlusion, deformation and illumination changes. We take the average value of scores of parts within the bounding box as the response value of the bounding box. To predict the existence of occlusion, we use the average of previous responses as a reference value, as depicted in real lines in Fig. 2. The valleys below the moving average lines indicate that there are dramatic appearance variations in the corresponding frame. For example, the two valleys in Fig. 2(a) are caused by partial occlusion and glaring lights. Based on the observation, we can conclude that adaptively updating the model according to the current state is necessary for reducing the artifacts incurred by dramatic appearance variations.

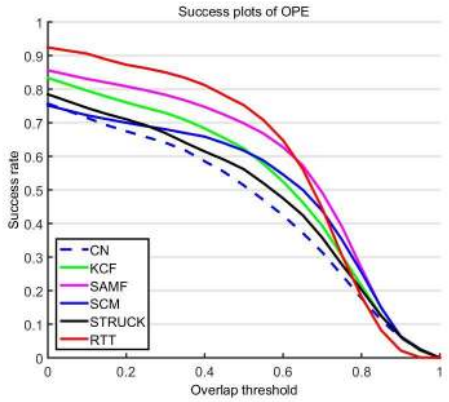


(a) VOR plot

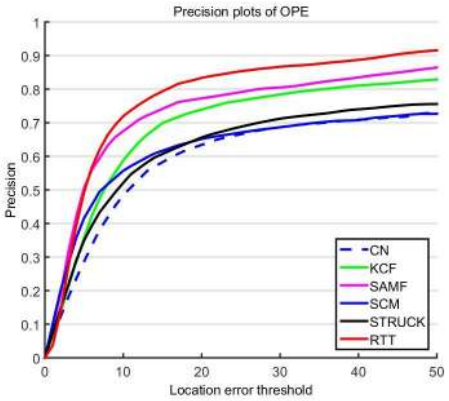


(b) CEL plot

Figure 3. Comparisons with correlation filter based methods.



(a) VOR plot



(b) CEL plot

Figure 4. Comparisons with five state-of-the-art methods.

5.2. Comparison with Correlation Filter Based Trackers

To evaluate the performance gain of our proposed RTT, we compare six correlation filter based methods, including CN [13], CSK [22], KCF [23], SA [28], MF [28] and SAMF [28]. As RTT only employs the HOG feature, SA should be a standard baseline. SA is a scale adaptive correlation filter method that also uses HOG feature. Fig. 3 shows VOR curves and CEL curves of One-Pass Evaluation (OPE) [44] for these compared trackers on the benchmark dataset. Although all of these trackers use the circulant filters, their tracking performance is quite different. CSK only employs the raw features, thus it gives worst results among the compared methods. CN exploits the color features and improves the performance. The remaining compared methods adopt the robust HOG feature. Moreover, SAMF and MF also fuse gray and color information as their features. Compared with the standard baseline SA, the performance gains of RTT are about 5% and 10% respectively in terms of VOR and CEL metrics. Among previous CF based methods, SAMF achieves the best performance with a VOR score of 56.7% and a CEL score of 77.4%. Our proposed RTT approach outperforms the SAMF tracker by about 2.1% and 4.7% in terms of VOR and CEL curves respectively. The experiments suggest that using reliable part information can improve the tracking performance.

5.3. Comparison with State-of-the-art Trackers

Fig. 4 provides the comparisons between RTT and well-established state-of-the-art methods on the benchmark dataset [44], including KCF [23], SCM [47], STRUCK [21], CN [13], SAMF [28], TGPR [18], and DSST [11]. Only the results of the top five trackers are reported. Correspondingly, the performance on the specific video sequences including object deformation, occlusion, and out-of-plane is shown in Fig. 5. From the results, we can observe that the proposed RTT achieves very appealing performance. Especially, the location errors of our proposed RTT are largely reduced on the sequences with dramatic appearance variations, which benefits from regularizing the correlation filter on those reliable parts. As shown in these VOR curves, RTT has slightly inferior performance on estimating accurate bounding boxes with an overlap threshold

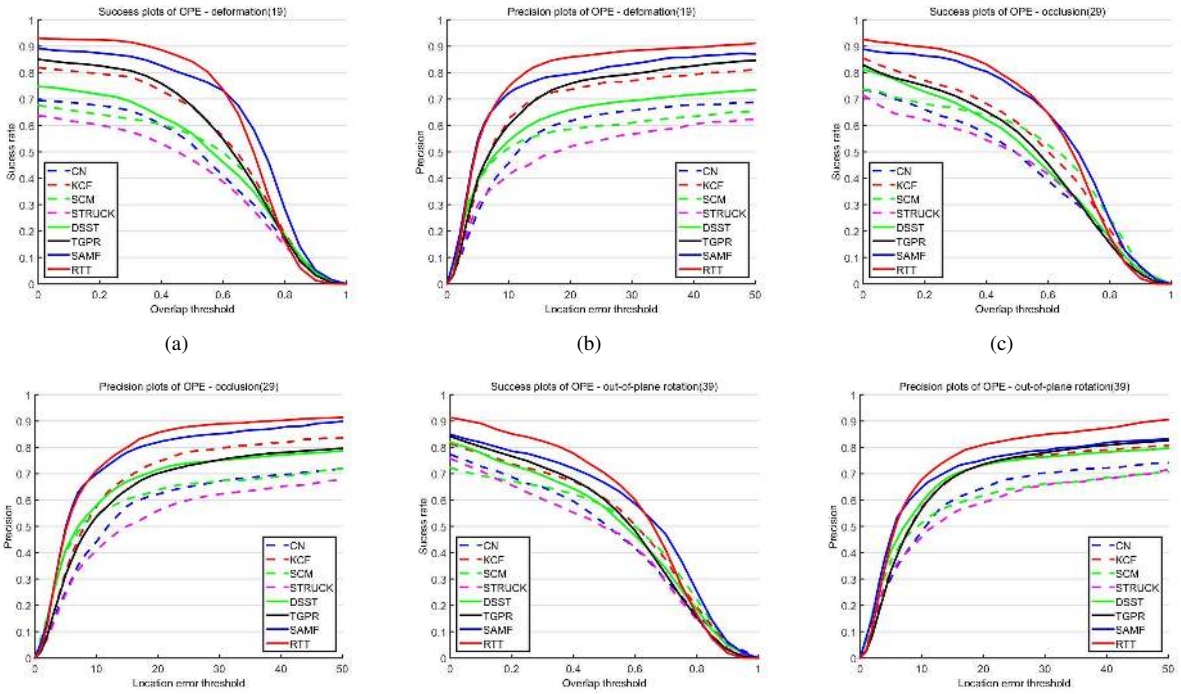


Figure 5. The plot curves on the tracking targets with deformation, occlusion and out-of-plane rotation. Numbers in parentheses are quantity of corresponding videos.

larger than 0.8. A possible reason may be that the confidence map estimated by multi-directional RNNs incorporates a few errors, which might contaminate the scale estimation.

5.4. Discussion

The proposed model predicts confidence scores on local parts, so it gives a rough probability estimation for the candidate regions. More accurate estimations at finer-grained level such as superpixel level should be meaningful for more robust tracking. A possible direction is to simultaneously perform object segmentation and tracking. However, this will lead to higher computation cost. Currently, our implementation runs about 3$\sim$4 Fps using the non-optimized python code on a general PC (2.80GHz, 16G Memory). The main time cost is still spent on the matrix inversion even though we have reduced the computation complexity by a factor of d^3. A future work to speed up the algorithm by approximating matrix inversion needs to be explored.

6. Conclusion

In this paper, we introduce Recurrently Target-attending Tracker (RTT) to identify and utilize those reliable components and achieve better tracking results. To efficiently find those reliable components, we employ the quaddirectional spatial recurrent neural networks to traverse the whole candidate region from different angles. Due to modeling local parts and their dependencies, recurrent networks are shown to be able to capture some invariant and reliable information even though partial occlusion exists. The produced confidence map from recurrent neural networks is shown to be effective for predicting the existence of occlusion. At the same time, the confidence map is used to weight the correlation filters during training, which successfully suppresses some clutter background information and makes full use of reliable components. To learn discriminative filters, we provide an accurate analytic solution with low computation complexity. Finally, we obtain encouraging empirical results from our extensive experiments compared with several state-of-the-art trackers.

Acknowledgement

This work was partly supported by the National Basic Research Program of China under Grant 2015CB351704. J. Feng is supported by NUS startup grant R-263-000-C08-133. This work was also partly supported by the National Natural Science Foundation of China (NSFC) under Grants 61231002 and the Natural Science Foundation of Jiangsu Province under Grant BK20130020.

References

[1] A. Adam, E. Rivlin, and I. Shimshoni. Robust fragments-based tracking using the integral histogram. In *CVPR*, 2006.

[2] S. Avidan. Support vector tracking. *T-PAMI*, 26(8):1064–1072, 2004.

[3] B. Babenko, M.-H. Yang, and S. Belongie. Visual tracking with online multiple instance learning. In *CVPR*, 2009.

[4] D. S. Bolme, J. R. Beveridge, B. Draper, Y. M. Lui, et al. Visual object tracking using adaptive correlation filters. In *CVPR*, 2010.

[5] W. Byeon, T. M. Breuel, F. Raue, and M. Liwicki. Scene labeling with lstm recurrent neural networks. In *CVPR*, 2015.

[6] Z. Cai, L. Wen, J. Yang, Z. Lei, and S. Z. Li. Structured visual tracking with dynamic graph. In *ACCV*. 2013.

[7] L. Cehovin, M. Kristan, and A. Leonardis. Robust visual tracking using an adaptive coupled-layer visual model. *T-PAMI*, 35(4):941–953, 2013.

[8] Z. Chen, Z. Hong, and D. Tao. An experimental survey on correlation filter-based tracking. *arXiv preprint arXiv:1509.05520*, 2015.

[9] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[10] G. E. Dahl, T. N. Sainath, and G. E. Hinton. Improving deep neural networks for lvcsr using rectified linear units and dropout. In *ICASSP*, 2013.

[11] M. Danelljan, G. Häger, F. Khan, and M. Felsberg. Accurate scale estimation for robust visual tracking. In *BMVC*, 2014.

[12] M. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg. Learning spatially regularized correlation filters for visual tracking. In *ICCV*, 2015.

[13] M. Danelljan, F. S. Khan, M. Felsberg, and J. van de Weijer. Adaptive color attributes for real-time visual tracking. In *CVPR*, 2014.

[14] J. Donahue, L. A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell. Long-term recurrent convolutional networks for visual recognition and description. *arXiv preprint arXiv:1411.4389*, 2014.

[15] Y. Du, W. Wang, and L. Wang. Hierarchical recurrent neural network for skeleton based action recognition. In *CVPR*, 2015.

[16] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 88(2):303–338, 2010.

[17] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *T-PAMI*, 32(9):1627–1645, 2010.

[18] J. Gao, H. Ling, W. Hu, and J. Xing. Transfer learning based visual tracking with gaussian processes regression. In *ECCV*. 2014.

[19] A. Graves, A.-r. Mohamed, and G. Hinton. Speech recognition with deep recurrent neural networks. In *ICASSP*, 2013.

[20] A. Graves and J. Schmidhuber. Offline handwriting recognition with multidimensional recurrent neural networks. In *NIPS*, 2009.

[21] S. Hare, A. Saffari, and P. H. Torr. Struck: Structured output tracking with kernels. In *ICCV*, 2011.

[22] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. Exploiting the circulant structure of tracking-by-detection with kernels. In *ECCV*. 2012.

[23] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. High-speed tracking with kernelized correlation filters. *T-PAMI*, 37(3):583–596, 2015.

[24] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

[25] X. Jia, H. Lu, and M.-H. Yang. Visual tracking via adaptive structural local sparse appearance model. In *CVPR*, 2012.

[26] J. Kwon and K. M. Lee. Tracking of a non-rigid object via a patch-based dynamic appearance modeling and adaptive basin hopping monte carlo sampling. In *CVPR*, 2009.

[27] J. Kwon and K. M. Lee. Highly nonrigid object tracking via patch-based dynamic appearance modeling. *T-PAMI*, 35(10):2427–2441, 2013.

[28] Y. Li and J. Zhu. A scale adaptive kernel correlation filter tracker with feature integration. In *ECCV Workshops*, 2014.

[29] Y. Li, J. Zhu, and S. C. Hoi. Reliable patch trackers: Robust visual tracking by exploiting reliable patches. In *CVPR*, 2015.

[30] B. Liu, J. Huang, L. Yang, and C. Kulikowsk. Robust tracking using local sparse appearance model and k-selection. In *CVPR*, 2011.

[31] T. Liu, G. Wang, and Q. Yang. Real-time part-based visual tracking via adaptive correlation filters. *Intelligence*, page 2345390, 2015.

[32] C. Ma, J.-B. Huang, X. Yang, and M.-H. Yang. Hierarchical convolutional features for visual tracking. In *ICCV*, 2015.

[33] C. Ma, X. Yang, C. Zhang, and M.-H. Yang. Long-term correlation tracking. In *CVPR*, 2015.

[34] P. Pinheiro and R. Collobert. Recurrent convolutional neural networks for scene labeling. In *ICML*, 2014.

[35] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang. Incremental learning for robust visual tracking. *IJCV*, 77(1-3):125–141, 2008.

[36] M. Schuster and K. K. Paliwal. Bidirectional recurrent neural networks. *IEEE Transactions on Signal Processing*, 45(11):2673–2681, 1997.

[37] B. Shuai, Z. Zuo, and G. Wang. Quaddirectional 2d-recurrent neural networks for image labeling. *IEEE Signal Processing Letters*, 22(11):1990–1994, 2015.

[38] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[39] R. Socher, C. C. Lin, C. Manning, and A. Y. Ng. Parsing natural scenes and natural language with recursive neural networks. In *ICML*, 2011.

[40] M. F. Stollenga, W. Byeon, M. Liwicki, and J. Schmidhuber. Parallel multi-dimensional lstm, with application to fast biomedical volumetric image segmentation. *arXiv preprint arXiv:1506.07452*, 2015.

[41] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *NIPS*, 2014.

[42] K. S. Tai, R. Socher, and C. D. Manning. Improved semantic representations from tree-structured long short-term memory networks. *arXiv preprint arXiv:1503.00075*, 2015.

[43] L. Wang, W. Ouyang, X. Wang, and H. Lu. Visual tracking with fully convolutional networks. In *ICCV*, 2015.

[44] Y. Wu, J. Lim, and M.-H. Yang. Online object tracking: A benchmark. In *CVPR*, 2013.

[45] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *Acm computing surveys (CSUR)*, 38(4):13, 2006.

[46] T. Zhang, K. Jia, C. Xu, Y. Ma, and N. Ahuja. Partial occlusion handling for visual tracking via robust part matching. In *CVPR*, 2014.

[47] W. Zhong, H. Lu, and M.-H. Yang. Robust object tracking via sparsity-based collaborative model. In *CVPR*, 2012.

[48] X. Zhu, P. Sobhani, and H. Guo. Long short-term memory over tree structures. *arXiv preprint arXiv:1503.04881*, 2015.