Robust Object Tracking based on Temporal and Spatial Deep Networks

Zhu Teng[†], Junliang Xing[‡], Qiang Wang[‡], Congyan Lang[†], Songhe Feng[†], Yi Jin[†]

[†]School of Computer and Information Technology, Beijing Jiaotong University, Beijing 100044, China

[‡]National Laboratory of Pattern Recognition, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China

{zteng,cylang,shfeng,yjin}@bjtu.edu.cn, jlxing@nlpr.ia.ac.cn, wangqiang2015@ia.ac.cn

Abstract

Recently deep neural networks have been widely employed to deal with the visual tracking problem. In this work, we present a new deep architecture which incorporates the temporal and spatial information to boost the tracking performance. Our deep architecture contains three networks, a Feature Net, a Temporal Net, and a Spatial Net. The Feature Net extracts general feature representations of the target. With these feature representations, the Temporal Net encodes the trajectory of the target and directly learns temporal correspondences to estimate the object state from a global perspective. Based on the learning results of the Temporal Net, the Spatial Net further refines the object tracking state using local spatial object information. Extensive experiments on four of the largest tracking benchmarks, including VOT2014, VOT2016, OTB50, and OTB100, demonstrate competing performance of the proposed tracker over a number of state-of-the-art algorithms.

1. Introduction

Visual object tracking is a fundamental problem in computer vision and can be applied in many practical systems like video surveillance [13], vehicle navigation [53], and human-machine interaction [35]. One common setting for this problem is that a bounding box of a target is provided in the first frame and the tracking objective is to predict locations of the target in subsequent frames. Although lots of efforts have been devoted into this problem and extensive studies are conducted in the past decades [4, 59], it still remains a very difficult problem due to challenges like scale variation, fast motion, occlusions, deformation, and background cluttering [58].

In order to overcome different kinds of challenges and achieve robust tracking results, previous works generally focus on some of the most important components in a visual tracking system, such as robust object appearance representation [20, 22, 40, 44], sophisticated object motion model [29, 30, 34], and adaptive object model updating

[9, 39, 46]. Conventional methods design the tracking observation model using different representations, some of the most typical ones include the subspace representation [46], the sparse representation [24, 40, 63], the structured representation [20, 27], and the correlation filter based representation [22, 61]. Recently the Convolutional Neural Network (CNN) based representations have been employed in many computer vision problems [18, 28, 36, 42, 55] due to its strong representation power and vast modeling capacity. Inspired by these successes, we in this work design a Temporal-Spatial Network (TSN) model to deal with different kinds of challenges in visual tracking.

As the power of deep learning models generally comes into effect when a large amount of labeled training samples are applied, it is not straightforward to directly deploy a deep learning model into an online visual tracking algorithm, because the only labeled sample for object tracking problem is the target annotated in the first frame. Previous deep learning model based tracking algorithms need numerous labeled videos to learn the feature representations through offline training [21, 42]. For example, the MDNet tracker [42] pre-trains a deep model via a number of video sequences from similar tracking benchmarks and fine-tunes the learned model online using the target benchmark sequence. It sets the best performance on the OTB benchmark. However, its offline pre-training on similar tracking benchmarks takes too much time and prone to over-fitting. The GOTURN tracker [21] only employs an offline pre-trained model to perform online tracking. It thus learns the appearance adaptation only from a limited adjacent frames rather than a reasonable interpretation on time series of a video. In the design of our TSN model, we do not perform offline training from similar tracking benchmarks but instead learn and fine-tune the model online from sampled historical targets. Our feature extraction network is directly picked from an off-the-shelf deep models (*e.g.*, VGG model [49] trained on ImageNet [47]) without further pre-training on similar tracking benchmarks.

To learn a complex inter-frame and intra-frame relationship, the PROST method [48] employs optical flow based

mean-shift tracker to adapt the object appearance changes, and the RTT method [7] models long-range contextual cue to identify and exploit reliable spatial-temporal object parts. In our TSN model, with the off-the-shelf feature extraction network, we use the historical object samples to learn a temporal sub-network to capture a temporal correspondence of the target, and a spatial sub-network is proposed to further refine the object localization.

In particular, our TSN model employs only a few convolutional layers from an existing deep model as the feature extraction network to produce general visual features for tracking. It is a small, compact model, and not fine-tuned online, thus can be computed efficiently. With the features extracted from the feature extraction network, the temporal network is designed to learn the temporal correspondence of the target to exploit a temporal relationship of targets from historical frames. To adapt to the target appearance variations through time, the temporal network is fine-tuned online. Afterwards, the spatial network is built to further exploit the spatial relationship of the target and refine the target localization. The proposed TSN model is facilitated by the temporal information interpreted from a global perspective and the spatial information portrayed from a local viewpoint. To summarize, the main contributions of the paper are listed as follows.

1) We present a novel deep architecture for the visual object tracking problems, which incorporates global and local, temporal and spatial information of the object to produce high performance tracking results.

2) We propose a temporal network to incorporate temporal information across frames using an online tuple learning process, which suppresses the drifting problem during the tracking procedure.

3) We develop a coarse-to-fine method to track the object using a spatial deep network which further refines the localization of the target.

Based on these technical contributions, we have developed a deep learning based tracking algorithm that obtains a state-of-the-art performance on four large benchmarks, VOT2014 [14], VOT2016 [15], OTB50 [57], and OTB100 [58].

2. Related Work

We in this section discuss the tracking methods that are most related to our work, and refer the readers to a thorough review on object tracking survey [33, 59] and benchmark evaluations [50, 58]. Generally, conventional object tracking methods can be roughly categorized into generative (*e.g.* mean-shift tracker [31], sparse representation based tracker [25, 40, 63], subspace learning [46], density estimation [6], template matching [39], correlation filter [43], *etc.*) and discriminative methods (such as online boosting tracker [1, 19], multi-instance learning [2], structure SVMs [20], random forest [12], *etc.*). Recently deep learning based tracking methods are developed and have demonstrated very competing performances [21, 37, 42, 54] against conventional tracking methods.

Deep learning based tracking methods. Deep learning models, especially CNN models, have been widely applied to the visual tracking problem, and most of which employ a pre-trained neural network model as the feature extractor [3, 7, 21, 23, 26, 32, 37, 42, 45, 54, 55]. Some CNN based tracking methods combine a CNN model with conventional tracking techniques such as the saliency map and SVM used in the DSCNN tracker [23], correlation filters employed in the HCFT tracker [37], particle filters utilized in the MD-Net tracker [42], feature selection described in the FCNT tracker [54], *etc.* Other CNN based methods either combine several CNN models (*e.g.* the DeepTrack model [32]) or establish an end-to-end CNN model, such as the GO-TURN tracker [21] with no online training but offline learn a generic relationship between object motion and appearance from large number of videos. Apart from CNN models, other deep models, such as Siamese network [3, 5, 51] and Recurrent Neural Networks (RNNs), are also employed in the tracking problem. For instance, the SIAM tracker [3] learns a similarity metric offline by a Siamese network, RD-M [5] proposed a template selection strategy based on a Siamese network, and the RTT tracker [7] describes a multi-directional RNN to capture long-range contextual cues.

Spatial-temporal model based tracking methods. Spatial and temporal information has been utilized to assist object tracking in many works. The CLRST tracker [62] exploits a temporal consistency of particles to constrain the candidate particles and prune irrelevant ones. The DeepTrack model [32] represents temporal adaptation through the update of CNNs in the CNN pool. The STT tracker [56] constructs temporal and spatial appearance context models to prevent drifting. DAVT [16] presents an approach of discriminative spatial attention that identifies some special regions on the target. The RTT tracker [7] uses long-range contextual cues which is also a temporal information in tracking. A Context Tracker [12] is proposed to express the regions with similar appearance as the target and local key-points around the object with some motion correlation to the target in a short time span, which is also a utilization of spatial and temporal information. The R-CNN [18], FCNT [54], and HCFT [37] combine features from different layers of CNN to describe the spatial information of the target. CCOT [11] proposes a formulation for training continuous convolution filters to integrate multi-resolution deep feature maps. Both CCOT and our tracker utilize spatial information from multiple feature maps. Unlike CCOT, our spatial net is built upon a temporal net to work as a refinement network. More distinctively, our spatial net proceeds in a fully convolutional network way, while CCOT proceeds in a correlation filter layer way.
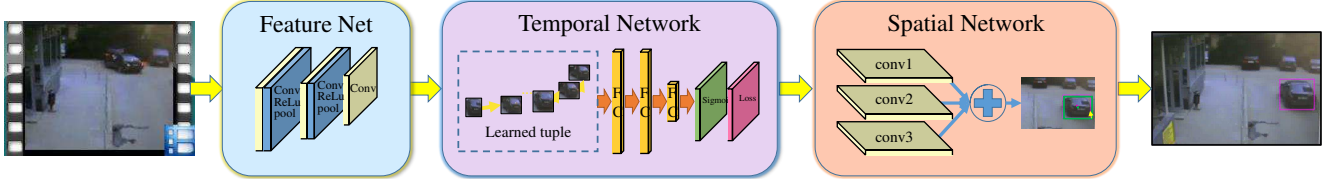
Figure 1. Pipeline of the proposed tracking approach. The Temporal Net is fed with the output of Feature Net borrowed from the VGG Net. Tuple learning in the Temporal Net generates a tuple consisted by key samples learnt from the target pool. The Temporal Net predicts similarities between current proposals and the tuple. The Spatial Net is provided by a hierarchical combination of features from three layers (conv1, conv2, and conv3) at the positions estimated by the Temporal Net. It is utilized to further refine the localization of the target.

3. Proposed Tracking Method

The pipeline of the proposed tracking algorithm is visualized in Fig. 1. The algorithm consists of three main parts, the Feature Net, the Temporal Net, and the Spatial Net.

- **Feature Net.** The designation of the Feature Net is to provide a general feature representation of the target. It contains three convolutional layers, which can be directly borrowed from off-the-shelf deep models.

- **Temporal Net.** The input of the Temporal Net is feature representations from the Feature Net and the tuple learning module functions as a reservoir to collect key historical samples. The output of the Temporal Net is the similarity between the current candidate regions and historical samples.

- **Spatial Net.** The Spatial Net is designed to exploit the spatial and local information of a deep model and yields a confidence map in the vicinity of a candidate predicted by the Temporal Net. It acts as a refinery of the target location.

In the following, we first briefly introduce the Feature Net, then we elaborate the Temporal Net and Spatial Net, which are our main contributions.

3.1. Feature Net

Feature Net is the backbone network of the proposed deep architecture to extract low-level features and does not require off-line training or online fine-tuning. Moreover, it can be tailored from any off-the-shelf deep models. Therefore, representations by the Feature Net can be computed rapidly. In our current implementation, the Feature Net is directly copied from the first three layers of the VGG net [49] trained on the ImageNet with 1000 classes [47], which is a general architecture to extract features and has been widely employed in many domains [17, 36, 42]. We employ it to model the target appearance and it will not be fine-tuned or re-trained in our tracking framework.

3.2. Temporal Net

As the input of an object tracking problem is a sequence of frames, temporal information is thus very important for predicting an accurate location of the target. We thus leverage temporal information via the Temporal Net in our deep network for robust tracking. We describe our Temporal Net

in two parts where tuple learning is proposed to collect key historical temporal samples described in Sec. 3.2.1 and details of the temporal network are delineated in Sec. 3.2.2.
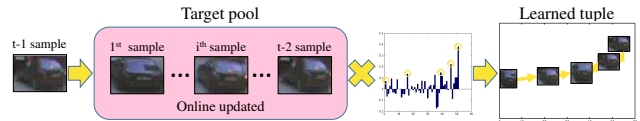

Figure 2. Illustration of the tuple learning procedure. A tuple for an incoming frame exploits temporal information of processed frames and is learnt by a sparse coding of the previous target on the target pool. The target pool is online updated through historical targets every frame.

3.2.1 Tuple Learning

The tuple learning module builds a target pool from all the historical targets with incremental updating and outputs a tuple expressed by key samples from this target pool. It is updated every frame and when the length of the pool exceeds a predefined length we update the target pool by removing the least confident sample and adding the newly tracked target. It is used to maintain the target template set to leverage temporal information, and its learning result is the key targets in historical frames optimized by L_1-induced dictionary learning and spare coding. The input of the fully connected layers of Temporal Net is the key targets from tuple learning and proposals in the current frame. An overview of tuple learning is presented in Fig. 2, where the estimated target found in the previous frame is sparsely represented by the target pool (Eqn. (1)) as a tuple to provide the temporal and global information. A tuple consists of several historical targets that have an akin correlation with the current target. Suppose after the tth frame is tracked, the target pool is denoted as $\mathbf{D}_t = \{\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_t\}$ where $\mathbf{p}_t$ is the target on the tth frame, a tuple is obtained by solving the following optimization problem:

$$\mathbf{c}^* = \underset{\mathbf{c}}{\operatorname{argmin}} \frac{1}{2} \|\mathbf{p}_t - \mathbf{D}_{t-1}\mathbf{c}\|_2^2 + \lambda \|\mathbf{c}\|_1. \qquad (1)$$

Here $\mathbf{c}^*$ is encouraged to be sparse and thus selects only a small set of samples from the pool, which encodes main modes of historical targets. This formulation is used to select temporal related target models for the next frame to provide the global temporal information for the target to track.

3.2.2 Network Details

The Temporal Net receives an output from the Feature Net to encode target appearance and the fine-tuned Temporal network consists of tuple learning, three fully connected layers, a Sigmoid layer and a loss layer. Provided a video sequence, the full network is trained on the first frame and we fine-tune the Temporal Net in the following frames. The intuition behind the Temporal Net is that we consider object tracking as a problem that searches the most similar region from the current frame to the target in the historical frames. Targets from several historical frames provide more information and perform better in resisting drifts than the situation of only using the single previous frame.

We regard the deep model as a metric learning problem, and the output of tuple learning feeds the fully connected layers with historical targets, which is combined with current proposals and outputs the similarity between them. The scale estimation is handled through the proposals sampling with scale variations. In the learning stage, a tuple of regions are extracted on the first frame around the target and we define the ground truth label of the tuple as the overlap rate of the latest two regions. The overlap could represent a similarity since the regions of the pair in the learning stage are from the same frame. In the tracking process, a similarity between the current proposals and a tuple that represents key samples in the historical frames is computed by the temporal network. The tuple is consisted by main modes optimized from a target pool that is online updated by the tuple learning module.

As we are more interested in the global representation in the Temporal Net, the temporal tuple of Feature Net representations is designed to feed through three fully connected layers and between each fully-connected layer we use dropout and ReLU non-linearities. The first two fully connected layers are designed to contain 512 nodes while the output of the last fully connected layer is 1 unit that represents the similarity. Multiple non-linear layers encode a more complex input-output relationship, which in our tracker characterizes the relationship between the temporal tuple of regions and the similarity of these regions. The output similarity is processed by a Sigmoid layer and the loss function in the learning stage employed is L_1 loss (an ablative analysis is conducted in Section 4.2) as shown in the following:

$$L(\mathbf{s}) = \sum_{i=1}^{E} \|s_i - r_i\|, \qquad (2)$$

where E is the number of training examples and r_i is the ground truth label, and typically using a form of stochastic gradient descent (SGD) and back-propagation to train models. We fine-tune the Temporal Net every ten frames and the update data are generated from the estimated locations in the historical frames.

3.3. Spatial Net

The temporal and global information is exploited in our Temporal Net while the spatial and local information is also very important, particularly for a deep network based object tracking, since spatial information is attenuated as a deep network goes deeper [23]. To address this problem, here we propose a spatial network, which takes the feature maps of lower layer rather than higher fully connected layer as input to directly predict a confidence map $\mathbf{M}$.

The proposed Spatial Net is consisted by three convolutional layers with ReLU, but without pooling layers installed since we are more interested in estimating a response of an input window. We employ relatively local information in the spatial network to capture the specific details of the target. Fig. 3 presents our spatial network.
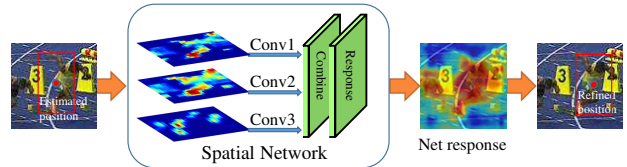

Figure 3. Spatial network. Given an input window, we extract deep CNN features and employ a combination of features on the conv1, conv2 and conv3 layer to learn a spatial response. The spatial network refines the estimated position (estimated by the Temporal Net) on the input window.

In the training process of Spatial Net, the training examples are extracted from the first frame with different overlaps with the labeled target and we employ soft labels generated by a Gaussian function centered at the target center with variance proportional to the target size. Multi-layer deep features are fused together in the spatial network to produce more accurate feature representations.

Denote l as the index of the current layer, the input-output function of the proposed Spatial Net is formulated as follows:

$$f_l(\mathbf{x}) = (\mathbf{b}_l + \mathbf{w}_l * f_{l-1}(\mathbf{x}))^+ \quad l = 1, 2, \qquad (3)$$

$$f_l(\mathbf{x}) = \text{sigmoid}(\mathbf{b}_l + \mathbf{w}_l * f_{l-1}(\mathbf{x})) \quad l = 3. \qquad (4)$$

Here $\mathbf{w}_l$ and $\mathbf{b}_l$ denote the filter and bias for the lth layer, and $^+$ suggests a rectified linear unit. For $l = 1$, $\mathbf{x} = \{\mathbf{X}^1, \ldots, \mathbf{X}^K\}$ where $\mathbf{X}^k$ are features extracted from the kth convolutional layer and the hierarchical combination of these features consist the input feature of the Spatial Net.

Let y_i denote the ground truth label of the ith training sample, and it is cropped from the Gaussian response map of the whole image. The L_2 weight decay and the square loss function $L(\hat{y}, y) = \|\hat{y} - y\|^2$ are combined to give a training criterion for our Spatial Net as following,

$$C(\boldsymbol{\theta}) = \frac{1}{n} \sum_{i=1}^{n} \|\mathbf{y}_i - F_i(\mathbf{x}_i; \boldsymbol{\theta})\|_2^2 + \lambda \|\boldsymbol{\theta}\|_2, \qquad (5)$$

where $(\mathbf{x}_i, \mathbf{y}_i)$ is the ith training example, and $\boldsymbol{\theta} = (\boldsymbol{\omega}, \mathbf{b})$ is a flattened version of all parameters in the network, and F indicates the overall spatial network whose input is the combined feature and output is the response. The parameter λ in Eqn. (5) controls the importance between the reconstruction term and regularization term. It is set to 10^{-4} experimentally in all our evaluations. The training procedure for the spatial network is stochastic gradient descent, which iteratively updates $\boldsymbol{\theta}$ according to Eqn. (6):

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \zeta(\nabla_{\boldsymbol{\theta}} L(F_{\boldsymbol{\theta}}(\mathbf{x}^i), \mathbf{y}^i)), \qquad (6)$$

where ζ is a learning rate and is set to 0.01 experimentally. $\mathbf{x}^i$ is the ith training example (features generated by the Feature Net), $\mathbf{y}^i$ is the soft labels generated by a Gaussian function, and L is the loss function.

Given a feature of a target candidate T_p, a confidence map $\mathbf{M}$ is responded by the Spatial Net. Let $P(x_k, y_k, w_k, h_k)$ describe an after-refined bounding box and then the after-refined target proposal is obtained by Eqn. (7).

$$P(x_k, y_k, w_k, h_k) = \underset{p}{\operatorname{argmax}}(\mathbf{M}(T_p)). \qquad (7)$$

The spatial network ensembles these after-refined candidate regions (Eqn. 8) and B is the candidate number and the importance π_k of a candidate is revealed by the confidences calculated by the Temporal Net and the Spatial Net.

$$P(x^*, y^*, w^*, h^*) = \sum_{k=1}^{B} \pi_k \cdot P(x_k, y_k, w_k, h_k). \qquad (8)$$

3.4. Tracking Algorithm

The main steps of the proposed algorithm are summarized in Algorithm 1. The overall training process takes several seconds. When a new frame flows in, we first sample target candidates (256 proposals in our experiment), and then Feature Net is forwarded and generates feature maps of target proposals. Tuple learning is conducted to determine a tuple originated from the target pool that is online updated by the most accurate samples collected from processed frames. The target of the first frame participates throughout the tracking process to suppress against drifts.

4. Experiments

We conduct extensive experiments to analyze and evaluate the proposed tracking method. In the following, we first introduce some experimental settings. Then we conduct some ablative experiments to study the two key components of our TSN model. Afterwards, we evaluate our TSN model on two of the largest benchmarks: object tracking benchmark (OTB) [57, 58] and visual object tracking (VOT) benchmark [14, 15]. Finally, we perform failure case analyses of our model.

Algorithm 1 Proposed TSN tracking algorithm
___
Require: A video sequence, target positions $P(x^1, y^1, w^1, h^1)$ in the first frame.
Ensure: Target positions in the following frames $P(x^t, y^t, w^t, h^t), t = 2, 3, \ldots$
1: For the initial frame, crop 5000 samples and calculate the corresponding label (overlap); training the proposed Temporal Net and the Spatial Net.
2: **for** $t = 2, 3, \ldots$ **do**
3:     a) Extract candidates from the incoming frame and learn a tuple for the current frame.
4:     b) Forward Temporal Net to calculate similarities between current proposals and the tuple, and then predict target candidates $P(x_k^t, y_k^t, w_k^t, h_k^t), k = 1, 2, \ldots, B$.
5:     c) Feed the target candidates to the Spatial Net and output responses.
6:     d) Find the refined target position $P(x^t, y^t, w^t, h^t)$ by Eqn. (7).
7:     e) Fine-tune Temporal Net and update the Spatial Net if necessary.
8:     f) Update target pool.
9: **end for**
10: **return** $P(x^t, y^t, w^t, h^t)$
___

4.1. Experimental Settings

We implement our TSN model in MATLAB using the MatConvNet toolbox [52] and the full model runs at around 1fps (including the training time of the Temporal Net and Spatial Net on the first frame), with an NVIDIA GTX 980ti GPU. The efficiency of our algorithm lies on two aspects. For one thing, the feature net we employed is shared with the other two nets, which is very efficient and does not introduce many extra computations. For another, since the proposed framework is coarse-to-fine, it does not require large number of proposals in the coarse process (Temporal Net), which saves a lot of computations. Our Feature Net is tailored from the off-the-shelf VGG model as used by many deep network based tracking methods [37,42,45]. But our model does not require pre-training on tracking datasets. The specific layer parameters of Feature Net, Temporal Net and Spatial Net are listed in Table 1. Training samples for the Temporal Net are extracted from the first frame with shifting and scaling versions around the target and 5000 samples are used in our experiment. The candidate number B in the Spatial Net is set to 5 in all experiments. For the learning of both Temporal Net and Spatial Net, we use SGD started with a learning rate of 10^{-4} with a momentum 0.9. The Temporal Net is fine-tuned every ten frames and spatial net is updated every frame. For the target pool, it is online updated through historical targets every frame. To measure

Table 1. Network specifics of the proposed TSN model.

	Feature Net			Temporal Net			Spatial Net		
Layer	Conv-F1	Conv-F2	Conv-F3	FC-T1	FC-T2	FC-T3	Conv-S1	Conv-S2	Conv-S3
Filter-Stride	7-2	5-2	3-1 -	-	-	-	7-2	5-2	3-1
Channels	96	256	512	512	512	512	96	256	512
Activation	ReLU	ReLU	ReLU	ReLU	ReLU	Sigmoid	ReLU	ReLU	ReLU
Norm	√	√	√	-	-	-	√	√	√
Dropout	-	-	-	√	√	-	-	-	-
Pool	3*3	3*3	-	-	-	-	-	-	-

the tracking performance, we adopt the standard protocols in the benchmarks including OTB50 [57], OTB100 [58], VOT2014 [14] and VOT2016 [15].

4.2. Ablative Analyses

To verify the effectiveness of our design for the Temporal Net and Spatial Net in our TSN model, we report the influences of a different loss function of Temporal Net on the overall tracking algorithm and also show how much the Temporal Net and Spatial Net contributes to the overall TSN tracker. We adopt the VOT2014 benchmark as the test dataset and compare four models including TSN tracker with L_1 loss function (our TSN tracker), TSN tracker with L_2 loss function, TSN tracker without Temporal Net and TSN tracker without Spatial Net measured by Expected Average Overlap (EAO), Robustness (R), and Accuracy (A) as shown in Fig. 4. The accuracy of both loss functions are similar and L_1 loss performs slightly better than that of L_2 loss function. While in terms of EAO score, TSN performs better than TSN_L_2. Because the L_2 loss penalizes outputs that are correct but not close compared with the L_1 loss. Fig. 4 also presents the assistance of the Temporal and Spatial Net in our TSN tracker. As shown, all metrics are getting worse (especially EAO) without using Temporal Net and robustness is largely boosted by adding the Spatial Net, which demonstrates the effectiveness of the Temporal Net and the Spatial Net.
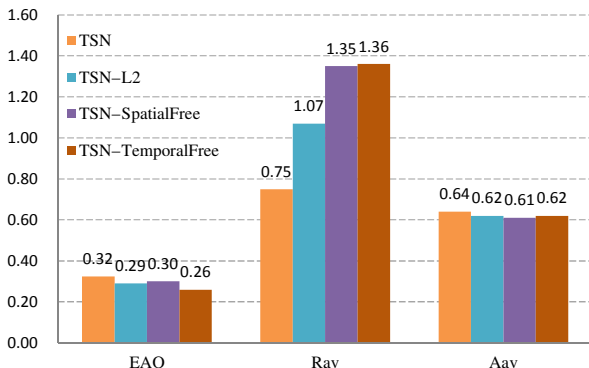

Figure 4. Ablative experiments on the VOT2014 benchmark.

4.3. Evaluations on VOT2014 and VOT2016

In our experiments we use the latest version of the Visual Object Tracking (VOT) toolkit (vot2016), which applies a reset-based methodology. Whenever a failure (zero
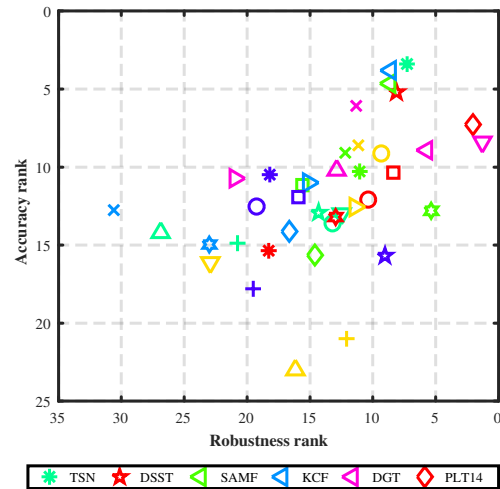

Figure 5. The accuracy-robustness ranking plots of our baseline experiments on the VOT2014 benchmark.

overlap with the ground truth) is detected and the tracker is re-initialized five frames after the failure. Trackers are evaluated by A (average overlap), R (total number of fails) and EAO. We report the comparisons with a number of the latest state-of-the-art trackers on both VOT2014 and VOT2016.

VOT2014 results. VOT2014 is incorporated by 25 video sequences and it provides a ranking analysis based on both statistical and practical significance of the performance gap between trackers (see more details in [14]). Fig. 5 shows the Accuracy-Robustness (AR) plot evaluated on VOT2014, where we compare our tracker with 38 trackers that submitted to the VOT2014 challenge and the best trackers are closer to the top-right corner. We legend the top 6 trackers in Fig. 5 including TSN, DSST [8], SAMF, KCF, DGT, and PLT14. The proposed tracker achieves the rightmost position in the AR plot, which indicates our TSN tracker performs best against the state-of-the-art trackers evaluated by the AR metric.

VOT2016 results. VOT2016 is incorporated by 60 video sequences, on which the proposed tracker is compared with 70 trackers submitted to the VOT2016 challenge. Note that top trackers in the challenges are the latest tracking models. EAO, A and R raw values and ranks are listed in Table 2. Our tracker is superior to other state-of-the-art trackers

Table 2. EAO, A and R raw values (A,R) and ranks $(A_{rank},$ $A_{rank})$ evaluated on the VOT2016 benchmark. The two best results are colored in **red** and **blue**, respectively.

| Tracker	EAO	A	R	A_{rank}	R_{rank}
Proposed	**0.336**	**0.582**	0.266	**1.000**	**1.000**
CCOT	**0.331**	0.539	**0.238**	13.000	**1.000**
TCNN	0.325	0.554	0.268	5.000	**2.000**
SSAT	0.321	**0.577**	0.291	**2.000**	3.000
MLDF	0.311	0.490	**0.233**	38.000	**1.000**
Staple	0.295	0.544	0.378	9.000	11.000
DDC	0.293	0.541	0.345	11.000	7.000
EBT	0.291	0.465	0.252	45.000	**1.000**
SRBT	0.290	0.496	0.350	33.000	11.000
STAPLEp	0.286	0.557	0.368	3.000	10.000
DNT	0.278	0.515	0.329	24.000	5.000
SSKCF	0.277	0.547	0.373	8.000	11.000
SiamRN	0.277	0.549	0.382	6.000	11.000
DeepSRDCF	0.276	0.528	0.326	18.000	6.000
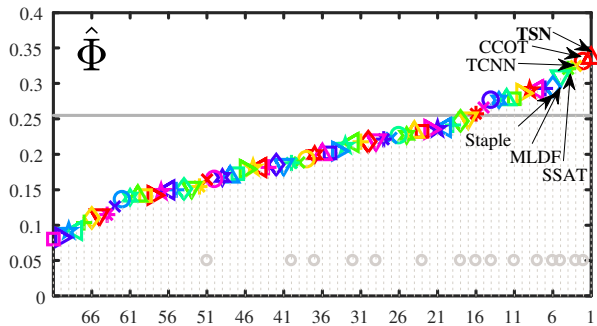
Figure 6. Expected average overlap graph with trackers ranked from right to left. The right-most tracker is the top-performing according to the VOT2016 expected average overlap values.

in the challenging VOT2016 benchmark. Under the VOT strict ranking protocol, the proposed TSN tracker is ranked number one in accuracy, meaning the overlap was clearly higher than other trackers. The second-best ranked tracker in accuracy is SSAT. In terms of robustness, our TSN tracker shares the first place with MLDF, CCOT [11], and EBT, and the second rank in robustness is occupied by TCNN [41]. AR analysis indicates high accuracy and rare failures and these results demonstrate the expressiveness of the proposed temporal and spatial network. Expected average overlap graph evaluated on VOT2016 is reported in Fig. 6. It is clear that our tracker achieves the highest EAO, and is followed by CCOT and TCNN.

4.4. Evaluations on OTB50 and OTB100

OTB [58] is a widely used public dataset in performance evaluation of tracking algorithms. The extended version contains 100 challenging targets with 11 annotated attributes including 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), out-of-view (OV), background clutters (BC), and low resolution (LR). We evaluate the pro-
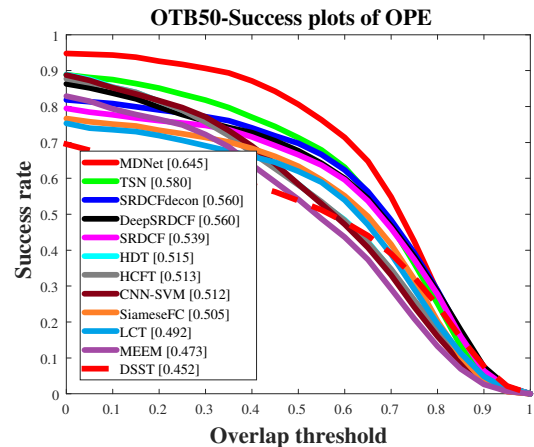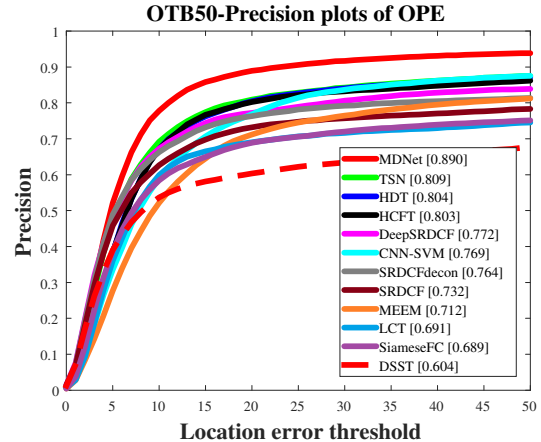

Figure 7. Precision and success plots on the OTB50 benchmark.

posed algorithms with comparisons to a number of state-of-the-art and recently released trackers including MDNet [42], MEEM [60], CNN-SVM [23], HCFT [37], HDT [45], SRDCFdecon [9], DeepSRDCF [10], SRDCF [9], LCT [38], SIAM [3], and DSST [8]. Note that MDNet, Deep-SRDCF, CNN-SVM, HCFT, HDT, and SIAM are latest C-NN based trackers, which provides a baseline for tracking methods that employ deep learning. We initialize all the trackers with the ground-truth object state in the first frame (HDT required ground-truth of the first 5 frames) and report the average precision and success rate of all results. Fig. 7 and Fig. 8 presents the precision and success plot on OTB50 and OTB100, respectively.

The results show that the proposed TSN tracker performs a second best among these trackers. However, on the overfitting problem, the feature net we employed is trained on a very large object detection dataset (ImageNet), and it has a better generalization capacity and is less to be overfitting on object tracking datasets. The proposed method ranked top one or two on the four tracking benchmarks, which suggests the proposed method did have some generalizations from sides. In contrast, the best tracker MDNet employs different training videos for different test videos, which is

Table 3. Average results evaluated on OTB100. The two best results are colored in **red** and **blue**, respectively.

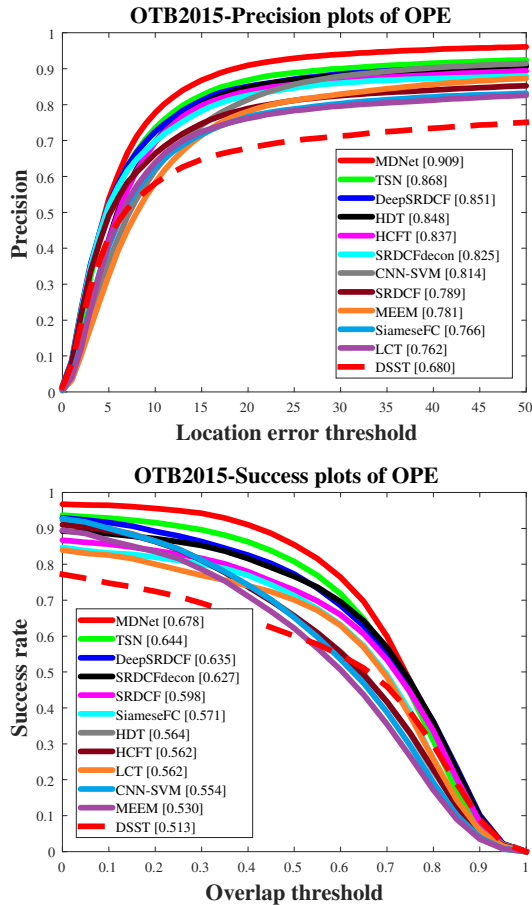| Tracker	TSN	MDNet	CNN-SVM	SIAM	SRDCFdecon	DeepSRDCF	SRDCF	HDT	HCFT	LCT	MEEM	DSST
AP	**0.868**	**0.909**	0.814	0.766	0.825	0.851	0.789	0.848	0.837	0.762	0.781	0.680
DP	**0.808**	**0.854**	0.651	0.713	0.766	0.772	0.728	0.657	0.655	0.701	0.622	0.601
CLE	**12.984**	**11.818**	21.762	38.019	31.518	21.348	38.532	20.096	22.772	67.116	27.714	50.340

Figure 8. Precision and success plots on the OTB100 benchmark.

more inclined to be overfitting, and our tracker does not employ any tracking benchmark offline. DeepSRDCF obtains a third best performance on OTB100. Table 3 summarizes the results where we use three metrics for evaluation. MDNet and TSN achieve the first and second best AP, DP, and CLE. DeepSRDCF achieves a third best AP and DP, and HDT obtains a third best of CLE. DeepSRDCF employs CNN features in a correlation filter but could not exert the best performance of CNN features due to insufficient learning. In our situation, we consider both the temporal and spatial information which leads to a better performance. We also report the results on OTB50 which is consisted by more challenging sequences and our TSN tracker holds the best compared with trackers not employing a pre-training of tracking benchmarks. Overall, the proposed method outperforms other state-of-the-art trackers with no pre-training on tracking benchmarks in all three metrics.

4.5. Failure Cases

The proposed tracker may fail to find tight bounding boxes when the size of the target varies a lot and / or due to severe occlusions (see Fig. 9 for two examples). Since the proposals in the current frame are generated based on the tracking result in the previous frame, if the target changes its size a lot and / or undergoes severe occlusions, the proposals may not capture the variations.
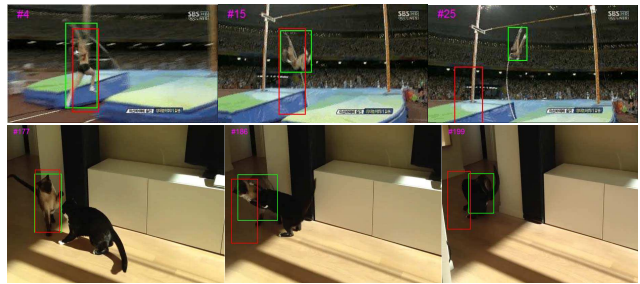

Figure 9. Failure cases of the proposed tracker due to large scale changes (top) and severe occlusions (bottom). Red boxes show our results and the green ones are ground truth.

5. Conclusions

We have proposed a new deep architecture for the visual tracking problem containing a Temporal Net and a Spatial Net. The Temporal Net exploits temporal correspondences between historical frames and the current frame. The Spatial Net employs spatial and local information of the target to refine the localization of the target. We show that the combination of the Temporal Net and the Spatial Net is effective in object tracking and it is evaluated on four benchmarks including OTB50, OTB100, VOT2014, and VOT2016. The experiments demonstrate the effectiveness of the proposed algorithm.

6. Acknowledgments

References

[1] S. Avidan. Ensemble tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(2):261–271, 2007.

[2] B. Babenko, M.-H. Yang, and S. Belongie. Visual tracking with online multiple instance learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 983–990, 2009.

[3] L. Bertinetto, J. Valmadre, J. F. Henriques, A. Vedaldi, and P. H. Torr. Fully-convolutional siamese networks for object tracking. In *European Conference on Computer Vision*, pages 850–865, 2016.

[4] K. Cannons. A review of visual tracking, 2008. Technical Report CSE-2008-07, York University, Canada.

[5] J. K. Choi, Janghoon and K. M. Lee. Visual tracking by reinforced decision making. In *arXiv preprint arXiv:1702.06291*, 2017.

[6] D. Comaniciu, V. Ramesh, and P. Meer. Kernel-based object tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(5):564–577, 2003.

[7] Z. Cui, S. Xiao, J. Feng, and S. Yan. Recurrently target-attending tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1449–1458, 2016.

[8] M. Danelljan, G. Häger, F. Khan, and M. Felsberg. Accurate scale estimation for robust visual tracking. In *British Machine Vision Conference*, 2014.

[9] M. Danelljan, G. Häger, F. S. Khan, and M. Felsberg. Adaptive decontamination of the training set: A unified formulation for discriminative visual tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1430–1438, 2016.

[10] M. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg. Convolutional features for correlation filter based visual tracking. In *IEEE International Conference on Computer Vision Workshops*, pages 58–66, 2015.

[11] M. Danelljan, A. Robinson, F. Khan, and M. Felsberg. Beyond correlation filters: Learning continuous convolution operators for visual tracking. In *European Conference on Computer Vision*, 2016.

[12] T. B. Dinh, N. Vo, and G. Medioni. Context tracker: Exploring supporters and distracters in unconstrained environments. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1177–1184, 2011.

[13] A. Emami, F. Dadgostar, A. Bigdeli, and B. Lovell. Role of spatiotemporal oriented energy features for robust visual tracking in video surveillance. In *International Conference on Advanced Video and Signal-Based Surveillance*, pages 349–354, 2012.

[14] M. K. et al. The visual object tracking vot2014 challenge results. In *European Conference on Computer Vision Workshops*, pages 191–217, 2014.

[15] M. K. et al. The visual object tracking vot2016 challenge results. In *European Conference on Computer Vision Workshops*, pages 777–823, 2016.

[16] J. Fan, Y. Wu, and S. Dai. Discriminative spatial attention for robust tracking. In *European Conference on Computer Vision*, pages 480–493, 2010.

[17] R. Girshick. Fast R-CNN. In *IEEE International Conference on Computer Vision*, pages 1440–1448, 2015.

[18] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 580–587, 2014.

[19] H. Grabner, M. Grabner, and H. Bischof. Real-time tracking via on-line boosting. In *British Machine Vision Conference*, pages 6.1–6.10, 2006.

[20] S. Hare, A. Saffari, and P. H. Torr. Struck: Structured output tracking with kernels. In *IEEE International Conference on Computer Vision*, pages 263–270, 2011.

[21] D. Held, S. Thrun, and S. Savarese. Learning to track at 100 fps with deep regression networks. In *European Conference Computer Vision*, pages 749–765, 2016.

[22] J. Henriques, R. Caseiro, P. Martins, and J. Batista. High-speed tracking with kernelized correlation filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(3):583–596, 2015.

[23] S. Hong, T. You, S. Kwak, and B. Han. Online tracking by learning discriminative saliency map with convolutional neural network. In *International Conference on Machine Learning*, pages 1–10, 2015.

[24] X. Jia, H. Lu, and M.-H. Yang. Visual tracking via adaptive structural local sparse appearance model. In *IEEE Conference on Computer vision and pattern recognition*, pages 1822–1829, 2012.

[25] X. Jia, H. Lu, and M.-H. Yang. Visual tracking via adaptive structural local sparse appearance model. In *IEEE Conference on Computer vision and pattern recognition*, pages 1822–1829, 2012.

[26] W. T. Kai Chen. Once for all: a two-flow convolutional neural network for visual tracking. In *arXiv preprint arXiv:1604.07507*, 2016.

[27] H. U. Kim, D. Y. Lee, J. Y. Sim, and C. S. Kim. Sowp: Spatially ordered and weighted patch descriptor for visual tracking. In *International Conference on Computer Vision*, pages 3011–3019, 2015.

[28] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[29] J. Kwon and K. Lee. Visual tracking decomposition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1269–1276, 2010.

[30] J. Kwon and K. M. Lee. Tracking by sampling trackers. In *IEEE Conference on Computer Vision*, pages 1195–1202, 2011.

[31] I. Leichter. Mean shift trackers with cross-bin metrics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(4):695–706, 2012.

[32] H. Li, Y. Li, and F. Porikli. Deeptrack: Learning discriminative feature representations online for robust visual tracking. *IEEE Transactions on Image Processing*, 25(4):1834–1848, 2016.

[33] X. Li, W. Hu, C. Shen, Z. Zhang, A. Dick, and A. V. D. Hengel. A survey of appearance models in visual object tracking.

ACM Transactions on Intelligent Systems and Technology, 4(4):58:1–58:48, 2013.

[34] Y. Li, H. Ai, T. Yamashita, S. Lao, and M. Kawade. Tracking in low frame rate video: A cascade particle filter with discriminative observers of different life spans. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(10):1728–1740, 2008.

[35] L. Liu, J. Xing, H. Ai, and X. Ruan. Hand posture recognition using finger geometric feature. In *IEEE International Conference on Pattern Recognition*, pages 565–568, 2012.

[36] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

[37] C. Ma, J.-B. Huang, X. Yang, and M.-H. Yang. Hierarchical convolutional features for visual tracking. In *IEEE International Conference on Computer Vision*, pages 3074–3082, 2015.

[38] C. Ma, X. Yang, C. Zhang, and M.-H. Yang. Long-term correlation tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 5388–5396, 2015.

[39] I. Matthews, T. Ishikawa, S. Baker, et al. The template update problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(6):810–815, 2004.

[40] X. Mei and H. Ling. Robust visual tracking and vehicle classification via sparse representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(11):2259–2272, 2011.

[41] H. Nam, M. Baek, and B. Han. Modeling and propagating cnns in a tree structure for visual tracking. In *CoRR abs/1608.07242*, 2016.

[42] H. Nam and B. Han. Learning multi-domain convolutional neural networks for visual tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 4293–4302, 2016.

[43] V. Naresh Boddeti, T. Kanade, and B. Vijaya Kumar. Correlation filters for object alignment. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2291–2298, 2013.

[44] P. Pérez, C. Hue, J. Vermaak, and M. Gangnet. Color-based probabilistic tracking. In *European Conference on Computer Vision*, pages 661–675, 2002.

[45] Y. Qi, S. Zhang, L. Qin, H. Yao, Q. Huang, and J. L. M.-H. Yang. Hedged deep tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

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

[47] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[48] J. Santner, C. Leistner, A. Saffari, T. Pock, and H. Bischof. Prost: Parallel robust online simple tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 723–730, 2010.

[49] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

[50] A. W. Smeulders, D. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah. Visual tracking: An experimental survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(7):1442–1468, 2014.

[51] R. Tao, E. Gavves, and A. W. M. Smeulders. Siamese instance search for tracking. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016.

[52] A. Vedaldi and K. Lenc. Matconvnet: Convolutional neural networks for matlab. In *ACM International Conference on Multimedia*, pages 689–692. ACM, 2015.

[53] A. Vu, A. Ramanandan, A. Chen, J. A. Farrell, and M. Barth. Real-time computer vision/dgps-aided inertial navigation system for lane-level vehicle navigation. *IEEE Transactions on Intelligent Transportation Systems*, 13(2):899–913, 2012.

[54] L. Wang, W. Ouyang, X. Wang, and H. Lu. Visual tracking with fully convolutional networks. In *IEEE International Conference on Computer Vision*, pages 3119–3127, 2015.

[55] N. Wang and D.-Y. Yeung. Learning a deep compact image representation for visual tracking. In *Advances in neural information processing systems*, pages 809–817, 2013.

[56] L. Wen, Z. Cai, Z. Lei, D. Yi, and S. Z. Li. Robust online learned spatio-temporal context model for visual tracking. *IEEE Transactions on Image Processing*, 23(2):785–796, 2014.

[57] Y. Wu, J. Lim, and M.-H. Yang. Online object tracking: A benchmark. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2411–2418, 2013.

[58] Y. Wu, J. Lim, and M.-H. Yang. Object tracking benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1834–1848, 2015.

[59] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *ACM Computing Surverys*, 38(4), 2006.

[60] J. Zhang, S. Ma, and S. Sclaroff. Meem: robust tracking via multiple experts using entropy minimization. In *European Conference on Computer Vision*, pages 188–203, 2014.

[61] M. Zhang, J. Xing, J. Gao, X. Shi, Q. Wang, and W. Hu. Joint scale-spatial correlation tracking with adaptive rotation estimation. In *IEEE International Conference on Computer Vision Workshops*, pages 32–40, 2015.

[62] T. Zhang, S. Liu, N. Ahuja, M.-H. Yang, and B. Ghanem. Robust visual tracking via consistent low-rank sparse learning. *International Journal of Computer Vision*, 111(2):171–190, 2015.

[63] W. Zhong, H. Lu, and M.-H. Yang. Robust object tracking via sparse collaborative appearance model. *IEEE Transactions on Image Processing*, 23(5):2356–2368, 2014.