Hierarchical Convolutional Features for Visual Tracking

Chao Ma
SJTU
chaoma@sjtu.edu.cn

Jia-Bin Huang
UIUC
jbhuang1@illinois.edu

Xiaokang Yang
SJTU
xkyang@sjtu.edu.cn

Ming-Hsuan Yang
UC Merced
mhyang@ucmerced.edu

Abstract

Visual object tracking is challenging as target objects often undergo significant appearance changes caused by deformation, abrupt motion, background clutter and occlusion. In this paper, we exploit features extracted from deep convolutional neural networks trained on object recognition datasets to improve tracking accuracy and robustness. The outputs of the last convolutional layers encode the semantic information of targets and such representations are robust to significant appearance variations. However, their spatial resolution is too coarse to precisely localize targets. In contrast, earlier convolutional layers provide more precise localization but are less invariant to appearance changes. We interpret the hierarchies of convolutional layers as a nonlinear counterpart of an image pyramid representation and exploit these multiple levels of abstraction for visual tracking. Specifically, we adaptively learn correlation filters on each convolutional layer to encode the target appearance. We hierarchically infer the maximum response of each layer to locate targets. Extensive experimental results on a large-scale benchmark dataset show that the proposed algorithm performs favorably against state-of-the-art methods.

1. Introduction

Visual object tracking is one of the fundamental problems in computer vision with numerous applications [33, 26]. A typical scenario of visual tracking is to track an unknown target object, specified by a bounding box in the first frame. Despite significant progress in the recent decades, visual tracking is still a challenging problem, mainly due to large appearance changes caused by occlusion, deformation, abrupt motion, illumination variation, and background clutter. Recently, features based on convolutional neural networks (CNNs) have demonstrated state-of-the-art results on a wide range of visual recognition tasks [20, 11]. It is thus of great interest to understand how to best exploit the rich feature hierarchies in CNNs for robust visual tracking.

Existing deep learning based trackers [30, 21, 29, 18] typically draw positive and negative training samples around the estimated target location to incrementally learn a classifier over features extracted from a CNN. Two issues ensue with such approaches. The first issue lies in the use of neural networks as an online classifier following recent object recognition algorithms, where only the outputs of the last layer are used to represent targets. For high-level visual recognition problems, it is effective to use features from the last layer as they are most closely related to category-level semantics and most invariant to nuisance variables such as intra-class variations and precise location. However, the objective of visual tracking is to locate targets precisely rather than to infer their semantic classes. Using only the features from the last layer is thus not the optimal representation for targets. The second issue is concerned with extracting training samples. Training a robust classifier requires a considerably large number of positive and negative samples, which is not available in visual tracking. In addition, there lies ambiguity in determining a decision boundary since positive and negative samples are highly correlated due to sampling near a target.

In this work, we address these two issues by (i) using the features from hierarchical layers of CNNs rather than only the last layer to represent targets; (ii) learning adaptive correlation filters on each CNN layer without the need of sampling. Our approach builds on the observation that although the last layers of CNNs are more effective to capture semantics, they are insufficient for capturing fine-grained spatial details such as object positions. The earlier layers, on the other hand, are precise in localization but do not capture semantics as illustrated in Figure 1. This observation suggests that reasoning with multiple layers of CNN features for visual tracking is of great importance as semantics are robust to significant appearance variations and spatial details are effective for precise localization. We exploit both the hierarchical features from the recent advances in CNNs and the inference approach across multiple levels in classical computer vision problems. For example, computing optical flow from the coarse levels of the image pyramid are efficient, but finer levels are required for obtaining an accurate and detailed flow field. A coarse-to-fine searching strategy is often adopted for best results [23]. In light

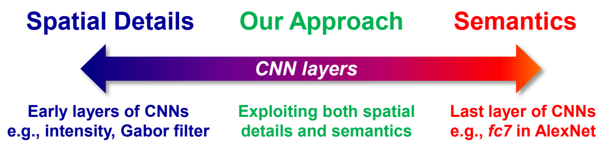


Figure 1. Convolutional layers of a typical CNN model, e.g., AlexNet [20], provide multiple levels of abstraction in the feature hierarchies. The features in the earlier layers retain higher spatial resolution for precise localization with low-level visual information similar to the response map of Gabor filters [4]. On the other hand, features in the latter layers capture more semantic information and less fine-grained spatial details. Our approach exploit the semantic information of last layers to handle large appearance changes and alleviate drifting by using features of earlier layers for precise localization.

of this connection, we learn one adaptive correlation filter [3, 16, 7, 35, 6, 17] over features extracted from each CNN layer and use these multi-level correlation response maps to collaboratively infer the target location. We consider *all* the shifted versions of features as training samples and regress them to a Gaussian function with a small spatial bandwidth, thereby alleviating the sampling ambiguity of training a binary discriminative classifier.

We make the following three contributions. First, we propose to use the rich feature hierarchies of CNNs as target representations for visual tracking, where both semantics and fine-grained details are simultaneously exploited to handle large appearance variations and avoid drifting. Second, we adaptively learn linear correlation filters on each CNN layer to alleviate the sampling ambiguity. We infer the target location using the multi-level correlation response maps in a coarse-to-fine fashion. Third, we carry out extensive experiments on a large-scale benchmark dataset [32] with 100 challenging image sequences and demonstrate that the proposed tracking algorithm performs favorably against existing state-of-the-art methods in terms of accuracy and robustness.

2. Related Work

In this section, we discuss tracking methods closely related to this work. We refer the readers to a comprehensive review on visual tracking in [33, 22, 26].

Tracking by Binary Classifiers. Visual tracking can be posed as a repeated detection problem in a local window (known as tracking-by-detection), where classifiers are often learned online. For each frame, a set of positive and negative training samples are collected for incrementally learning a discriminative classifier to separate a target from its backgrounds. However, the sampling ambiguity problem arises with such approaches that draw samples for learning online classifiers. Slight inaccuracies in labeling samples affect the classifier and gradually cause the trackers to drift. Considerable efforts have been made to alleviate these model update problems caused by sample ambiguity. The core idea of these algorithms lie in how to properly update a discriminative classifier to reduce drifts. Examples include multiple instance learning (MIL) [1], semi-supervised learning [10, 12], and P-N learning [19]. Instead of learning only one single classifier, Zhang et al. [34] combine multiple classifiers with different learning rates. On the other hand, Hare et al. [13] show that the objective of label prediction using a classifier is not explicitly coupled to the objective of tracking (accurate position estimation) and pose tracking as a joint structured output prediction problem. By alleviating the sampling ambiguity problem, these methods perform well in a recent benchmark study [31]. We address the sample ambiguity with correlation filters where training samples are regressed to soft labels of a Gaussian function rather than binary labels for discriminative classifier learning.

Tracking by Correlation Filters. Correlation filters for visual tracking have attracted considerable attention due to its high computational efficiency with the use of fast Fourier transforms. Tracking methods based on correlation filters regress all the circular-shifted versions of input features to a target Gaussian function and thus no hard-thresholded samples of target appearance are needed. Bolme et al. [3] learn a minimum output sum of squared error filter over luminance channel for fast visual tracking. Several extensions have been proposed to considerably improves tracking accuracy, including kernelized correlation filters [16], multi-dimensional features [17, 7], context learning [35] and scale estimation [6]. In this work, we propose to learn correlation filters over multi-dimensional features in a way similar to existing methods [7, 17]. The main differences lie in the use of learned CNN features rather than hand-crafted features (e.g., HOG [5] or color-attributes [7]) and we construct multiple correlation filters on hierarchical convolutional layers as opposed to only one single filter by existing approaches.

Tracking by CNNs. Visual representations are of great importance for object tracking. Numerous hand-crafted features have been used to represent the target appearance such as subspace representation [24] and color histograms [37]. The recent years have witnessed significant advances of CNNs on visual recognition problems. Wang and Yeung [30] propose a deep learning tracker (DLT) using a multi-layer autoencoder network. This network is pretrained on part of the 80M Tiny Image dataset [27] in an unsupervised fashion. On the other hand, Wang et al. [29] propose to learn a two-layer neural network on a video repository [39], where temporally slowness constraints are imposed for feature learning. Li et al. [21] construct multiple CNN classifiers on different instances of target objects to rule out noisy samples during model update. The

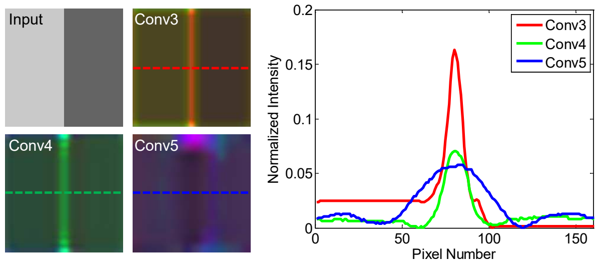


Figure 2. Visualization of the CNN features of an image with a horizontal step edge. The first three principle components of features from three layers at the dash lines are visualized. Note that the *conv5-4* layer is less effective to locate the step edge due to its low spatial resolution while the *conv3-4* layer is more useful for precise localization.

DeepTrack [21] learns two-layer CNN classifiers from binary samples and does not require a pre-training procedure. Hong et al. [18] learns target-specific saliency map using a pre-trained CNN. We note that the aforementioned CNN trackers all rely on positive and negative training samples and only exploit the features from the last layer. In contrast, our approach builds on adaptive correlation filters which regress the dense, circularly shifted samples with soft labels and effectively alleviate sampling ambiguity. In addition, we exploit the features from multiple convolutional layers to encode target appearance. We extract CNN features using the VGG-Net [25], which is trained on the large-scale ImageNet dataset [8] with category-level label. We also note that the DLT [30] and DeepTrack [21] methods update the appearance models by finetuning CNNs online, while Wang et al. [29] and our algorithm use classifier learning for model update.

3. Overview

Our approach builds on the observation that the last layers of CNNs encode semantic abstraction of targets and their outputs are robust to appearance variations. On the other hand, the early layers retain more fine-grained spatial details and thus are useful for precise localization. We show in Figure 2 an image of a horizontal step edge and visualize the CNN features on the third, fourth, and fifth convolutional layers, where the fifth convolutional layer is less effective to locate the sharp boundary due to its low spatial resolution while the third layer is more useful to locate it precisely. Our goal is to exploit the best of both semantics and fine-grained details for visual object tracking. Figure 3 illustrates the main steps of our algorithm: we learn adaptive linear correlation filter over the outputs of each convolutional layer and coarse-to-fine search the multi-level correlation response maps to infer the location of targets.

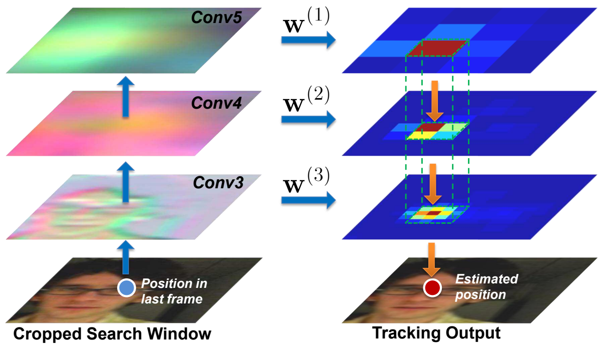


Figure 3. Main steps of the proposed algorithm. Given an image, we first crop the search window centered at the estimated position in the previous frame. We use the third, fourth and fifth convolutional layers as our target representations. Each layer indexed by i is then convolved with the learned linear correlation filter $\mathbf{w}^{(i)}$ to generate a response map, whose location of the maximum value indicates the estimated target position. We search the multi-level response maps to infer the target location in a coarse-to-fine fashion.

4. Proposed Algorithm

In this section, we first present the CNN features used in this work, technical details on learning linear correlation filters, and the coarse-to-fine searching strategy. We introduce the online model update in the end.

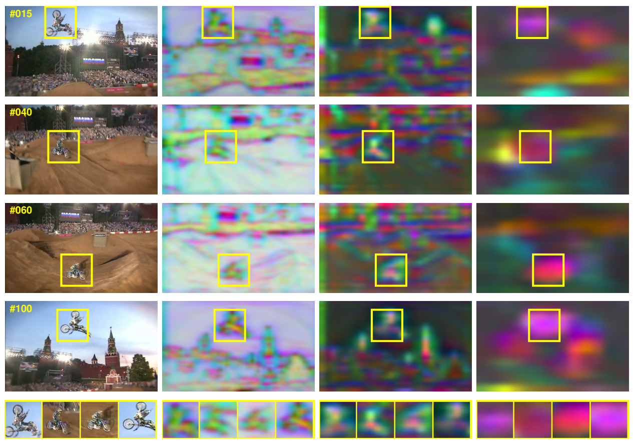
4.1. Convolutional Features

We uses the convolutional feature maps from a CNN, e.g., AlexNet [20] or VGG-Net [25], to encode target appearance. Along with the CNN forward propagation, the semantical discrimination between objects from different categories is strengthened, as well as a gradual reduction of spatial resolution for precise localization (See also Figure 1). For visual object tracking, we are interested in accurate locations of a target object. We thus ignore the fully-connected layers as they show little spatial resolution, i.e., 1×1.

Due to the pooling operators used in the CNNs, spatial resolution is gradually reduced with the increase of the depth of convolutional layers. For example, the convolutional feature maps of *pool5* in the VGG-Net are of spatial size 7×7, which is $\frac{1}{32}$ of the input image size 224×224. Such low spatial resolution is insufficient to locate targets accurately. We alleviate this issue by resizing each feature map to a fixed larger size with bilinear interpolation. Let $\mathbf{h}$ denote the feature map and $\mathbf{x}$ be the upsampled feature map, the feature vector for the i-th location is:

$$\mathbf{x}_i = \sum_k \alpha_{ik} \mathbf{h}_k, \tag{1}$$

where the interpolation weight α_{ik} depends on the position of i and k neighboring feature vectors respectively. Note



(a) Input (b) *conv3-4* (c) *conv4-4* (d) *conv5-4*

Figure 4. Visualization of convolutional layers. (a) four frames from the challenging *MotorRolling* sequence. (b)-(d) features are extracted on convolutional layers *conv3-4*, *conv4-4*, and *conv5-4* using the VGG-Net [25]. The yellow bounding boxes indicate the tracking results by our method. Notice that although the appearance of the target changes significantly, the features using the output of the *conv5-4* convolution layer (d) is able to discriminate it readily even the background has dramatically changed. The *conv4-4* (c) and *conv3-4* (b) layers encode more fine-grained details and are useful to locate target precisely.

that this interpolation takes place in the spatial space and can be seen as an interpolation of location. We visualize the upsampled outputs of the third, fourth, and fifth layers by projecting the features onto their corresponding first three principal components on the *MotorRolling* sequence in Figure 4. As shown in Figure 4(d), features on the fifth layer are effective in discriminating the targets even with dramatic background changes. We note that this insight is also exploited in [14] for segmentation and fine-grained localization using CNNs in which features from multiple layers are concatenated together. However, this feature representation ignores the coarse-to-fine hierarchy in the CNN architecture, and does not work well for visual tracking as shown in our experimental validation (See Section 6).

4.2. Correlation Filters

A typical correlation tracker [3, 16, 7, 35, 6] learns a discriminative classifier and estmate the translation of target objects by searching for the maximum value of correlation response map. In this work, the outputs of each convolutional layer are used as multi-channel features [9, 2, 17]. Denote $\mathbf{x}$ as the l-th layer of feature vector of size $M \times N \times D$, where M, N, and D indicates the width, height, and the number of channels, respectively. Here we denote $\mathbf{x}^{(l)}$ concisely as $\mathbf{x}$ and ignore the dependence of M, N, and D on the layer index l. We consider all the circular shifts of $\mathbf{x}$ along the M and N dimensions as training samples. Each shifted sample $\mathbf{x}_{m,n}$, $(m,n) \in \{0,1,\ldots,M-1\} \times \{0,1,\ldots,N-1\}$, has a Gaussian function label $y(m,n) = e^{-\frac{(m-M/2)^2+(n-N/2)^2}{2\sigma^2}}$, where σ is the kernel width. A correlation filter $\mathbf{w}$ with the same size of $\mathbf{x}$ is then learned by solving the following minimization problem:

$$\mathbf{w}^* = \operatorname*{argmin}_{\mathbf{w}} \sum_{m,n} \|\mathbf{w} \cdot \mathbf{x}_{m,n} - y(m,n)\|^2 + \lambda \|\mathbf{w}\|_2^2, \tag{2}$$

where λ is a regularization parameter ($\lambda \geq 0$) and the inner product is induced by a linear kernel in the Hilbert space, e.g., $\mathbf{w} \cdot \mathbf{x}_{m,n} = \sum_{d=1}^{D} \mathbf{w}_{m,n,d}^{\top} \mathbf{x}_{m,n,d}$. As the label $y(m,n)$ is soft (not binary), so no hard-thresholded sample is required. Notice that the minimization problem in (2) is akin to training the vector correlation filters in [2], and can be solved in each indivual feature channel using fast Fourier transformation (FFT). Let the capital letters be the corresponding Fourier transformed signals. The learned filter in the frequency domain on the d-th ($d \in \{1,\ldots,D\}$) channel can be written as

$$\mathbf{W}^d = \frac{\mathbf{Y} \odot \bar{\mathbf{X}}^d}{\sum_{i=1}^{D} \mathbf{X}^i \odot \bar{\mathbf{X}}^i + \lambda}. \tag{3}$$

In (3), $\mathbf{Y}$ is the Fourier transformation form of $\mathbf{y} = \{y(m,n) | (m,n) \in \{0,1,\ldots,M-1\} \times \{0,1,\ldots,N-1\}\}$ and the bar means complex conjugation. The operator $\odot$ is the Hadamard (element-wise) product. Given an image patch in the next frame, the feature vector on the l-th layer is denoted by $\mathbf{z}$ and of size $M \times N \times D$. The l-th correlation response map is computed by

$$f_l = \mathscr{F}^{-1}\Big(\sum_{d=1}^{D} \mathbf{W}^d \odot \bar{\mathbf{Z}}^d\Big). \tag{4}$$

The operator $\mathscr{F}^{-1}$ denotes the inverse FFT transform. The target location on the l-th convolution layer can then be estimated by searching for the position of maximum value of the correlation response map f_l of size $M \times N$.

4.3. Coarse-to-Fine Translation Estimation

Given the set of correlation response maps $\{f_l\}$, we hierarchically infer the target translation of each layer, i.e., the location of maximum value in last layer is used as a regularization to search for the maximum value of the earlier layer. Let $(\hat{m},\hat{n}) = \operatorname{argmax}_{m,n} f_l(m,n)$ indicate the location of the maximum values on the l-th layer, the optimal location of target in the $(l-1)$-th layer is formulated as:

$$\begin{aligned} &\operatorname*{argmax}_{m,n} \quad f_{l-1}(m,n) + \gamma f_l(m,n), \\ &\text{s.t.} \quad |m-\hat{m}| + |n-\hat{n}| \leq r. \end{aligned} \tag{5}$$

The constraint indicates that only the $r \times r$ neighboring regions of $(\hat{m},\hat{n})$ are searched in the $(l-1)$-th correlation response map. The response values from the last layers are weighted by a regularization term γ and then back-propagated to the response maps of early layers. The target location is finally estimated by maximizing (5) on the layer with the finest spatial resolution.

4.4. Model Update

An optimal filter on l-th layer can be updated by minimizing the output error over all tracked results so far as described in [2]. However, this involves solving a $D \times D$ linear system of equations per location at (m,n), which is computationally expensive as the channel number is usually large with the CNN features (e.g., $D = 512$ in the *conv5-4* and *conv4-4* layers in the VGG-Net). To obtain a robust approximation, we update the numerator $\mathbf{A}^d$ and denominator $\mathbf{B}^d$ of the correlation filter $\mathbf{W}^d$ in (3) separately using a moving average:

$$\mathbf{A}_t^d = (1-\eta)\mathbf{A}_{t-1}^d + \eta \mathbf{Y} \odot \bar{\mathbf{X}}_t^d; \tag{6a}$$

$$\mathbf{B}_t^d = (1-\eta)\mathbf{B}_{t-1}^d + \eta \sum_{i=1}^{D} \mathbf{X}_t^i \odot \bar{\mathbf{X}}_t^i; \tag{6b}$$

$$\mathbf{W}_t^d = \frac{\mathbf{A}_t^d}{\mathbf{B}_t^d + \lambda}, \tag{6c}$$

where t is the frame index and η is a learning rate.

5. Implementation Details

We present the main steps of the proposed tracking algorithm in Algorithm 1 and the implementation details as follows. We adopt the VGG-Net-19 [25] trained on ImageNet [8] for feature extraction. We first remove the fully-connected layers and use the outputs of the *conv3-4*, *conv4-4* and *conv5-4* convolutional layer as our features. Notice that we do not use the outputs of the pooling layers because we want to retain more spatial resolution on each convolutional layer. Given an image frame with searching window of size $M \times N$ (e.g., 1.8 times of the target size), we set a fixed spatial size of $\frac{M}{4} \times \frac{N}{4}$ to resize the features from each covolutional layer.

The parameters for training correlation filters on each layer are kept the same. We set the regularization parameter of (2) to $\lambda = 10^{-4}$, and use a kernel width of 0.1 for generating the Gaussian function labels. Their learning rate η in (6) is set to 0.01. To remove the boundary discontinuities, the extracted feature channels of each convolutional layer are weighted by a cosine window [3]. We set value of γ as 1, 0.5 and 0.02 for the *conv4-4*, *conv3-4*, *conv5-4* layers, respectively. We observe that the results are not sensitive to the parameter r for the neighborhood search constraint This amounts to simply sum over the weighted response maps from multiple layers to infer the target location.

Algorithm 1: Proposed tracking algorithm.

```
Input  : Initial target position $\mathbf{p}_0$,
Output: Estimated object positition $\mathbf{p}_t = (x_t, y_t)$, and
        learned correlation filters $\{\mathbf{w}_t^l\}$, $l \in \{5,4,3\}$.
1 repeat
2     Crop out the searching window in frame $t$ centered
      at $(x_{t-1}, y_{t-1})$ and extract covolutional features
      with spatial interpolation using (1);
3     foreach layer $l$ do computing confidence score $f_l$
      using $\mathbf{w}_{t-1}^{(l)}$ and (4);
4     Coarse-to-fine estimate the new position $(x_t, y_t)$ on
      response map set $\{f_l\}$ using (5);
5     Crop out new patch centered at $\mathbf{p}_t = (x_t, y_t)$ and
      extract convolutional features with interpolation;
6     foreach layer $l$ do updating correlation filters
      $\{\mathbf{w}_t^l\}$ using (6);
7 until End of video sequences;
```

6. Experimental Validations

We evaluate the proposed method on a large benchmark dataset [32] containing 100 videos with comparisons to state-of-the-art methods. For completeness, we also report the results on the benchmark dataset [31] with 50 videos (a subset of [32]). We quantitatively evaluate trackers using distance precision rate, overlap ratio, and center location error. We follow the protocol in [32] and use same parameter values for all the sequences and all sensitivity analysis. More results can be found in the supplementary material. We implement our tracker in MATLAB on an Intel I7-4770 3.40 GHz CPU with 32 GB RAM, and use the MatConvNet toolbox [28], where the computation of forward propagation on CNNs is transferred to a GeForce GTX Titan GPU. The source code is publicly available on our project page [1].

Quantitative Evaluation. We evaluate the proposed algorithm with comparisons to 12 state-of-the-art trackers. These trackers can be broadly categorized into three classes: (i) deep learning tracker DLT [30] (ii) correlation filter trackers including the CSK [16], STC [35], and KCF [17]; and (iii) representative tracking algorithms using single or multiple online classifiers, including the MIL [1], Struck [13], CT [36], LSHT [15], TLD [19], SCM [38], MEEM [34], and TGPR [10] methods.

Figure 5 shows the results under one-pass evaluation (OPE), temporal robustness evaluation (TRE) and spatial robustness evaluation (SRE) using the distance precision rate and overlap success rate. Additional comparisons about OPE, SRE and TRE on the first 50 sequences can be found in the supplementary materials. Overall, the proposed algo-

[1]`https://sites.google.com/site/chaoma99/iccv15-tracking`

Table 1. Comparisons with state-of-the-art trackers on the first 50 (I) [31] and entire 100 (II) [32] benchmark sequences. Our approach performs favorably against existing methods in distance precision (DP) rate at a threshold of 20 pixels, overlap success (OS) rate at an overlap threshold of 0.5 and center location error (CLE). The first and second best values are highlighted by bold and underline.

		Ours	DLT [30]	KCF [17]	STC [35]	Struck [13]	SCM [38]	CT [36]	LSHT [15]	CSK [16]	MIL [1]	TLD [19]	MEEM [34]	TGPR [10]
DP rate (%)	I	**89.1**	54.8	74.1	54.7	65.6	64.9	40.6	56.1	54.5	47.5	60.8	<u>83.0</u>	70.5
	II	**83.7**	52.6	69.2	50.7	63.5	57.2	35.9	49.7	51.6	43.9	59.2	<u>78.1</u>	64.3
OS rate (%)	I	**74.0**	47.8	62.2	36.5	55.9	61.6	34.1	45.7	44.3	37.3	52.1	<u>69.6</u>	62.8
	II	**65.5**	43.0	54.8	31.4	51.6	51.2	27.8	38.8	41.3	33.1	49.7	<u>62.2</u>	53.5
CLE (pixel)	I	**15.7**	65.2	35.5	80.5	50.6	54.1	78.9	55.7	88.8	62.3	48.1	<u>20.9</u>	51.3
	II	**22.8**	66.5	45.0	86.2	47.1	61.6	80.1	68.2	305	72.1	60.0	<u>27.7</u>	55.5
Speed (FPS)	I	11.0	8.59	245	**687**	10.0	0.37	38.8	39.6	<u>269</u>	28.1	21.7	20.8	0.66
	II	10.4	8.43	243	**653**	9.84	0.36	44.4	39.9	<u>248</u>	28.0	23.3	20.8	0.64

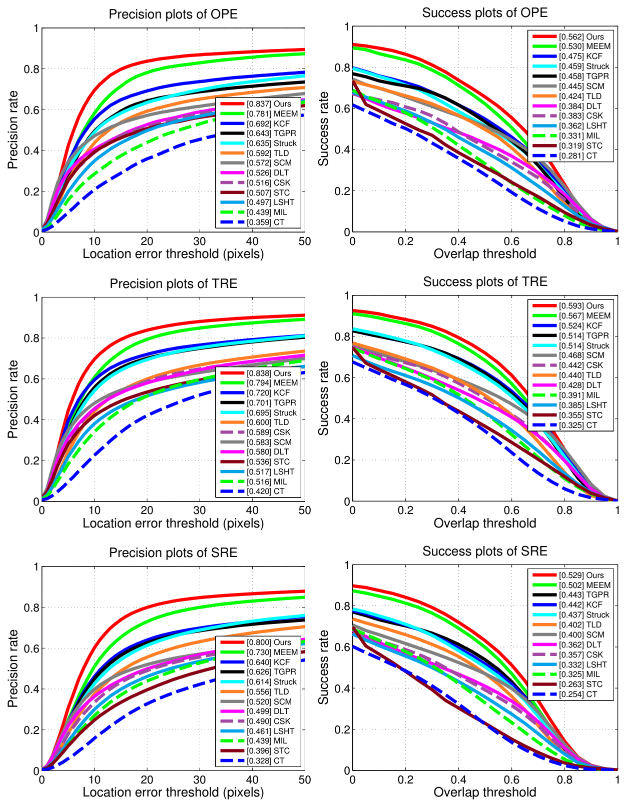


Figure 5. Distance precision and overlap success plots over 100 benchmark sequences using one-pass evaluation (OPE), temporal robustness evaluation (TRE) and spatial robustness evaluation (SRE). The legend of distance precision contains threshold scores at 20 pixels while the legend of overlap success contains area-under-the-curve score for each tracker. Our proposed algorithm performs favorably against the state-of-the-art trackers.

rithm perform favorably against the state-of-the-art methods in all three metrics: OPE, TRE and SRE. We present the quantitative comparisons of distance precision rate at 20 pixels, overlap success rate at 0.5, center location errors, and tracking speed in Table 1. We report both the results on the first 50 sequences (Benchmark I) [31] and all 100 sequences (Benchmark II) [32]. Table 1 shows that our algorithm performs well against state-of-the-art trackers in distance precision (DP) rate, overlap success (OS) rate and center location error (CLE). Notice that with entire 100 sequences, Benchmark II is more challenging where all the compared trackers perform worse than on Benchmark I. Among the state-of-the-art trackers, the MEEM method [34] achieves the second best results. The proposed method achieves lower CLE of 22.8 pixels over 100 video sequences, compared to the second best result from the MEEM tracker with 27.7 pixels. Our tracker runs at around 10 frames per second. The main computational load of our tracker is the forward propagation process to extract features (around 45% of the computing time for each frame).

Attribute-based Evaluation. We further analyze the tracker performance under different video attributes (e.g., background clutter, occlusion, fast motion) annotated in the benchmark [32]. Figure 6 shows the OPE for eight main video attributes. From Figure 6, we have the following observations. First, our method is effective in handling background clutters which can be explained by considering features with semantics and spatial details from the hierarchical layers of CNNs. In contrast, the DLT method pre-trains the network with an unsupervised model and only uses the output of last layer of the trained neural network as features. This suggests CNN features (e.g., VGG-Net) learned with category-level supervision are more effective to discriminate targets from background. Second, our method performs well in the presence of scale variations as the last layer of the pre-trained model retains semantic information insensitive to scale changes. Third, our method does not perform as well in the presence of occlusion and object deformation. This can be attributed to the holistic feature representation used in our model. Re-detection modules or part-based models will be considered in our future work.

Feature Analysis. To analyze the effectiveness of the proposed algorithm, we compare the performance of using different convolutional layers as features on the benchmark

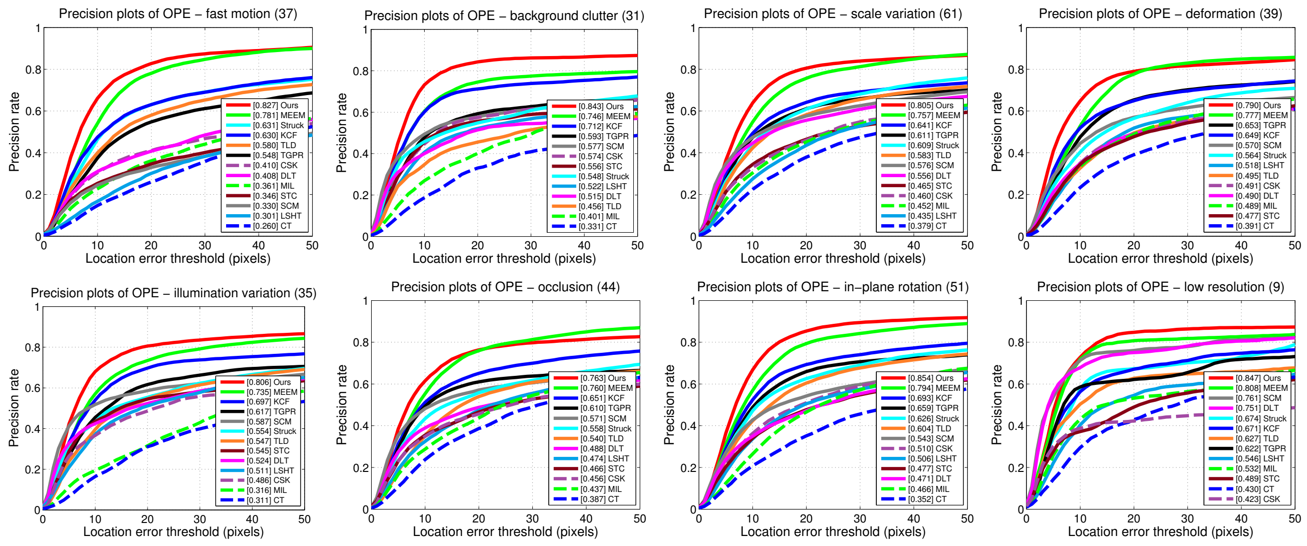


Figure 6. Distance precision plots over eight tracking challenges of fast motion, background clutter, scale variation, deformation, illumination variation, occlusion, in-plane rotation, and low resolution. The legend contains the scores at a threshold of 20 pixels for each tracker. Our proposed algorithm performs favorably against the state-of-the-art trackers with these eight challenging attributes.

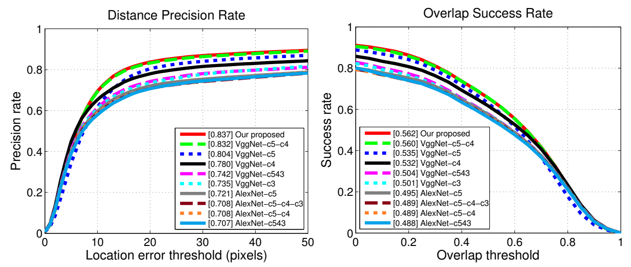


Figure 7. Performance evaluation using different convolutional layers as features. Each single layer (c5,c4 and c3), the combination of the *conv5-4* and *conv4-4* layers (c5-c4), and the concatenation of three layers (c543) are evaluated using both VGG-Net and AlexNet.

with 100 sequences. We first test on each single layer (c5, c4 and c3), and then perform coarse-to-fine search on the fifth and fourth layers (c5-c4). We also concatenate these three layers together (c543) as the hypercolumns used in [14]. However, such concatenation breaks down the hierarchies over CNN layers and thus does not perform well for visual tracking. In addition, we test the features extracted from the AlexNet [20] with a same scheme. Figure 7 shows the top 10 performing methods with different features using OPE, where the values at the legends with DP is based on the threshold of 20 pixels while values at the legend of OS is based on area under the curve (AUC). Note that the features extracted from the VGG-Net are more effective than the AlexNet for tracking because strengthened semantic with deeper architecture is more insensitive to significant appearance change. In addition, the tracking performance is improved with hierarchically inference on the translation cues using multiple CNN layers.

Qualitative Evaluation. Figure 8 shows some tracking results of the top performing tracking methods: MEEM [34], KCF [17], DLT [30], Struck [13], and the proposed algorithm on 12 challenging sequences. The MEEM tracker performs well in sequences with deformation, rotation and occlusion (*Basketball*, *Bolt*, *Jogging-1*, and *Skiing*), but fails when background clutter and fast motion occur (*Board*, *Soccer*, *Diving*, *MotorRolling*, and *Human9*) as the quantized color channels features are less effective in handling cluttered backgrounds. We also note that the MEEM drifts quickly as it use only luminance intensity features in the *Freeman4* sequence. The KCF tracker learns a kernelized correlation filter with a Gaussian kernel over HOG features. It performs well in sequences with partial deformation and fast motion (*Basketball*, *Bolt*), but drifts when target objects undergo heavy occlusions (*Jogging-1*) and rotations (*MotorRolling* and *Skiing*). The DLT method does not fully exploit the semantic and fine-grained information as we did, thus fails to track the targets on the selected challenging sequences. The Struck method does not perform well in sequences with deformation, background clutter and rotation (*Basketball*, *Bolt*, *MotorRolling* and *Skiing*), and heavy occlusions (*Jogging-1*). Although the use of structured outputs effectively alleviates the sampling ambiguity issues, the representation with hand-crafted features are not effective in accounting for large appearance changes.

The reasons that the proposed algorithm performs well can be explained by two main aspects. First, the visual representation using hierarchical convolutional features learned from a large-scale dataset are more effective than conventional hand-crafted features. With CNN

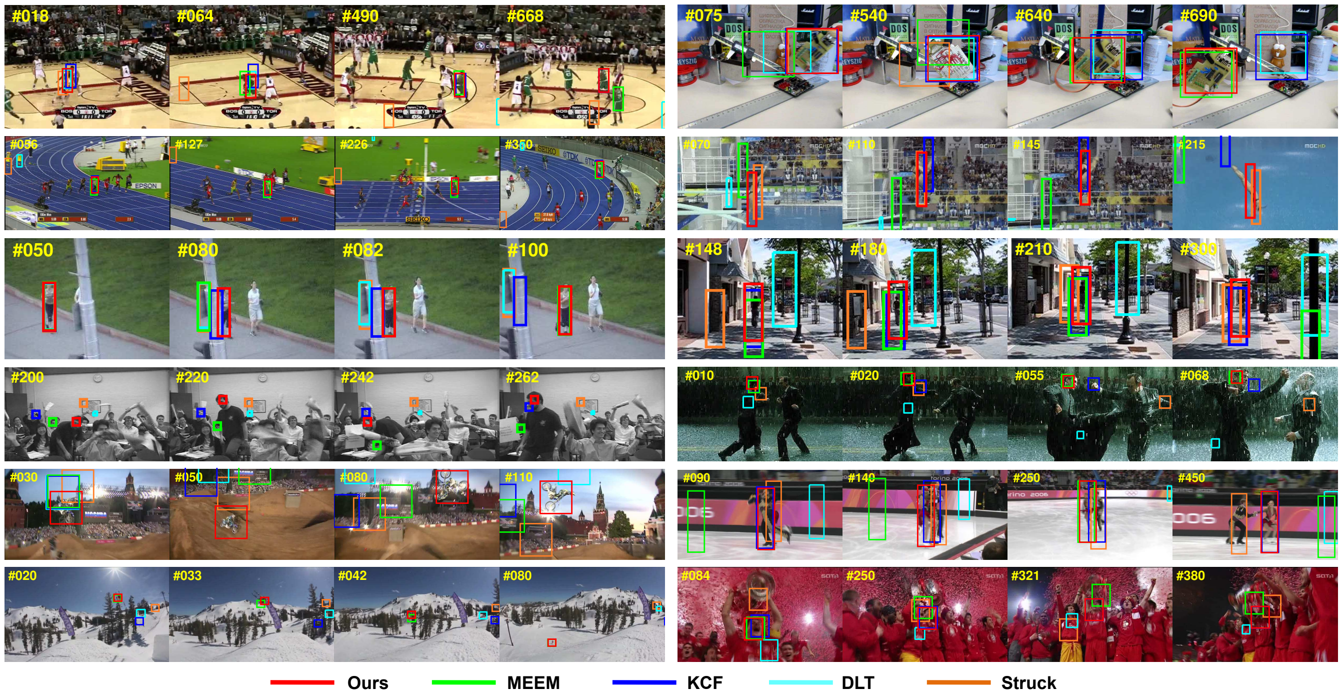


Figure 8. Qualitative evaluation of the proposed algorithm, the MEEM [34], KCF [17], DLT [30], Struck [13] methods on twelve challenging sequences (from left to right and top to down are *Basketball*, *Board*, *Bolt*, *Diving*, *Jogging-1*, *Human9*, *Freeman4*, *Matrix*, *MotorRolling*, *Skating2-1*, *Skiing* and *Soccer*, respectively).

features from multiple levels, our features contains both category-level semantics and fine-grained details, which account for appearance changes caused by deformations, rotations and background clutters (*Board*, *Soccer*, *Diving*, *Skiing*, and *Human9*). It is worth mentioning that for the most challenging *MotorRolling* sequence, none of the 12 state-of-the-art methods are able to track targets well whereas our method achieves the distance precision rate of 94.5%. Second, the linear correlation filters trained on convolutional features are updated properly to account for appearance variations.

Failure Cases. We show a few failure cases in Figure 9. For the *Girl2* and *Lemming* sequences, when long-term occlusions occur, the proposed tracker fails to follow targets as the proposed method is not equipped with a re-detection module as opposed to the TLD and MEEM methods. An alternative implementation with conservative update of correlation filters using (6) succeeds in following targets. For the *Singer2* sequence, it is not effective to use semantic features to differentiate the dark foreground from the bright background. In such cases, the use of features extracted on the first layer of CNNs alone is able to track the target well as fine-grained spatial details weigh more in this sequence.

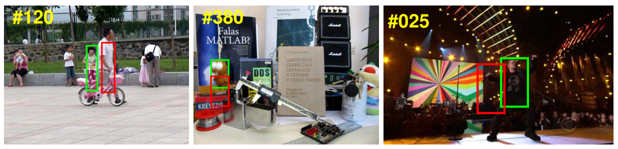


Figure 9. Failure cases (*Girl2*, *Lemming* and *Singer2*). Red boxes show our results and the green ones are ground truth.

7. Conclusions

In this paper, we propose an effective algorithm for visual object tracking by exploiting rich feature hierarchies of CNNs learned from a large-scale dataset. The last convolutional layers of CNNs retain semantics of target objects, which are robust to significant appearance variations. The early convolutional layers encode more fine-grained spatial details, which are useful for precise location. Both features with semantics and fine-grained details are simultaneously exploited for visual tracking. We train a liner correlation filter on each convolutional layer and infer the target position with a coarse-to-fine searching approach. Extensive experimental results show that the proposed algorithm performs favorably against the state-of-the-art methods in terms of accuracy and robustness.

Acknowledgment. C. Ma and X. Yang are supported in part by NSFC Grants (61527804, 61025005, 61129001, and 61221001), STCSM Grants (14XD1402100, and 13511504501), 111 Program Grant (B07022). M.-H. Yang is supported in part by NSF CAREER Grant (1149783) and NSF IIS Grant (1152576).

References

[1] B. Babenko, M.-H. Yang, and S. Belongie. Robust object tracking with online multiple instance learning. *TPAMI*, 33(8), 2011. 2, 5, 6

[2] V. N. Boddeti, T. Kanade, and B. V. K. V. Kumar. Correlation filters for object alignment. In *CVPR*, 2013. 4, 5

[3] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui. Visual object tracking using adaptive correlation filters. In *CVPR*, 2010. 2, 4, 5

[4] A. C. Bovik, M. Clark, and W. S. Geisler. Multichannel texture analysis using localized spatial filters. *TPAMI*, 12(1):55–73, 1990. 2

[5] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005. 2

[6] M. Danelljan, G. Hager, F. S. Khan, and M. Felsberg. Accurate scale estimation for robust visual tracking. In *BMVC*, 2014. 2, 4

[7] M. Danelljan, F. S. Khan, M. Felsberg, and J. van de Weijer. Adaptive color attributes for real-time visual tracking. In *CVPR*, 2014. 2, 4

[8] J. Deng, W. Dong, R. Socher, L. Li, K. Li, and F. Li. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 3, 5

[9] H. K. Galoogahi, T. Sim, and S. Lucey. Multi-channel correlation filters. In *ICCV*, 2013. 4

[10] J. Gao, H. Ling, W. Hu, and J. Xing. Transfer learning based visual tracking with gaussian processes regression. In *ECCV*, 2014. 2, 5, 6

[11] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014. 1

[12] H. Grabner, C. Leistner, and H. Bischof. Semi-supervised on-line boosting for robust tracking. In *ECCV*, 2008. 2

[13] S. Hare, A. Saffari, and P. H. S. Torr. Struck: Structured output tracking with kernels. In *ICCV*, 2011. 2, 5, 6, 7, 8

[14] B. Hariharan, P. A. Arbeláez, R. B. Girshick, and J. Malik. Hypercolumns for object segmentation and fine-grained localization. *CVPR*, 2015. 4, 7

[15] S. He, Q. Yang, R. W. H. Lau, J. Wang, and M. Yang. Visual tracking via locality sensitive histograms. In *CVPR*, 2013. 5, 6

[16] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. Exploiting the circulant structure of tracking-by-detection with kernels. In *ECCV*, 2012. 2, 4, 5, 6

[17] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. High-speed tracking with kernelized correlation filters. *TPAMI*, 37(3):583–596, 2015. 2, 4, 5, 6, 7, 8

[18] S. Hong, T. You, S. Kwak, and B. Han. Online tracking by learning discriminative saliency map with convolutional neural network. In *ICML*, 2015. 1, 3

[19] Z. Kalal, K. Mikolajczyk, and J. Matas. Tracking-learning-detection. *TPAMI*, 34(7):1409–1422, 2012. 2, 5, 6

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 1, 2, 3, 7

[21] H. Li, Y. Li, and F. Porikli. Deeptrack: Learning discriminative feature representations by convolutional neural networks for visual tracking. In *BMVC*, 2014. 1, 2, 3

[22] X. Li, W. Hu, C. Shen, Z. Zhang, A. R. Dick, and A. van den Hengel. A survey of appearance models in visual object tracking. *ACM TIST*, 4(4):58, 2013. 2

[23] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *IJCAI*, 1981. 1

[24] D. A. Ross, J. Lim, R. Lin, and M. Yang. Incremental learning for robust visual tracking. *IJCV*, 77(1-3):125–141, 2008. 2

[25] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *ICLR*, 2015. 3, 4, 5

[26] A. W. M. Smeulders, D. M. Chu, R. Cucchiara, S. Calderara, A. Dehghan, and M. Shah. Visual tracking: An experimental survey. *TPAMI*, 36(7):1442–1468, 2014. 1, 2

[27] A. Torralba, R. Fergus, and W. T. Freeman. 80 million tiny images: A large data set for nonparametric object and scene recognition. *TPAMI*, 30(11):1958–1970, 2008. 2

[28] A. Vedaldi and K. Lenc. Matconvnet – convolutional neural networks for matlab. *CoRR*, abs/1412.4564, 2014. 5

[29] L. Wang, T. Liu, G. Wang, K. L. Chan, and Q. Yang. Video tracking using learned hierarchical features. *TIP*, 24(4):1424–1435, 2015. 1, 2, 3

[30] N. Wang and D. Yeung. Learning a deep compact image representation for visual tracking. In *NIPS*, 2013. 1, 2, 3, 5, 6, 7, 8

[31] Y. Wu, J. Lim, and M.-H. Yang. Online object tracking: A benchmark. In *CVPR*, 2013. 2, 5, 6

[32] Y. Wu, J. Lim, and M.-H. Yang. Object tracking benchmark. *TPAMI*, PrePrints. 2, 5, 6

[33] A. Yilmaz, O. Javed, and M. Shah. Object tracking: A survey. *ACM Computing Surveys*, 38(4), 2006. 1, 2

[34] J. Zhang, S. Ma, and S. Sclaroff. MEEM: robust tracking via multiple experts using entropy minimization. In *ECCV*, 2014. 2, 5, 6, 7, 8

[35] K. Zhang, L. Zhang, Q. Liu, D. Zhang, and M.-H. Yang. Fast visual tracking via dense spatio-temporal context learning. In *ECCV*, 2014. 2, 4, 5, 6

[36] K. Zhang, L. Zhang, and M.-H. Yang. Real-time compressive tracking. In *ECCV*, 2012. 5, 6

[37] Q. Zhao, Z. Yang, and H. Tao. Differential earth mover's distance with its applications to visual tracking. *TPAMI*, 32(2):274–287, 2010. 2

[38] W. Zhong, H. Lu, and M.-H. Yang. Robust object tracking via sparse collaborative appearance model. *TIP*, 23(5):2356–2368, 2014. 5, 6

[39] W. Y. Zou, A. Y. Ng, S. Zhu, and K. Yu. Deep learning of invariant features via simulated fixations in video. In *NIPS*, 2012. 2