Deep learning to frame objects for visual target tracking

Shuchao Pang[b], Juan José del Coz[a], Zhezhou Yu[b], Oscar Luaces[a,*], Jorge Díez[a]

[a] *Artificial Intelligence Center*
University of Oviedo
33204 Gijón, Spain
[b] *Coll. Computer Science and Technology*
Jilin University
Changchun, 130012, China

Abstract

We present a new approach to deal with visual tracking target tasks. This method uses a convolutional neural network able to rank a set of patches depending on how well the target is framed (centered). To cover the possible interferences our proposal is to feed the network with patches located in the surroundings of the object detected in the previous frame, and with different sizes, thus taking into account eventual changes of scale. In order to train the network, we had to create an ad-hoc large dataset with positive and negative examples of framed objects extracted from the Imagenet detection database. The positive examples were those containing the object in a correct frame, while the negative ones were the incorrectly framed. Finally, we select the most promising patch, using a matching function based on the deep features provided by the well-known AlexNet network. All the training stage of this method is offline, so it is fast and useful for real-time visual tracking. Experimental results show that the method is very competitive with respect to state-of-the-art algorithms, being also very robust against typical interferences during the visual target tracking process.

Keywords: Deep convolutional networks, deep learning, target tracking visualization

1. Introduction

Among all applications of computer vision, visual target tracking is probably the most challenging problem in recent years, which has also been attracting more and more attention. A wide range of applications are based on visual

*Corresponding author
Email addresses: `pangshuchao1212@sina.com` (Shuchao Pang), `juanjo@uniovi.es` (Juan José del Coz), `yuzz@jlu.edu.cn` (Zhezhou Yu), `oluaces@uniovi.es` (Oscar Luaces), `jdiez@uniovi.es` (Jorge Díez)

tracking technology, like vehicle navigation, augmented reality and video safe surveillance (see more examples in (Wu et al., 2013)). The goal is to locate and follow the trajectory of a moving target in a video sequence starting with little a priori knowledge about the object, mainly the bounding box (e.g. location and size) in the first frame of the sequence. Due to severe visual appearance changes caused by different reasons such as geometric deformation, illumination variations, partial and full occlusions, motion blur, scale changes and/or fast motion, visual target tracking problem is a tough and a challenging task (Wu et al., 2015). These circumstances make target tracking an active research field in the last years.

Among the different tasks that must be tackled to accomplish a visual target tracking system, some authors think that *feature extraction* and *representation* are the most important (Li et al., 2013; Black and Jepson, 1998; Ross et al., 2008). Based on this idea, many algorithms were proposed (Comaniciu et al., 2003; Dalal and Triggs, 2005; Tuzel et al., 2006; Wu et al., 2012; Grabner et al., 2006), and further promoted the development of generative models for target tracking (Liu et al., 2011; Zhong et al., 2012; Jia et al., 2012; Kwon and Lee, 2010, 2011; Ross et al., 2008). In addition to these traditional and hand-crafted features, nowadays, deep features are showing a strong ability in image representation, in the same way than deep neural networks have been successfully applied on many computer vision applications recently (Ma et al., 2015; Wang et al., 2015; Wang and Yeung, 2013; Taigman et al., 2014; Ouyang et al., 2016; Qi, 2016; Carreira et al., 2016).

On the other hand, other papers payed more attention on the classification ability of algorithms in order to decide which candidate image patch could be classified and ranked as the final real target in each frame. This kind of methods generate a discriminative model. For example, some good discriminative models can be found in (Babenko et al., 2009; Zhong et al., 2012; Avidan, 2004; Hare et al., 2011; Henriques et al., 2015). However, all the methods proposed only address the tracking problem up to a certain extent. Moreover, analyzing the existing works in this field, we can conclude that i) they usually extract target features firstly and then find the most similar image patch (called Generative Model) or ii) they train and fine-tune a classifier to distinguish the positive and negative image patches (called Discriminative Model).

In this paper, we present a robust visual target tracking approach based on deep learning, in which we propose a fusion between both discriminative and generative models.

In our framework, we firstly use a deep learning network to obtain a discriminative object location model. This network will be trained to discriminate from well framed and poorly framed objects. Then, we construct a matching score function to verify which object in the current frame matches the target object set in the first frame. Therefore, the *motion object location* combined with *target verification* is the essence of the visual target tracking proposed in this paper.

This method improves the performance of the tracking process, its training is completely offline, and it is able to deal with a large number of disturbing interferences.

In order to accurately detect motion objects similar to the original target established at the beginning of the tracking process, we use a deep learning network specially tuned to detect correctly framed objects, based on a domain transferred deep convolutional neural network (DT-DCNN) architecture. We call it Deep Framer Network (DFN). To train our DFN, we manually built more than half a million positive (well framed) and negative (poorly framed) object patches. After training our network, we propose a matching score function based on deep features to verify whether the objects from next frame are the tracked target or not. The detailed procedure steps are reported in the next sections.

The major contributions of our work in this paper are:

- we built a large object dataset, with 668404 images, that can be used to train models to discriminate well framed from poorly framed objects (we will make this dataset public available when publishing the paper - 10GB),
- we deeply analyzed the essence of visual target tracking from a large object dataset and construct the Deep Framer Network (DFN) to discover and frame objects under difficult situations with numerous distracting factors during the tracking process,
- we developed an adaptive matching score function to verify the tracked target in the current frame from a group of candidate framed objects, with no online model update.

2. Related work

In this section, we shortly introduce some works related to our proposed method, including an overview to visual target tracking, as well as to deep neural networks.

2.1. Visual target tracking overview

In the past decade, visual target tracking has been extensively studied and significant progress has also been made in the area of computer vision (Wu et al., 2015). This section provides an overview for visual target tracking from two perspectives, which are i) tracking models and ii) target feature representations.

From the point of view of *tracking models*, most tracking methods fall into generative or discriminative models. Those generative methods describe the target appearance by a generative model and search for the candidate with maximum likelihood with respect to the tracked target in the next frame. LSK method (Liu et al., 2011) proposed a local sparse appearance model to enhance visual target tracking robustness by combining with the mean shift algorithm to locate targets. In (Jia et al., 2012), the authors regarded the target as the composition of different local image patches with spatial layout based on sparse codes. To deal with drastic lighting changes and fast motion, Kwon and Lee (2010) used multiple observation and motion models to construct a target tracking decomposition approach, which accounted for a relatively large appearance variation. In (Kwon and Lee, 2011), the sampling of trackers using Markov Chain Monte Carlo was proposed to search for more suitable trackers, which was an extension work based on (Kwon and Lee, 2010). Due to the ceaseless appearance changes of the tracked target, Ross et al. (2008) developed a system that incrementally updates the eigenbasis and it adapts to the changes in the tracking process.

In contrast, discriminative modeling algorithms regard the target tracking problem as a kind of classification problem using a built model to distinguish the target from the background. Babenko et al. (2009) proposed an online target tracking algorithm by constructing the Multiple Instance Learning framework (MIL). In (Avidan, 2004), a trained Support Vector Machine (SVM) classifier was integrated in an optical flow framework to deal with target appearance changes. Struck algorithm (Hare et al., 2011) utilized kernelized structured SVM to design the target tracking model which can exploit the constraints of the predicted outputs. Henriques et al. (2015) derived a new Kernelized Correlation Filter (KCF) with Discrete Fourier Transform to reduce both storage and computation by several orders of magnitude for target tracking tasks, which showed a powerful discriminative capability between the target and the surrounding environment. In addition, Zhong et al. (2012) tried a fusion target tracking model between a sparsity appearance generative model and a discriminative classifier. Generally, those tracking methods by detection and deep learning methods for target tracking also can be regarded as the special category of discriminative models.

From the view of *feature representations*, there are several features which are used into visual target tracking task. Generally speaking, we can divide the features into two categories, *hand-crafted features* and *deep features*. Hand-crafted features can be usually regarded as artificial, low-level features. They were widely used in target tracking process to address the challenging problem of interferences up to a certain extent. Belonging to this category of features, we can mention, for instance, color histograms (Comaniciu et al., 2003), histograms of oriented gradients (HOG) (Dalal and Triggs, 2005), covariance region descriptors (Tuzel et al., 2006; Wu et al., 2012), and Haar-like features (Grabner et al., 2006).

Hand-crafted features are designed at the pixel-level of the image, which can be unstable due to the numerous disturbances during the tracking process. Moreover, the key point is that pixel-level features belong to shallow features that are not capable to capture deep and semantic information of the target, which will lead to target tracking drift in some challenging video sequences.

Deep features became more attractive for computer vision tasks, since neural networks were again in the spotlight. These compact and semantic feature representations are discovered by a ceaseless iterative training on large image datasets. Many vision tasks use deep features which also proved to be very effective at extracting semantic features and classifying objects of various categories. or visual target tracking, on the one hand, deep features are used as a black box to represent the tracked target, like SDAE (Wang and Yeung, 2013;

Hong et al., 2015). On the other hand, several recent visual tracking algorithms began to directly train their deep model with existing target tracking sequences for learning real and semantic target features, such as (Nam and Han, 2016; Bertinetto et al., 2016; Zhang et al., 2016).

2.2. Deep neural networks in computer vision

In recent years, deep neural networks have been developed and applied to numerous computer vision tasks, for example, image classification and recognition (Krizhevsky et al., 2012; Simonyan and Zisserman, 2014), object detection (Girshick et al., 2014; Ouyang et al., 2016), image segmentation (Long et al., 2015; Qi, 2016), face verification (Taigman et al., 2014), and human pose estimation (Toshev and Szegedy, 2014; Yang et al., 2016; Carreira et al., 2016). The main reasons for deep neural networks to be more popular these days are their accuracy, efficiency and flexibility. Worth of mention are the advances in hardware, which make possible for deep neural networks to deal with problems whose size would prevent us from using such techniques some years ago. More specifically, the great success of deep neural networks in the field of computer vision is mostly attributed to their hierarchical hidden feature layers that outperforms hand-crafted features. However, the popularity of deep neural networks is still not very extended for visual target tracking task since it is hard to collect a large number of labeled images to deal with this problem.

To alleviate this data scarceness problem, some authors (Wang and Yeung, 2013; Zhang et al., 2016; Hong et al., 2015) propose to *transfer* deep features from a previously (offline) trained deep neural network on a large auxiliary dataset. For example, Wang and Yeung (2013) chose stacked denoising autoencoder (SDAE) as their deep model, which was trained offline using unsupervised learning, on the large Tiny dataset. Then, a fine-tuning of model parameters was performed according to the target tracking on different videos, in order to prevent drift failure. This type of deep networks were widely used, because unsupervised learning does not need labeled images. However, fully connected neural networks have millions of parameters to tune and, in addition, they ignore the spatial structure of images. These issues lead the researchers to propose Convolutional Neural Networks (CNN) (Krizhevsky et al., 2012; Simonyan and Zisserman, 2014; Szegedy et al., 2015), whose use is spreading in all areas related to image processing.

CNNs are specially devised to deal with images as input, so certain properties can be encoded in the architecture, making the forward function more efficient, as well as reducing the amount of parameters in the network.

As opposed to typical multilayer perceptrons, CNNs have the following characteristic features:

- Layers are arranged in 3 dimensions: width, height and depth, and each processing unit or neuron of a layer is only connected to a small region of the previous layer; it is the *receptive field* of the neuron. In addition, a CNN is composed of distinct types of layers, some locally and other fully connected.

- The local connectivity takes advantage of spatially local correlation in the images. These subsets of locally connected neurons can create good representations of small parts of the input image. They act as local filters, once the network is trained. More complex representations can be achieved by stacking several locally connected layers.

- The weights of the locally connected regions are shared among all neurons of the same layer. In other words, the filter learned is the same for each neuron of a given layer, but since it receives its input from a different region of the input image, it allows to detect a feature regardless of its position in the image. This gives the CNN the property of translation invariance.

Several successful CNN models were presented in the Imagenet Classification Challenge Match, such as Alexnet (Krizhevsky et al., 2012), VGG (Simonyan and Zisserman, 2014) and GoogLeNet (Szegedy et al., 2015).

3. Our proposed algorithm

Target tracking in a video sequence requires processing each frame, i.e. each static image, so we expect to take advantage of the benefits provided by deep neural networks, and particularly CNNs, in image processing. For this purpose we need to design a specific convolutional neural network which we will call Deep Framer Network (DFN).

In this section, we give a detailed explanation of our proposed DFN tracker. First we justify and explain the use of a domain transferred convolutional neural network. Then we describe the full architecture of our proposal, how it was trained and how it works during the tracking process. The end of the section is devoted to explain our approach to tackle the changes of scale in the target, a recurrent problem in visual tracking tasks.

3.1. Domain Transferred Deep Convolutional Neural Networks (DT-DCNN)

Training a deep neural network with millions of parameters is difficult and infeasible in some domains due to the lack of properly labeled training samples. To tackle this issue, other authors have adopted *transfer learning*, which consists in using the obtained parameters (all or part of them) of a network previously trained on a different, but related, domain (Oquab et al., 2014; Tompson et al., 2015). Tracking an object in a video sequence can be thought of as locating the same given object in each frame of the video sequence. Thus, instead of building from scratch a completely new DCNN mode, we propose to develop a domain transferred deep convolutional neural network starting from a deep model pre-trained to identify different objects in a static image. This model will then be fine-tuned for our specific tracking task, i.e. for correctly frame the target inside a bounding box.

For this purpose, we tested Alexnet, VGG and GoogLeNet deep models: all these deep models are pre-trained on millions of images from the Imagenet challenge. The classification performance among the three deep models is different, as shown in (Szegedy et al., 2015): VGG and GoogLeNet yield a similar performance, and they are better than Alexnet. However, we chose Alexnet despite its lower performance in classification because it was trained on small size images (227×227), which is more appropriate for our purpose; the size of image patches in target tracking is usually several dozens of pixels, at most. VGG and GoogLeNet were trained with $\approx 1000 \times 1000$ sized images, and we empirically found that the deep features generated from their hidden layers will become more blurred and difficult to distinguish from their semantic information in these larger deep models.

Once we have decided to use Alexnet to provide the parameters for our network we have to fine-tune them. For this purpose our first attempt could have been to use the target samples of the previous frames, but this approach would suffer from severe overfitting and would lead to track drift, as stated in (Wang et al., 2016). During the tracking process the target usually undergo all kinds of interferences, like changes in its appearance, partial and/or full occlusion. Therefore, focusing on recently obtained target patches (from previous frames) would yield a model that would be less likely to fit the changes that the target will suffer in future frames. As we introduce in next subsection, we will construct a more general training dataset to fine-tune the domain transferred from Alexnet and to adjust the pre-trained model to our needs.

3.2. Deep Framer Network (DFN)

The multi-layered architecture of our proposed network is depicted in Figure 1(b). It takes $227 \times 227 \times 3$ images as input, it has 5 convolutional layers (conv1-5) to convolute local features with convolutional kernels and 3 fully connected layers (fc6-8) to perform classification operation and represent target features. Additionally, there are 3 max pooling layers (pool1-2, 5) to reduce the dimensionality of feature maps and 2 local response normalization layers (norm1-2) to refine the pooling layers.

We use three different sized convolutional kernels, which are 11×11, 5×5 and 3×3 pixels. In order to transfer the pre-trained parameters from the image classification domain, the initial weight parameters of all convolutional layers and the first fully connected layer are identical to the corresponding parts of Alexnet networks. However, the parameters of the second and third fully connected layers in our DFN network are randomly initialized and their size is 500 and 2, respectively.

The core technology of CNNs is the convolutional operation applied to the input images and the feature maps from last layer. The first convolutional operation in our DFN is performed between the input image (candidate patch) and the `conv1` layer. The number of input channels is $C_0 = 3$, corresponding to the RGB channels of the input images, and the number of channels in `conv1` is set to $C_1 = 96$, so the c-channel feature map in the `conv1` layer is calculated by convolving the 11×11 kernel with the input image $F_0(X)$, which is a $227 \times 227 \times 3$

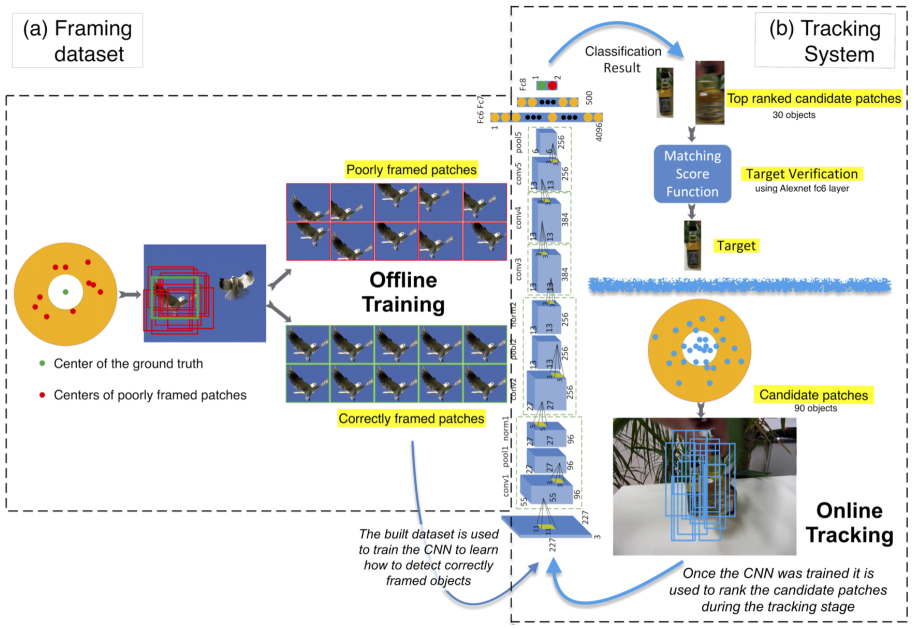

(a) Framing dataset
(b) Tracking System
Classification Result
Top ranked candidate patches
30 objects
Matching Score Function
Target Verification
using Alexnet fc6 layer
Target
Poorly framed patches
Offline Training
Correctly framed patches
Center of the ground truth
Centers of poorly framed patches
Candidate patches
90 objects
Online Tracking
The built dataset is used to train the CNN to learn how to detect correctly framed objects
Once the CNN was trained it is used to rank the candidate patches during the tracking stage

Figure 1: Our proposed Deep Framer Network (DFN) architecture for visual target tracking. It comprises two stages: (a) positive and negative image patches were generated as training dataset to offline train our DFN. (b) once the DFN was fine-tuned to detect correctly framed objects, it is used to complete online the visual target tracking task.

matrix, as follows

$$F^c_{\text{conv1}}(X) = \sum_{k=1}^{C_0} w^c_k \times F^k_0(X) + b_c, \quad c = 1 \dots 96, \tag{1}$$

where w^c_k are the parameters of the 11×11 convolutional kernel connecting the k-th channel of input image with the c-th channel of the conv1 layer.

In our network, we choose ReLU layers interleaved among these hidden layers as the nonlinear activation unit, which can vanish the gradient problem and can improve the learning speed and the classification accuracy more than the traditional sigmoid and tangent functions.

3.2.1. Training the DFN

Our objective is to use the proposed DFN to learn how to correctly enclose the target in a bounding box in each frame of a video sequence. For this purpose we built-up a training dataset from the ILSVRC2013 Validation Dataset in order to fine-tune the parameters transferred from Alexnet. We used a data augmentation approach to obtain several patches from each detected object in the original ILSVRC2013 dataset. Provided that the ground truth is bounded by a rectangular frame of width w, height h and center $C(x_c, y_c)$, these patches were generated as follows:

- **10 positive patches** were generated by repeating two times the original (correctly framed) patch and adding 8 more patches with the same dimensions of the original but varying their centers ± 1 pixel both in X and Y coordinates.
- **10 negative patches** were also generated with centers $C_i(x_i, y_i)$, where $x_i \sim \mathcal{N}(x_c, w/6)$ and $y_i \sim \mathcal{N}(y_c, h/6)$, and provided C_i does not fall inside a rectangle of $\frac{w}{5} \times \frac{h}{5}$ pixels, centered in C.

This idea is graphically depicted in Figure 1(a) with an orange pseudo-ring.

Thus, we build 20 positive and negative examples for each object in each image. Since the ILSVRC2013 Validation Dataset has 20121 images, we generate 555020 positive and 521360 negative examples with our sampling strategy. These examples were then filtered to remove those frames that only contained part of the tracking object. ILSVRC2013 contains many objects which appear only partially on an edge of the image, but we are not interested in showing those partial objects to our DFN since detecting only part of an object in target tracking tasks is suboptimal. Therefore, after carefully analyzing the generated dataset, we manually removed the undesired examples, leaving a total of 335960 positive and 332444 negative examples.

The DFN was trained using a stochastic gradient descent (SGD) algorithm. In every training iteration the parameters of the network were updated based on a minibatch of randomly selected positive and negative examples. We set the learning rate to $1 \cdot 10^{-5}$, momentum to 0.9, weight decay to $5 \cdot 10^{-4}$ and the size of the training minibatch was 128 examples. The stopping criterion in the training process was based on convergence of the model, but limited to a predefined number of 36000 maximum iterations.

3.2.2. Using the trained model

In this section we describe how the trained DFN is used to track an object in a video sequence. This operation is depicted graphically in Figure 1(b). The starting point is the bounding box of the object to be tracked in the first frame of the video sequence. The procedure then processes the rest of the video sequence, starting in the second frame, as follows:

Firstly, we generate 90 candidate patches whose center coordinates, $C_p(x_p, y_p)$, are randomly selected as $x_p \sim \mathcal{N}(x_c, w/6)$ and $y_p \sim \mathcal{N}(y_c, h/6)$, where $C(x_c, y_c)$ is the center of the bounding box of the previous frame. The size of this 90 patches is variable to deal with variations in scale and will be detailed in the next section.

Each patch is used as input image to feed the DFN, which will rank the patches according to its probability of being a correct bounding box for the tracked object.

The top 30 ranked patches go to the next stage, conceived to decide which one better corresponds to the tracked object. For this purpose, a *matching score* *function* was devised to compares the feature vector of the tracked object in the first frame of the video and the feature vectors of the patches, selecting the patch which yields the minimum Euclidean distance. In symbols,

$$L\left(T^f, \mathcal{F}^1\right) = \arg \min_{\mathcal{F}_i^f \in T^f} \|\mathcal{F}_i^f - \mathcal{F}^1\|_2^2 \tag{2}$$

where T^f is the set of 30 top ranked patches obtained from the foregoing DFN in frame f, F_i^f is the feature vector of the i-th patch in T^f and F^1 is the feature vector we extracted from the target in the first frame of the video sequence.

Note that we need a robust feature representation for this last stage. For this purpose we use AlexNet's fc6 layer to provide the feature vector for the first patch (the one in the first frame) of the tracked object as well as those for the 30 candidate patches of the current frame being processed. These feature vectors will be used in (2) to select the final patch.

We carried out a visual analysis to support our election of AlexNet as feature provider instead of our own DFN. Thus, we constructed a t-distributed Stochastic Neighbor Embedding (t-SNE) map to evaluate the representation ability and classification capability of the deep features for both AlexNet and DFN. T-SNE maps are used for dimensionality reduction, and they are particularly suited for the visualization of high-dimensional datasets. We used an implementation based on the Barnes-Hut algorithm (Van Der Maaten, 2014).

Figure 2(a) shows a t-SNE map made using the feature vectors from our DFN for the 90 candidate patches taken from the second frame of 50 video sequences used in our experiments. Correspondingly, Figure 2(b) shows the t-SNE map made with AlexNet's feature vectors. Both mappings show that the deep features obtained from the convolutional networks yield a similar grouping of patches, in the sense that patches belonging to the same object are usually closer one each other.

However, the feature representation from our DFN also takes into account how well positioned is the bounding box, since it was fine-tuned for that purpose. Thus, we can observe, for instance, that there is a group of close patches belonging to different objects (zoom number 1 in Figure 2). Their similarity relies in the fact that all those patches are poorly framed. We can also see some patches of the same object (zoom number 2) which are mapped separately, differencing the well framed from the poorly framed patches.

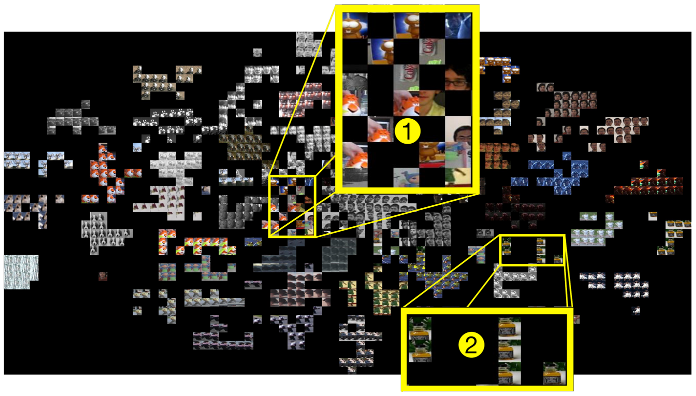
This bias on the property of being correctly framed could eventually lead our method to select an incorrect patch in this last stage, if there is more than one object in it, and the undesired one is better framed than the true target. Therefore, we opted for using the representation given by AlexNet's features.

3.3. Dealing with changes of scale

Changes of scale in the tracked object pose a challenging task in visual tracking processes. They are due mainly to two different reasons:

- When the change of scale is due to variation of the distance between the target and the camera, which is the general situation.

(a) DFN fc6 layer



(b) AlexNet fc6 layer

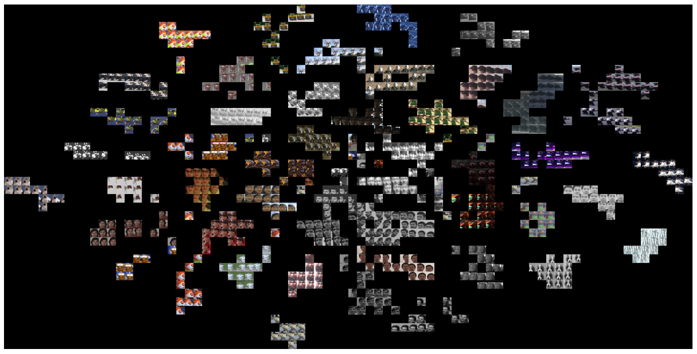


Figure 2: (a) t-SNE mapping of the 90 patches from the second frame of 50 video sequences using the deep features given by DFN as input. (b) t-SNE mapping with the same patches but using AlexNet's deep features. Although the grouping depends in both cases on the object of the patch, DFN features also represent the property of being (in)correctly framed.

- When the target object is partially occluded, so the bounding box of the visible part is smaller than in the frames previous to the occlusion.

Here, we do not adopt neither particle filter nor dense sampling search. These traditional methods search or sample too many candidate objects to distinguish which one is the most similar to the target, but these approaches usually increase the computing burden and reduce the running speed. In turn, our proposal consists of generating patches of different sizes in each frame f. Thus, the 90 patches mentioned in the previous section to feed the DFN will be of 3 different sizes (30 of each size):

i) *same size*, the size of the patch used in the previous frame $f-1$;

ii) *larger size*, expanding the dimensions of the previous patch in 2 pixels; and

iii) *smaller size*, shrinking the dimensions of the previous patch in 2 pixels.

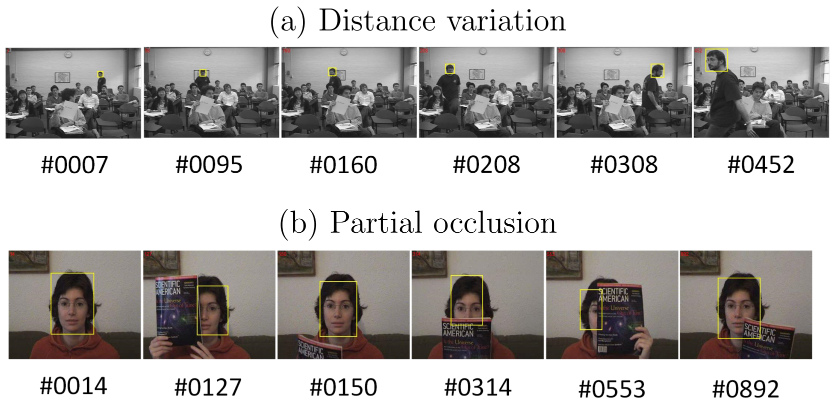


Figure 3: Target change of scale due to distance variation (a) and to partial occlusion (b) in a target tracking process over two different videos (Freeman3 and FaceOcc1, respectively) tracked by our proposed method.

Figure 3 illustrates the results obtained with this approach in two examples of scale change. In Figure 3(a) the man in the video is approaching to the camera so that the bounding box tracking his face should become larger as the video sequence advances. In turn, in Figure 3(b) the size of the bounding box has variations due to the partial occlusion of the target along the video sequence. We can observe that, when the occlusion disappears, the size of bounding box quickly recovers its normal size.

4. Experimental evaluation and results

In the experiments, we evaluate our proposed DFN tracker on a target tracking benchmark dataset (Wu et al., 2013, 2015), and compare its performance with 35 state-of-the-art trackers, including the 29 trackers used in (Wu et al., 2013) plus another 6 newer trackers: KCF (Henriques et al., 2015), TGPR (Gao et al., 2014), MEEM (Zhang et al., 2014), CNT (Zhang et al., 2016), CNN-SVM (Hong et al., 2015) and SCT4(Choi et al., 2016). Our deep network is trained with Caffe framework (Jia et al., 2014), and the source code for the tracking system is implemented in Matlab on a computer with 16 GB RAM, Intel(R) Core(TM) i7-4710HQ CPU @ 2.5GHz.

4.1. Evaluation methology

In order to compare our proposed method among other state-of-the-art visual trackers we will test all of them on 50 fully annotated video sequences presented in (Wu et al., 2013, 2015). The quality of the tracking process will be assessed in terms of *precision* and *success*, showing the corresponding plots to evaluate and compare the behavior of the trackers:

- *Precision plot*: describes the percentage of frames where the distance between the center of the bounding box of the tracked target and the center of the labeled ground truth is lower than a given amount of pixels. We used 51 different thresholds, $k_p \in \{0, 1, \ldots 50\}$

- *Success plot*: shows the percentage of frames whose overlap is larger than a given threshold. Overlap is defined as the intersection area between the predicted tracking box and the ground truth area divided by the union area of these two regions. Here, we tested 21 different thresholds, $k_s \in \{0, 0.05, 0.1, \ldots 1\}$.

Thus, precision and success rates of a tracker $t \in \{1, \ldots 36\}$ on a video sequence $s \in \{1, \ldots 50\}$ for a given threshold (k_p or k_s, respectively) are computed following the equations:

$$P_{t,s,k_p} = \frac{\sum_{n=1}^{F_s} [D_{t,s,f} \leq k_p]}{F_s} \tag{3}$$

and

$$S_{t,s,k_s} = \frac{\sum_{n=1}^{F_s} [O_{t,s,f} > k_s]}{F_s}, \tag{4}$$

where the number of frames of the video sequence s is F_s, the distance between the centers of the ground truth and the tracked target in frame f is $D_{t,s,f}$, and the overlap rate is $O_{t,s,n}$.

We can obtain one precision and one success plot for each tracker t and each sequence s varying the thresholds (k_p and k_s). To summarize such amount of graphs we need to aggregate the results. The authors of (Wu et al., 2013, 2015) propose to compute the average precision and success on all the video sequences as

$$\begin{aligned} \text{AveSqPrecision}_{t,k_p} &= \frac{\sum_{s=1}^{50} P_{t,s,k_p}}{50} \\ \text{AveSqSuccess}_{t,k_s} &= \frac{\sum_{s=1}^{50} S_{t,s,k_s}}{50} \end{aligned}$$

These equations have a main drawback: sequences with more frames have more importance in the final result. Instead, in this paper we are going to use the average precision and average success plots over the whole tracking sequences:

$$\text{AveFrPrecision}_{t,k_p} = \frac{\sum_{s=1}^{50} \sum_{n=1}^{F_s} [D_{t,s,f} \leq k_p]}{\sum_{s=1}^{50} F_s} \tag{5}$$

$$\text{AveFrSuccess}_{t,k_s} = \frac{\sum_{s=1}^{50} \sum_{n=1}^{F_s} [O_{t,s,f} > k_s]}{\sum_{s=1}^{50} F_s} \tag{6}$$

where the averages are computed at frame level, so the frames of all sequences have the same importance in the final result. In other words, we consider the

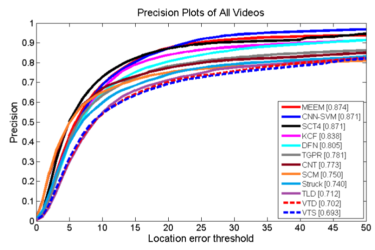
50 video sequences as a single large sequence with as many frames as the sum of frames of every original sequence (i.e., 29179 frames).

Another important characteristic of this benchmark dataset is that sequences are categorized according to 11 different attributes (Wu et al., 2013, 2015). Each attribute represents a specific challenging factor in object tracking (e.g., occlusion, fast motion, or illumination variation). One sequence may be annotated with many attributes, and some attributes occur more frequently than others. In Section 4.3 we will analyze the behavior of our approach dealing with these 11 attributes.

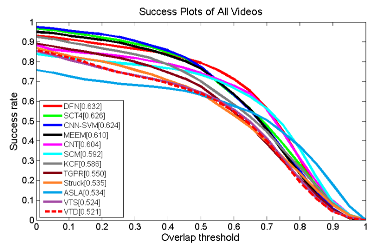
4.2. Overall results

In order to avoid cluttering the graphs, we only represent the results of the 12 algorithms with the best performance over all the 50 video sequences, as it can be seen in Figure 4.

The proposed algorithm in this paper (DFN) is good in precision, Figure 4(a): in the 80.5% of the frames, the distance between the center of the tracked bounding box and the ground truth is at most 20 pixels. It ranks 5 out of 36 state-of-the-art algorithms. In particular, the distance between our algorithm and the best one is getting lower when the threshold is increased.



(a)



(b)

Figure 4: Precision and success plots tested on the 50 sequences of the benchmark. We only show the top 12 trackers out of the 36 trackers used in the comparison. (a) precision plots drawn based on center location error; the numbers in the legend are the precision at 20 pixels. (b) success plots based on bounding box overlap ratio; the legend numbers indicate AUC scores.

Figure 4(a) shows the success plots. In order to analyze these curves we use the *area under the curve* (AUC) score. Our DFN tracker, denoted with the red line, outperforms all the state-of-the-art trackers, including recently published trackers as KCF, TGPR, MEEM, and SCT4. It also achieved a higher AUC than well reputed trackers based on convolutional neural networks, like CNT and CNN-SVM. Moreover, Figure 4(b) shows that when the overlap threshold is from 0.50 to 0.65, our DFN tracker is superior to all other trackers (including those not shown in the graph) with a considerable margin.

Let us recall that precision rate is only to measure the deviation of the center of the tracked bounding box with respect to the center of the true bounding box, it does not measure the deviation in size of the tracked bounding from the ground truth. On the other hand, success rate measure the overlap that takes into account both position and size of the bounding box.

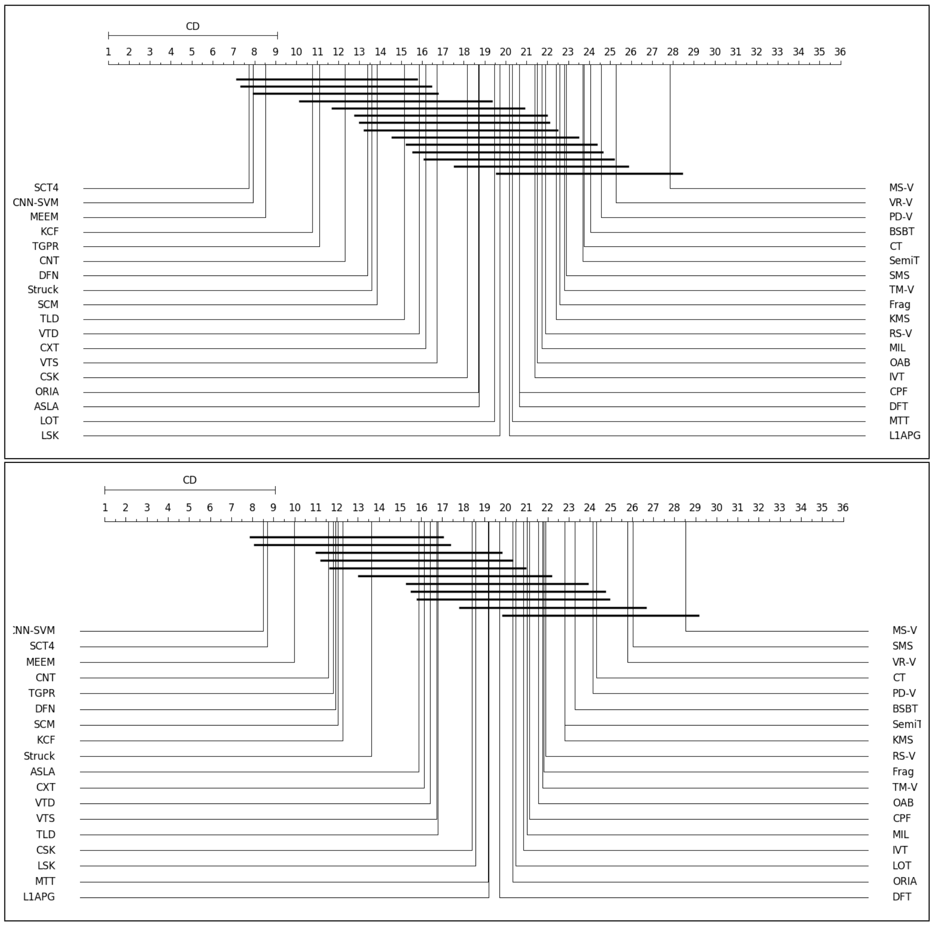


Figure 5: Precision (top) and success (bottom) Nemenyi tests

Tables with detailed numerical scores of precision and success rate AUCs obtained by each tracker over each video sequence are reported in an appendix. To sum up the results, the average rank position of each tracker is shown in the last row of each table. in order to provide statistical support to the conclusions of this analysis, and following the recommendations of (Demšar, 2006), we carried out a two-step statistical test procedure. The first step consists of a Friedman test where the null hypothesis states that all the trackers perform equally. Such hypothesis is rejected. Then, a Nemenyi test is performed to compare the methods in a pairwise way. Both tests are based on average ranks. The comparison includes 36 trackers over 50 videos, so the critical difference (CD) in the Nemenyi test is 8.085 for significance level of 5%. The results are depicted in Figure 5.

In these tests, DFN ranks 7th in precision and 6th in success with no significant difference between our method and the best algorithm in both cases. Most of the differences are not significant due, in part, to the high number of tracking algorithms used in the experiments. Note that there are some differences with respect the results in Figure 4. As was explained above, Figure 4 shows results at frame level (all the frames weight equal) while the scores in the Tables of the appendix and Figure 5 show results at video sequence level (frames in short video sequences weight more than in long sequences). However, the top 9 algorithms in the rank are the same: SCT4, CNN-SVM, MEEM, DFN, CNT, KCF, TGPR, SCM, and Struck.

4.3. Attribute-based Performance

It is acknowledge that there can be many kinds of interferences during a visual target tracking process, such as occlusion, fast motion, illumination variation, etc. The 50 video sequences in the benchmark are labeled with different attributes, as it is explained in (Wu et al., 2013, 2015), depending on the challenging factors present in the sequence. Moreover, many videos undergo several kinds of interference at the same time. To gain insight on the performance of our DFN tracker, we further report the performance of the top 12 trackers grouping the results with respect to the attributes of the video sequences.

Figure 6 shows precision plots of the trackers on the 11 different attributes. DFN presents a stable performance, having no particular difficulties with any specific attribute. Figure 7 shows the success rate plots. In this case, DFN is the best algorithm in 4 groups of videos, those presenting illumination variation, in-plane rotation, out-of-plane rotation and scale variation. Scale change is hard to handle for most trackers, regardless of the cause for the scale variation. Illumination variation is another tough and common problem in which significant changes of the illumination in the target region are produced. In-plane and out-of-plane rotation are the challenging difficulties for building a stable appearance model, because the target rotates in the image plane and out of the image plane.

In general, DFN yields good results, being robust to many different challenging situations as we have shown in this section.

5. Conclusion

We presented in this paper a robust method to perform visual target tracking. The main component of our system is a Deep Framer Network (DFN), a convolutional neural network whose initial parameters are taken from a pretrained AlexNet (domain transfer), and then they are fine-tuned to learn how to identify patches of correctly framed objects. The fine-tuning is carried out using a dataset created ad hoc, with 668404 positive (correctly framed) and negative (incorrectly framed) image patches. During the visual target tracking process the DFN ranks candidate patches which are submitted to a matching process against the features representing the original object to be tracked. For a more robust comparison our method uses deep features from AlexNet fc6 layer.

We have obtained very good results compared to state-of-the-art algorithms, our proposal does not need model updating and online fine-tuning process, making it very fast and, therefore, very useful for online real-time visual target tracking. In addition, the method has proven to be robust against typical interferences during the visual target tracking process.

Acknowledgments

This work was funded by Ministerio de Economía y Competitividad de España (grant TIN2015-65069-C2-2-R), Specialized Research Fund for the Doctoral Program of Higher Education of China (grant 20120061110045) and the Project of Science and Technology Development Plan of Jilin Province, China (grant 20150204007GX). The work was partially developed while Shuchao Pang was visiting the University of Oviedo at Gijón.

Avidan, S., 2004. Support vector tracking. IEEE transactions on pattern analysis and machine intelligence 26 (8), 1064–1072.

Babenko, B., Yang, M.-H., Belongie, S., 2009. Visual tracking with online multiple instance learning. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR'09). pp. 983–990.

Bertinetto, L., Valmadre, J., Henriques, J. F., Vedaldi, A., Torr, P. H., 2016. Fully-convolutional siamese networks for object tracking. In: European Conference on Computer Vision. Springer, pp. 850–865.

Black, M. J., Jepson, A. D., 1998. Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. International Journal of Computer Vision 26 (1), 63–84.

Carreira, J., Agrawal, P., Fragkiadaki, K., Malik, J., 2016. Human pose estimation with iterative error feedback. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4733–4742.

Choi, J., Jin Chang, H., Jeong, J., Demiris, Y., Young Choi, J., 2016. Visual tracking using attention-modulated disintegration and integration. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4321–4330.

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

Dalal, N., Triggs, B., 2005. Histograms of oriented gradients for human detection. In: Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on. Vol. 1. IEEE, pp. 886–893.

Demšar, J., 2006. Statistical comparisons of classifiers over multiple data sets. Journal of Machine Learning Research 7, 1–30.

Gao, J., Ling, H., Hu, W., Xing, J., 2014. Transfer learning based visual tracking with gaussian processes regression. In: European Conference on Computer Vision. Springer, pp. 188–203.

Girshick, R., Donahue, J., Darrell, T., Malik, J., 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 580–587.

Grabner, H., Grabner, M., Bischof, H., 2006. Real-time tracking via on-line boosting. In: Bmvc. Vol. 1. p. 6.

Hare, S., Saffari, A., Torr, P. H., 2011. Struck: Structured output tracking with kernels. In: IEEE International Conference on Computer Vision (ICCV'11). pp. 263–270.

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

Hong, S., You, T., Kwak, S., Han, B., 2015. Online tracking by learning discriminative saliency map with convolutional neural network. In: ICML. pp. 597–606.

Jia, X., Lu, H., Yang, M.-H., 2012. Visual tracking via adaptive structural local sparse appearance model. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR'12). pp. 1822–1829.

Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R., Guadarrama, S., Darrell, T., 2014. Caffe: Convolutional architecture for fast feature embedding. In: Proceedings of the 22nd ACM international conference on Multimedia. ACM, pp. 675–678.

Krizhevsky, A., Sutskever, I., Hinton, G. E., 2012. Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems. pp. 1097–1105.

Kwon, J., Lee, K. M., 2010. Visual tracking decomposition. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR'10). pp. 1269–1276.

Kwon, J., Lee, K. M., 2011. Tracking by sampling trackers. In: IEEE International Conference on Computer Vision (ICCV'11). pp. 1195–1202.

Li, X., Hu, W., Shen, C., Zhang, Z., Dick, A., Hengel, A. V. D., 2013. A survey of appearance models in visual object tracking. ACM Transactions on Intelligent Systems and Technology (TIST) 4 (4), 58.

Liu, B., Huang, J., Yang, L., Kulikowsk, C., 2011. Robust tracking using local sparse appearance model and k-selection. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR'11). pp. 1313–1320.

Long, J., Shelhamer, E., Darrell, T., 2015. Fully convolutional networks for semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3431–3440.

Ma, C., Huang, J.-B., Yang, X., Yang, M.-H., 2015. Hierarchical convolutional features for visual tracking. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 3074–3082.

Nam, H., Han, B., 2016. Learning multi-domain convolutional neural networks for visual tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 4293–4302.

Oquab, M., Bottou, L., Laptev, I., Sivic, J., 2014. Learning and transferring mid-level image representations using convolutional neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 1717–1724.

Ouyang, W., Wang, X., Zhang, C., Yang, X., 2016. Factors in finetuning deep model for object detection with long-tail distribution. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 864–873.

Qi, G.-J., 2016. Hierarchically gated deep networks for semantic segmentation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 2267–2275.

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

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

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., Rabinovich, A., 2015. Going deeper with convolutions. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1–9.

Taigman, Y., Yang, M., Ranzato, M., Wolf, L., 2014. Deepface: Closing the gap to human-level performance in face verification. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1701–1708.

Tompson, J., Goroshin, R., Jain, A., LeCun, Y., Bregler, C., 2015. Efficient object localization using convolutional networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 648–656.

Toshev, A., Szegedy, C., 2014. Deeppose: Human pose estimation via deep neural networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1653–1660.

Tuzel, O., Porikli, F., Meer, P., 2006. Region covariance: A fast descriptor for detection and classification. In: European conference on computer vision. Springer, pp. 589–600.

Van Der Maaten, L., 2014. Accelerating t-sne using tree-based algorithms. Journal of machine learning research 15 (1), 3221–3245.

Wang, L., Ouyang, W., Wang, X., Lu, H., 2015. Visual tracking with fully convolutional networks. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 3119–3127.

Wang, L., Ouyang, W., Wang, X., Lu, H., 2016. STCT: Sequentially training convolutional networks for visual tracking. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 1373–1381.

Wang, N., Yeung, D.-Y., 2013. Learning a deep compact image representation for visual tracking. In: Advances in Neural Information Processing Systems. pp. 809–817.

Wu, Y., Cheng, J., Wang, J., Lu, H., Wang, J., Ling, H., Blasch, E., Bai, L., 2012. Real-time probabilistic covariance tracking with efficient model update. IEEE Transactions on Image Processing 21 (5), 2824–2837.

Wu, Y., Lim, J., Yang, M., 2015. Object tracking benchmark. IEEE Transactions on Pattern Analysis and Machine Intelligence 37 (9), 1834–1848.

Wu, Y., Lim, J., Yang, M.-H., 2013. Online object tracking: A benchmark. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR'13). pp. 2411–2418.

Yang, W., Ouyang, W., Li, H., Wang, X., 2016. End-to-end learning of deformable mixture of parts and deep convolutional neural networks for human pose estimation. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 3073–3082.

Zhang, J., Ma, S., Sclaroff, S., 2014. Meem: robust tracking via multiple experts using entropy minimization. In: European Conference on Computer Vision. Springer, pp. 188–203.

Zhang, K., Liu, Q., Wu, Y., Yang, M.-H., 2016. Robust visual tracking via convolutional networks without training. IEEE Transactions on Image Processing 25 (4), 1779–1792.

Zhong, W., Lu, H., Yang, M.-H., 2012. Robust object tracking via sparsity-based collaborative model. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR'12). pp. 1838–1845.

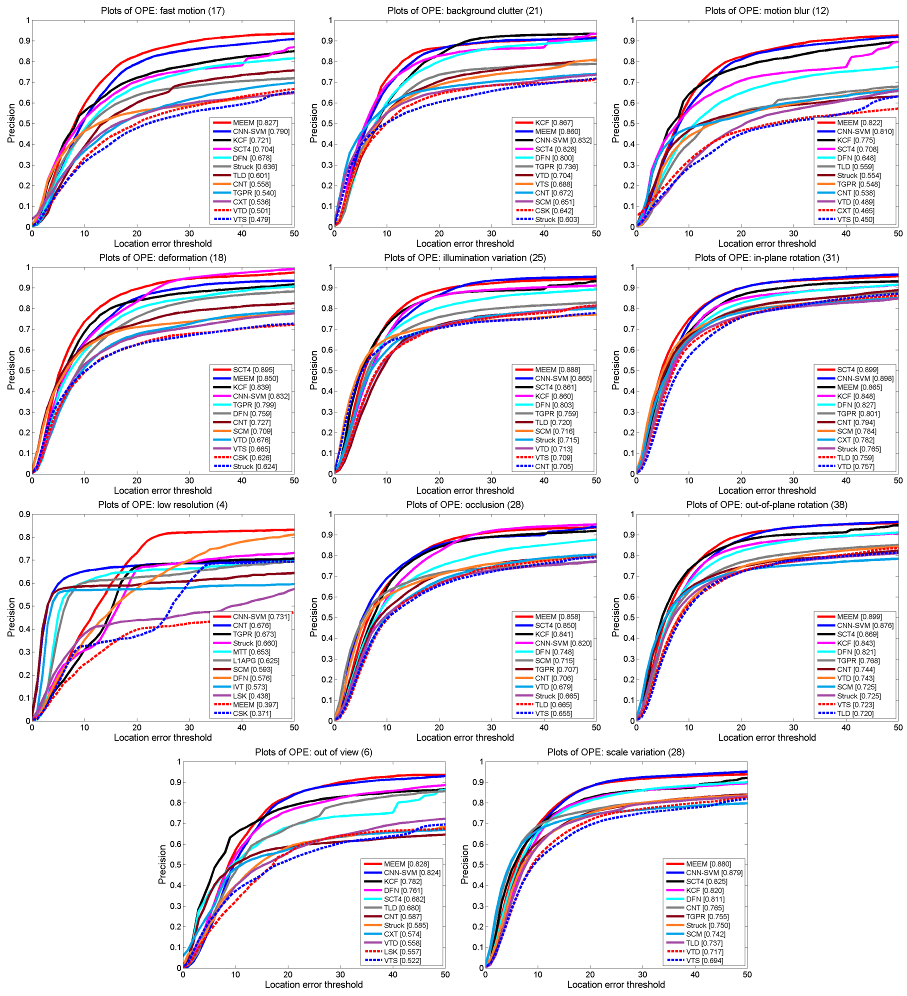


Figure 6: Average Precision rate on all related attribute sequences with the 12 leading trackers regarding each video attribute.

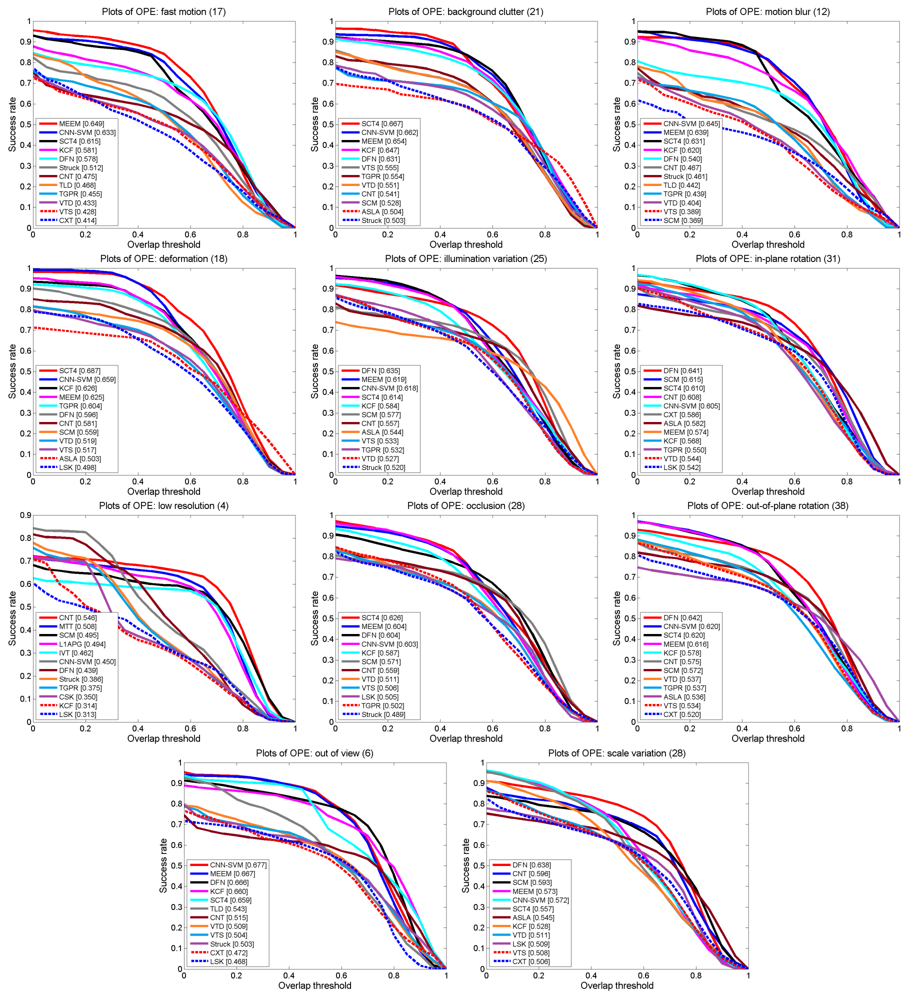


Figure 7: Average AUC scores based on success rate on all related attribute sequences with the 12 leading trackers regarding each video attribute.

Table A.1: Basic information of the video sequences used as benchmark: Start and End frame, and total number of frames.

Dataset	Start	End	Total	Dataset	Start	End	Total
david2	1	537	537	fleetface	1	707	707
sylvester	1	1345	1345	freeman1	1	326	326
mhyang	1	1490	1490	freeman3	1	460	460
david	300	770	471	david3	1	252	252
trellis	1	569	569	carScale	1	252	252
fish	1	476	476	dog1	1	1350	1350
carDark	1	393	393	suv	1	945	945
matrix	1	100	100	motorRolling	1	164	164
ironman	1	166	166	mountainBike	1	228	228
soccer	1	392	392	faceocc1	1	892	892
deer	1	71	71	football	1	362	362
singer1	1	351	351	subway	1	175	175
singer2	1	366	366	freeman4	1	283	283
skating1	1	400	400	skiing	1	81	81
coke	1	291	291	faceocc2	1	812	812
woman	1	597	597	tiger1	6	354	349
walking	1	412	412	tiger2	1	365	365
couple	1	140	140	lemming	1	1336	1336
football1	1	74	74	liquor	1	1741	1741
doll	1	3872	3872	basketball	1	725	725
girl	1	500	500	jumping	1	313	313
boy	1	602	602	jogging-2	1	307	307
dudek	1	1145	1145	bolt	1	350	350
crossing	1	120	120	car4	1	659	659
walking2	1	500	500	shaking	1	365	365

AppendixA. Detailed results

We include in this appendix detailed information about the experiments carried out in this work. Table A.1 shows the length of the video sequences used as benchmark. Tables A.2, A.3 and A.4 show the

Due to the large number of experiments, the scores obtained by all the trackers in all the video sequences, both in success and precision, are spread in several tables. Thus, Tables A.2, A.3 and A.4 contain two columns for each tracker, showing its score in terms of area under the curve in the success plots, and its position in the ranking with respect to the rest of trackers in the comparison. Correspondingly, Tables A.5, A.6 and A.7 display the scores in terms of area under the curve of precision.

Table A.2: (1/3) The left column under each tracker name contains the area under the curve of **success** for each video sequence, while the right column shows the position in the ranking with respect to the rest of the trackers.

Datasets	DFN		CNT		KCF		TGPR		MEEM		VR-V		TM-V		RS-V		PD-V		MS-V		DFT		ORIA	
david2	0.69	14	0.86	3	0.81	8	0.79	11	0.83	7	0.40	29	0.68	17	0.59	22	0.40	30	0.06	35	0.53	23	0.66	20
sylvester	0.67	6	0.61	16	0.64	11	0.73	2	0.67	5	0.26	34	0.26	33	0.62	14	0.52	27	0.36	32	0.38	31	0.64	9
mhyang	0.80	8	0.79	9.5	0.78	13	0.76	17	0.77	16	0.43	32.5	0.68	23	0.63	25	0.43	32.5	0.15	36	0.70	21	0.85	2
david	0.67	4	0.48	18	0.54	12	0.53	15	0.53	14	0.16	35	0.20	32	0.42	21	0.51	16	0.19	33	0.30	26	0.43	20
trellis	0.58	12	0.80	1	0.62	7	0.62	9	0.61	10	0.27	27	0.20	34	0.38	18	0.33	22	0.39	17	0.36	19	0.34	20
fish	0.80	7	0.73	14	0.82	3	0.81	5	0.75	11	0.46	21	0.45	22	0.26	29	0.45	23	0.28	28	0.74	12	0.51	20
carDark	0.61	18	0.78	9	0.61	19	0.75	11	0.84	3	0.04	34	0.62	17	0.32	29	0.15	32	0.38	25	0.38	27	0.38	26
matrix	0.05	32	0.14	16	0.12	20.5	0.29	3	0.39	1	0.01	36	0.06	29	0.24	7	0.22	9	0.06	28	0.06	30	0.22	10
ironman	0.04	32	0.13	6	0.14	5	0.15	4	0.37	1	0.05	30	0.05	26	0.03	34	0.11	12	0.07	23	0.05	25	0.11	11
soccer	0.19	11	0.16	24	0.43	2	0.15	25	0.32	5	0.24	7	0.14	28	0.18	15	0.05	36	0.26	6	0.17	20	0.19	12
deer	0.42	16	0.72	5	0.61	11	0.63	10	0.73	2	0.07	30	0.44	15	0.40	18	0.17	24	0.07	29	0.25	21	0.04	34
singer1	0.33	27	0.81	2	0.36	17.5	0.36	19	0.29	29	0.35	23	0.36	20	0.36	21.5	0.26	31	0.17	35	0.36	17.5	0.64	5
singer2	0.55	7	0.80	1	0.72	3	0.75	2	0.04	31	0.33	11	0.04	36	0.07	25	0.31	12	0.19	19	0.62	6	0.13	21
skating1	0.43	11	0.56	1	0.49	6	0.44	10	0.40	13	0.13	27	0.23	19	0.26	18	0.10	31	0.06	35	0.13	26	0.22	20
coke	0.53	8	0.35	12	0.55	7	0.63	4	0.66	2	0.03	35	0.06	33	0.19	23	0.22	17	0.02	36	0.11	32	0.20	20
woman	0.15	24	0.71	5	0.69	7	0.68	8	0.70	6	0.10	33	0.16	18	0.34	13	0.58	10	0.12	32	0.74	1	0.15	23
walking	0.36	28	0.75	3	0.53	20	0.56	12	0.54	16	0.24	32	0.32	29	0.51	24	0.25	30	0.03	36	0.56	13	0.12	35
couple	0.37	15	0.18	21	0.20	20	0.38	14	0.60	3	0.06	31	0.32	18	0.05	34	0.05	35	0.07	29	0.08	24	0.04	36
football1	0.54	17	0.50	20	0.70	4	0.59	12	0.67	5	0.63	10	0.14	34	0.55	15	0.64	9	0.29	31	0.84	1	0.10	36
doll	0.73	6	0.70	8	0.53	19	0.56	15	0.57	13	0.21	35	0.36	31	0.49	21	0.39	29	0.43	27	0.29	33	0.83	1
girl	0.68	7	0.58	13	0.54	17	0.60	12	0.70	6	0.32	29	0.61	11	0.38	26	0.36	28	0.30	32	0.29	33	0.24	34
boy	0.67	16	0.79	1	0.76	8	0.72	11	0.78	3	0.69	15	0.78	4	0.69	14	0.19	35	0.57	23	0.39	30	0.15	36
dudek	0.79	2	0.75	5	0.72	12	0.67	22	0.71	17	0.40	34	0.63	26	0.48	33	0.20	36	0.48	32	0.68	21	0.71	15
crossing	0.69	10	0.77	1.5	0.70	8	0.70	7	0.71	5	0.64	17	0.68	11	0.66	15	0.63	18	0.23	31	0.49	21	0.18	34
walking2	0.53	6	0.81	1	0.40	15	0.49	9	0.27	33	0.27	31	0.29	28	0.30	27	0.31	25	0.25	36	0.41	14	0.45	13
fleetface	0.74	1	0.61	5	0.58	12	0.60	8	0.60	9	0.18	34	0.45	28	0.23	31	0.17	35	0.21	33	0.48	24	0.52	18
freeman1	0.54	2	0.52	3	0.22	25	0.39	7	0.38	11	0.05	36	0.41	5	0.23	24	0.05	35	0.14	33	0.39	8	0.29	19
freeman3	0.63	5	0.66	4	0.33	13	0.29	23	0.35	11	0.15	31	0.31	18	0.22	26	0.15	32	0.17	30	0.32	16	0.36	9
david3	0.64	10	0.59	11	0.76	2	0.69	4	0.67	6	0.44	18	0.41	21	0.69	5	0.44	19	0.10	34	0.55	13	0.12	33
carScale	0.52	8	0.60	4	0.42	19	0.42	22	0.39	30	0.40	27	0.31	34	0.31	35	0.40	28	0.03	36	0.41	26	0.65	2
dog1	0.78	3	0.70	10	0.55	17.5	0.54	22	0.55	17.5	0.44	31.5	0.44	30	0.49	29	0.44	31.5	0.27	35	0.43	33	0.70	8
suv	0.58	15	0.81	3	0.86	1	0.58	16	0.53	18	0.18	32.5	0.67	8	0.34	29	0.18	32.5	0.05	36	0.08	35	0.54	17
motorRolling	0.59	1	0.09	33	0.09	28	0.13	9	0.10	25	0.11	18	0.07	35	0.16	5	0.07	36	0.10	26	0.09	34	0.12	14
mountainBike	0.60	19	0.73	2	0.70	9	0.71	6	0.59	20	0.06	36	0.13	32	0.67	14	0.17	30	0.57	22	0.29	27	0.64	16
faceocc1	0.60	28	0.70	16	0.74	8	0.72	12	0.72	10	0.74	7	0.78	2	0.69	18	0.68	19	0.56	32	0.68	20	0.64	25
football	0.50	25	0.55	16	0.55	17	0.67	5	0.68	3	0.07	34	0.63	9	0.12	31	0.07	35	0.11	32	0.65	6	0.51	24
subway	0.29	18	0.64	10	0.74	3	0.77	1	0.67	7	0.55	16	0.16	30	0.55	15	0.56	14	0.03	36	0.71	4	0.16	28
freeman4	0.55	1	0.13	24	0.18	14	0.31	6	0.42	2	0.23	8	0.12	27	0.03	34	0.23	9	0.03	32	0.17	17	0.20	13
skiing	0.05	29	0.06	22.5	0.05	25.5	0.09	9	0.41	2	0.07	13	0.09	6	0.07	14.5	0.32	3	0.05	24	0.05	25.5	0.06	19
faceocc2	0.66	17	0.59	25	0.74	6	0.47	28	0.69	15	0.23	32	0.69	14	0.17	34	0.22	33	0.13	35	0.76	4	0.71	13
tiger1	0.40	12	0.15	29	0.64	3	0.28	22	0.64	2	0.43	10	0.51	6	0.37	17.5	0.51	7	0.37	17.5	0.53	5	0.13	31
tiger2	0.39	10	0.16	28	0.36	13	0.56	4	0.54	7	0.09	35	0.54	5	0.23	21	0.12	32	0.18	25	0.67	1	0.18	26
lemming	0.68	1	0.34	25	0.38	22	0.35	24	0.67	2	0.11	35	0.36	23	0.57	11	0.57	10	0.64	6	0.40	21	0.04	36
liquor	0.74	5	0.45	16	0.84	1	0.26	24	0.77	4	0.48	14	0.32	22	0.55	9	0.20	34	0.42	19	0.21	31	0.23	29
basketball	0.65	9	0.05	31	0.67	7	0.65	8	0.80	1	0.51	12	0.13	28	0.47	16	0.51	13	0.38	20	0.60	11	0.03	33
jumping	0.34	14	0.56	11	0.28	15	0.59	9	0.69	2	0.20	18	0.68	3	0.06	34	0.63	7	0.25	16	0.11	26	0.09	29
jogging-2	0.68	7	0.73	3	0.12	33	0.70	5	0.63	10	0.13	26	0.12	34	0.12	32	0.14	24	0.10	36	0.23	17	0.46	13
bolt	0.56	5	0.57	4	0.67	3	0.01	32.5	0.43	10	0.12	21	0.01	35.5	0.43	9	0.12	20	0.11	22	0.03	24	0.18	12
car4	0.78	3	0.88	1	0.48	11	0.49	8	0.46	13	0.18	32	0.20	27	0.08	34	0.19	28.5	0.19	30	0.25	21	0.23	23
shaking	0.69	5	0.07	28	0.04	31	0.63	8	0.69	4	0.03	34	0.48	11	0.09	25	0.05	29	0.03	33	0.63	9	0.44	14
Avg*	0.63	11.94	0.60	11.61	0.59	12.28	0.55	11.83	0.61	9.97	0.29	25.79	0.41	21.75	0.42	21.89	0.34	24.12	0.29	28.53	0.41	19.72	0.46	20.34

Table A.3: (2/3) The left column under each tracker name contains the area under the curve of **success** for each video sequence, while the right column shows the position in the ranking with respect to the rest of the trackers.

Datasets	ASLA		L1APG		CT		TLD		IVT		MTT		CSK		SCM		LOT		CPF		OAB		SemiT	
david2	0.88	1	0.84	6	0.00	36	0.68	16	0.69	15	0.84	5	0.81	10	0.73	13	0.60	21	0.52	26	0.32	32	0.53	24
sylvester	0.58	19	0.40	30	0.66	8	0.67	7	0.51	28	0.64	10	0.63	12	0.68	4	0.56	22	0.55	24	0.56	23	0.53	25
mhyang	0.90	1	0.81	6	0.60	28	0.63	26	0.78	12	0.84	3	0.78	14	0.79	9.5	0.21	35	0.35	34	0.74	19	0.68	22
david	0.74	1	0.53	13	0.50	17	0.71	3	0.64	6	0.30	27	0.41	22	0.71	2	0.27	28	0.14	36	0.39	23	0.25	30
trellis	0.79	2	0.21	33	0.34	21	0.48	14	0.25	29	0.22	32	0.48	15	0.67	3	0.31	24	0.25	28	0.14	36	0.20	35
fish	0.83	2	0.35	26	0.71	15	0.80	8	0.76	10	0.18	34	0.22	31	0.74	13	0.24	30	0.20	33	0.04	36	0.22	32
carDark	0.83	4	0.86	2	0.00	36	0.44	22	0.65	15	0.81	8	0.74	12	0.83	5	0.42	24	0.08	33	0.77	10	0.81	7
matrix	0.13	17.5	0.16	14	0.02	35	0.16	13	0.02	34	0.23	8	0.03	33	0.26	6	0.12	20.5	0.07	27	0.26	5	0.14	15
ironman	0.12	8	0.09	19	0.07	21	0.11	13	0.05	28	0.09	18	0.12	9	0.12	7	0.09	16	0.04	33	0.03	35	0.04	31
soccer	0.11	31	0.17	22	0.17	21	0.13	30	0.16	23	0.18	16	0.15	26	0.24	8	0.21	10	0.21	9	0.09	34	0.07	35
deer	0.03	36	0.60	13	0.04	33	0.59	14	0.03	35	0.60	12	0.73	1	0.07	28	0.21	22	0.12	26	0.71	6	0.66	9
singer1	0.78	3	0.28	30	0.36	21.5	0.71	4	0.57	6	0.34	26	0.36	14	0.85	1	0.19	34	0.45	11	0.34	25	0.17	36
singer2	0.05	30	0.04	34	0.09	23	0.22	15	0.04	32	0.04	35	0.05	29	0.17	20	0.26	13	0.23	14	0.05	28	0.10	22
skating1	0.49	7	0.10	32	0.09	33	0.19	22	0.07	34	0.10	30	0.50	5	0.47	8	0.26	17	0.19	23	0.39	14	0.20	21
coke	0.17	27	0.18	26	0.24	15	0.40	11	0.12	30	0.44	9	0.57	6	0.32	14	0.13	29	0.23	16	0.34	13	0.22	18
woman	0.15	25	0.16	20	0.13	31	0.13	29	0.14	27	0.16	19	0.19	16	0.65	9	0.09	34	0.07	35	0.48	12	0.34	14
walking	0.76	1	0.74	4	0.52	21	0.45	27	0.75	2	0.66	7	0.53	18	0.70	5	0.69	6	0.64	8	0.54	15	0.25	31
couple	0.07	26.5	0.47	11	0.46	12	0.76	1	0.07	28	0.48	9	0.07	26.5	0.10	23	0.44	13	0.56	5	0.36	17	0.37	16
football1	0.49	21	0.56	14	0.24	33	0.37	26	0.55	16	0.54	18	0.46	22	0.40	24	0.54	19	0.61	11	0.27	32	0.14	35
doll	0.81	3	0.45	26	0.45	25	0.57	12	0.43	28	0.39	30	0.32	32	0.81	2	0.66	9	0.71	7	0.53	18	0.13	36
girl	0.70	5	0.72	2	0.31	30	0.57	14	0.17	35	0.66	10	0.37	27	0.67	8	0.41	23	0.43	22	0.71	4	0.48	20
boy	0.36	33	0.72	10	0.59	22	0.65	17	0.26	34	0.50	27	0.65	18	0.37	32	0.52	26	0.70	13	0.78	5	0.40	29
dudek	0.72	8	0.68	20	0.64	25	0.64	24	0.74	7	0.75	6	0.71	16	0.76	4	0.53	27	0.49	31	0.65	23	0.51	30
crossing	0.77	1.5	0.21	32	0.67	12	0.40	23	0.31	27	0.19	33	0.48	22	0.77	3	0.30	29	0.53	20	0.66	16	0.67	13
walking2	0.37	17	0.75	5	0.26	35	0.31	26	0.78	3	0.77	4	0.46	11	0.80	2	0.33	21	0.32	22	0.36	19	0.47	10
fleetface	0.56	14	0.45	27	0.52	17	0.49	22	0.46	26	0.50	19	0.58	11	0.60	7	0.57	13	0.41	29	0.52	16	0.36	30
freeman1	0.26	21	0.20	27	0.15	32	0.28	20	0.43	4	0.21	26	0.24	23	0.61	1	0.20	28	0.37	12	0.36	14	0.18	31
freeman3	0.73	1	0.35	12	0.00	36	0.44	7	0.39	8	0.46	6	0.30	21	0.71	2	0.13	33	0.11	34	0.26	24.5	0.17	29
david3	0.43	20	0.37	24	0.30	28	0.10	35	0.48	17	0.09	36	0.49	16	0.39	23	0.66	8	0.58	12	0.32	27	0.15	30
carScale	0.60	5	0.49	9	0.43	16	0.45	13	0.62	3	0.48	11	0.41	25	0.59	6	0.34	32	0.46	12	0.40	29	0.42	18
dog1	0.71	5	0.70	9	0.53	25	0.58	15	0.73	4	0.68	12	0.54	19	0.69	11	0.81	1	0.71	6	0.54	21	0.51	28
suv	0.49	21	0.47	24	0.25	30	0.68	7	0.40	28	0.44	26	0.51	20	0.74	4	0.64	9	0.61	12	0.62	10	0.48	22
motorRolling	0.11	23	0.09	27	0.11	24	0.23	3	0.09	30	0.09	29	0.09	31	0.12	17	0.13	10	0.12	13	0.11	19	0.16	6.5
mountainBike	0.71	7	0.73	3	0.14	31	0.20	29	0.72	5	0.73	1	0.70	8	0.66	15	0.57	21	0.11	35	0.62	18	0.44	25
faceocc1	0.33	36	0.74	9	0.63	26	0.58	30	0.72	13	0.69	17	0.78	3	0.78	4	0.41	35	0.53	33	0.65	24	0.71	14
football	0.53	23	0.54	19	0.60	10	0.49	26	0.55	15	0.57	13	0.54	18	0.48	27	0.65	7	0.63	8	0.33	28	0.21	30
subway	0.19	20	0.16	31	0.57	12	0.18	22	0.16	27	0.06	35	0.19	21	0.71	6	0.56	13	0.12	34	0.16	29	0.29	19
freeman4	0.13	26	0.34	5	0.01	36	0.22	10	0.15	21	0.22	11	0.13	25	0.26	7	0.16	18	0.06	30	0.11	29	0.21	12
skiing	0.09	7	0.06	18	0.07	14.5	0.07	16	0.08	10	0.09	8	0.06	20	0.08	11.5	0.02	36	0.03	35	0.08	11.5	0.05	27
faceocc2	0.64	20	0.68	16	0.60	24	0.61	23	0.72	11	0.73	8	0.77	2	0.72	12	0.46	30	0.43	31	0.59	26	0.51	27
tiger1	0.29	21	0.31	20	0.42	11	0.38	16	0.10	36	0.26	25	0.26	24	0.16	27	0.14	30	0.39	13	0.11	34	0.47	8
tiger2	0.15	30	0.25	18	0.45	9	0.27	17	0.09	36	0.29	16	0.18	24	0.09	34	0.13	31	0.24	19	0.15	29	0.24	20
lemming	0.15	30	0.13	34	0.54	14	0.53	15	0.14	33	0.28	29	0.33	26	0.14	32	0.59	8	0.56	13	0.59	9	0.14	31
liquor	0.24	27	0.20	35	0.21	32	0.51	10	0.23	28	0.20	33	0.25	26	0.32	23	0.81	2	0.46	15	0.44	17	0.51	12
basketball	0.38	21	0.23	24	0.26	23	0.02	36	0.11	30	0.19	27	0.70	5	0.46	17	0.67	6	0.49	15	0.03	34	0.03	32
jumping	0.23	17	0.15	21	0.05	36	0.66	5	0.12	24	0.10	28	0.05	35	0.12	23	0.57	10	0.10	27	0.08	31	0.06	33
jogging-2	0.14	22	0.15	20	0.10	35	0.65	8	0.14	21	0.12	31	0.14	23	0.72	4	0.13	27.5	0.60	11	0.41	15	0.70	6
bolt	0.01	30	0.01	29	0.01	35.5	0.16	15	0.01	34	0.01	31	0.02	25	0.02	26	0.52	6	0.48	8	0.04	23	0.13	19
car4	0.74	5	0.25	22	0.21	25	0.63	6	0.86	2	0.45	14	0.47	12	0.75	4	0.04	36	0.18	31	0.21	24	0.33	17
shaking	0.47	12	0.08	27	0.11	24	0.39	16	0.04	32	0.05	30	0.57	10	0.68	6	0.13	20	0.13	22	0.01	36	0.27	18
Avg*	0.53	15.89	0.43	19.2	0.39	24.31	0.50	16.78	0.43	20.84	0.44	19.16	0.45	18.41	0.59	12.06	0.47	20.46	0.45	21.14	0.45	21.56	0.35	22.81

Table A.4: (3/3) The left column under each tracker name contains the area under the curve of **success** for each video sequence, while the right column shows the position in the ranking with respect to the rest of the trackers.

Datasets	BSBT		Frag		KMS		SMS		Struck		MIL		LSK		VTS		VTD		CXT		SCT4		CNN-SVM	
david2	0.52	25	0.24	33	0.35	31	0.08	34	0.85	4	0.45	28	0.50	27	0.67	19	0.68	18	0.87	2	0.79	12	0.81	9
sylvester	0.57	21	0.58	20	0.48	29	0.09	36	0.71	3	0.53	26	0.23	35	0.62	15	0.61	17	0.59	18	0.62	13	0.74	1
mhyang	0.60	27	0.64	24	0.52	29	0.50	31	0.80	7	0.51	30	0.82	5	0.75	18	0.72	20	0.83	4	0.79	11	0.78	15
david	0.39	24	0.17	34	0.38	25	0.24	31	0.25	29	0.43	19	0.56	8	0.58	7	0.56	9	0.64	5	0.54	11	0.56	10
trellis	0.28	26	0.29	25	0.31	23	0.23	31	0.61	11	0.25	30	0.67	4	0.49	13	0.45	16	0.65	5	0.62	8	0.63	6
fish	0.59	17	0.54	19	0.39	25	0.13	35	0.84	1	0.45	24	0.31	27	0.68	16	0.55	18	0.77	9	0.80	6	0.82	4
carDark	0.43	23	0.30	30	0.36	28	0.02	35	0.87	1	0.20	31	0.82	6	0.74	13	0.53	21	0.56	20	0.64	16	0.72	14
matrix	0.17	12	0.06	31	0.18	11	0.08	25	0.10	24	0.13	17.5	0.30	2	0.10	22	0.12	19	0.07	26	0.29	4	0.10	23
ironman	0.10	15	0.03	36	0.05	29	0.08	20	0.09	17	0.07	22	0.11	14	0.18	3	0.21	2	0.05	27	0.11	10	0.06	24
soccer	0.10	33	0.17	19	0.19	14	0.13	29	0.19	13	0.17	18	0.10	32	0.33	4	0.33	3	0.15	27	0.46	1	0.18	17
deer	0.39	19	0.18	23	0.41	17	0.10	27	0.73	3	0.13	25	0.27	20	0.05	32	0.06	31	0.69	7	0.67	8	0.73	4
singer1	0.21	33	0.21	32	0.32	28	0.54	7	0.36	13	0.36	16	0.35	24	0.49	10	0.49	9	0.49	8	0.36	15	0.37	12
singer2	0.21	17	0.20	18	0.22	16	0.06	27	0.04	33	0.51	8	0.09	24	0.33	10	0.42	9	0.07	26	0.71	4	0.70	5
skating1	0.16	24	0.13	29	0.31	15	0.03	36	0.31	16	0.13	28	0.47	9	0.51	3	0.52	2	0.14	25	0.50	4	0.42	12
coke	0.18	25	0.04	34	0.19	22	0.12	31	0.66	1	0.21	19	0.20	21	0.18	24	0.15	28	0.42	10	0.63	3	0.60	5
woman	0.19	17	0.15	26	0.57	11	0.06	36	0.72	4	0.15	21	0.15	22	0.13	30	0.14	28	0.20	15	0.73	3	0.73	2
walking	0.13	34	0.54	17	0.52	22	0.47	25	0.57	11	0.54	14	0.45	26	0.61	9	0.60	10	0.17	33	0.53	19	0.52	23
couple	0.11	22	0.56	4	0.06	32	0.22	19	0.53	6	0.49	8	0.07	25	0.06	33	0.06	30	0.47	10	0.50	7	0.63	2
football1	0.30	30	0.37	27	0.38	25	0.34	28	0.66	7	0.65	8	0.32	29	0.44	23	0.57	13	0.75	3	0.78	2	0.66	6
doll	0.21	34	0.53	20	0.49	22	0.47	23	0.54	16	0.47	24	0.78	4	0.64	10	0.64	11	0.73	5	0.56	14	0.54	17
girl	0.41	24	0.45	21	0.11	36	0.51	18	0.73	1	0.40	25	0.30	31	0.49	19	0.55	16	0.55	15	0.67	9	0.72	3
boy	0.54	24	0.38	31	0.70	12	0.64	19	0.75	9	0.49	28	0.79	2	0.63	20	0.62	21	0.54	25	0.77	6	0.76	7
dudek	0.69	19	0.53	28	0.53	29	0.25	35	0.72	9	0.70	18	0.72	10	0.80	1	0.79	3	0.72	13	0.72	14	0.72	11
crossing	0.14	35	0.31	28	0.53	19	0.32	25.5	0.67	14	0.72	4	0.13	36	0.29	30	0.32	25.5	0.36	24	0.71	6	0.69	9
walking2	0.27	34	0.27	30	0.27	32	0.31	24	0.51	7	0.29	29	0.46	12	0.33	20	0.32	23	0.37	18	0.38	16	0.51	8
fleetface	0.48	23	0.46	25	0.22	32	0.11	36	0.60	6	0.49	21	0.50	20	0.62	3	0.62	4	0.56	15	0.59	10	0.64	2
freeman1	0.19	30	0.38	10	0.19	29	0.05	34	0.34	17	0.35	16	0.25	22	0.36	13	0.35	15	0.34	18	0.39	9	0.41	6
freeman3	0.21	27	0.32	15	0.17	28	0.35	10	0.26	24.5	0.01	35	0.32	17	0.30	22	0.30	20	0.70	3	0.33	14	0.31	19
david3	0.36	26	0.66	7	0.65	9	0.14	31	0.29	29	0.53	15	0.36	25	0.53	14	0.40	22	0.12	32	0.77	1	0.73	3
carScale	0.39	31	0.43	17	0.33	33	0.49	10	0.42	23	0.41	24	0.57	7	0.43	14	0.43	15	0.67	1	0.42	20	0.42	21
dog1	0.53	27	0.54	23	0.39	34	0.13	36	0.54	20	0.53	26	0.70	7	0.60	13	0.59	14	0.78	2	0.55	16	0.53	24
suv	0.58	14	0.62	11	0.42	27	0.09	34	0.51	19	0.21	31	0.60	13	0.48	23	0.45	25	0.72	5	0.86	2	0.71	6
motorRolling	0.16	4	0.12	12	0.12	15	0.11	22	0.16	6.5	0.13	11	0.11	21	0.11	20	0.12	16	0.14	8	0.09	32	0.55	2
mountainBike	0.31	26	0.13	33	0.48	23	0.12	34	0.70	10	0.45	24	0.64	17	0.69	12	0.69	11	0.22	28	0.68	13	0.72	4
faceocc1	0.76	6	0.80	1	0.71	15	0.58	31	0.72	11	0.59	29	0.48	34	0.66	22	0.68	21	0.63	27	0.78	5	0.65	23
football	0.29	29	0.69	2	0.08	33	0.02	36	0.53	22	0.58	11	0.53	21	0.58	12	0.56	14	0.54	20	0.68	4	0.70	1
subway	0.17	25.5	0.46	17	0.15	33	0.18	23	0.65	9	0.64	11	0.66	8	0.17	25.5	0.15	32	0.17	24	0.76	2	0.71	5
freeman4	0.15	22	0.14	23	0.03	33	0.03	35	0.17	16	0.06	31	0.15	20	0.11	28	0.16	19	0.17	15	0.36	4	0.36	3
skiing	0.04	30	0.03	32.5	0.06	21	0.16	4	0.04	31	0.06	22.5	0.03	32.5	0.03	34	0.07	17	0.09	5	0.05	28	0.44	1
faceocc2	0.63	21	0.64	19	0.46	29	0.10	36	0.77	1	0.66	18	0.62	22	0.73	7	0.73	10	0.73	9	0.74	5	0.76	3
tiger1	0.23	26	0.27	23	0.39	14	0.39	15	0.15	28	0.12	32	0.46	9	0.11	35	0.12	33	0.33	19	0.76	1	0.57	4
tiger2	0.16	27	0.12	33	0.20	22	0.20	23	0.54	6	0.46	8	0.36	12	0.31	14.5	0.31	14.5	0.36	11	0.61	2	0.57	3
lemming	0.31	27	0.30	28	0.64	5	0.66	3	0.48	17	0.64	4	0.63	7	0.45	18	0.43	20	0.45	19	0.50	16	0.56	12
liquor	0.57	8	0.32	21	0.57	7	0.51	11	0.40	20	0.22	30	0.16	36	0.43	18	0.49	13	0.25	25	0.69	6	0.80	3
basketball	0.13	29	0.61	10	0.42	18	0.39	19	0.21	26	0.22	25	0.33	22	0.71	4	0.71	3	0.02	35	0.77	2	0.50	14
jumping	0.15	20	0.66	4	0.11	25	0.08	30	0.61	8	0.52	13	0.07	32	0.15	19	0.13	22	0.52	12	0.70	1	0.65	6
jogging-2	0.63	9	0.47	12	0.20	19	0.46	14	0.20	18	0.13	25	0.34	16	0.13	30	0.13	29	0.13	27.5	0.77	1	0.76	2
bolt	0.17	14	0.14	18	0.15	16	0.14	17	0.01	28	0.01	32.5	0.49	7	0.17	13	0.37	11	0.02	27	0.74	2	0.80	1
car4	0.21	26	0.19	28.5	0.27	19	0.05	35	0.49	9	0.26	20	0.16	33	0.37	15	0.37	16	0.31	18	0.49	10	0.50	7
shaking	0.12	23	0.08	26	0.23	19	0.03	35	0.36	17	0.43	15	0.46	13	0.70	3	0.70	2	0.13	21	0.67	7	0.71	1
Avg*	0.38	23.29	0.43	21.8	0.41	22.8	0.30	26.03	0.54	13.64	0.41	21.01	0.50	18.57	0.52	16.72	0.52	16.42	0.52	16.13	0.63	8.7	0.62	8.5

Table A.5: (1/3) The left column under each tracker name contains the area under the curve of **precision** for each video sequence, while the right column shows the position in the ranking with respect to the rest of the trackers.

Datasets	DFN		CNT		KCF		TGPR		MEEM		VR-V		TM-V		RS-V		PD-V		MS-V		DFT		ORIA	
david2	1.00	11.5	1.00	11.5	1.00	11.5	1.00	11.5	1.00	11.5	0.66	29.5	0.88	24	1.00	11.5	0.66	29.5	0.12	35	0.80	26	1.00	11.5
sylvester	0.96	4	0.89	11	0.84	16	0.96	5	0.95	6.5	0.33	34	0.31	35	0.87	12	0.72	25	0.47	32	0.44	33	1.00	1
mhyang	1.00	18	1.00	8.5	1.00	8.5	0.95	20	1.00	8.5	0.53	32.5	0.78	24	0.77	26.5	0.53	32.5	0.19	36	0.77	26.5	1.00	8.5
david	0.94	13	0.47	23	1.00	4.5	0.98	10	0.99	9	0.22	33	0.22	32	0.65	19	0.95	12	0.18	35	0.31	29	0.50	20.5
trellis	0.80	13	1.00	2	1.00	2	0.98	5	0.96	8	0.37	25	0.10	36	0.42	21	0.44	18.5	0.43	20	0.51	15	0.39	23.5
fish	1.00	6	1.00	6	1.00	6	0.97	13	1.00	6	0.47	22	0.34	25	0.19	29	0.46	23	0.09	32	0.86	16	0.54	20
carDark	1.00	9	1.00	9	1.00	9	1.00	9	1.00	9	0.10	34	0.71	21	0.64	25.5	0.47	31	0.64	25.5	0.54	29	0.65	23
matrix	0.04	32	0.16	18.5	0.17	17	0.39	5	0.65	1	0.01	35.5	0.06	29	0.33	9	0.36	6	0.09	25.5	0.06	29	0.30	10
ironman	0.05	30	0.19	6	0.22	4.5	0.22	4.5	0.55	1	0.04	31.5	0.08	25	0.04	34	0.10	19.5	0.10	22	0.09	24	0.25	2.5
soccer	0.26	12	0.12	31	0.79	2	0.16	26	0.40	5	0.40	6	0.13	28	0.18	23.5	0.02	36	0.28	9	0.22	17	0.31	8
deer	0.42	18.5	1.00	3.5	0.82	12	0.86	11	1.00	3.5	0.07	27	0.62	15	0.42	18.5	0.17	24	0.04	30	0.31	21	0.03	34.5
singer1	0.41	23	1.00	3.5	0.81	16	0.68	17	0.35	26	0.30	28	0.27	30	0.36	25	0.13	36	0.16	35	0.51	20	0.99	8.5
singer2	0.50	7	0.95	2	0.95	3	0.97	1	0.04	24.5	0.05	21	0.03	33.5	0.04	24.5	0.01	35	0.09	16.5	0.60	6	0.09	16.5
skating1	0.62	13	1.00	3	1.00	1.5	0.81	7	0.69	11	0.16	28	0.28	20	0.29	18	0.15	29	0.09	35	0.19	25	0.46	15
coke	0.76	8	0.52	12	0.84	7	0.95	4	0.95	2.5	0.04	33	0.06	31.5	0.11	28	0.00	36	0.01	35	0.09	30	0.18	20
woman	0.20	25	0.94	8.5	0.94	8.5	0.97	4	1.00	1.5	0.12	35	0.20	28.5	0.50	14	0.83	11	0.18	33	0.95	5	0.23	18
walking	0.76	27	1.00	11	1.00	11	1.00	11	1.00	11	0.37	31.5	0.47	29	0.94	26	0.37	31.5	0.07	36	1.00	11	0.22	34
couple	0.61	13	0.23	21	0.26	20	0.60	15	1.00	2	0.11	27	0.44	19	0.09	33	0.09	33	0.11	27	0.09	33	0.19	22
football1	0.82	20	0.99	12	0.96	14	0.99	12	1.00	5.5	1.00	5.5	0.26	34	0.95	15	1.00	5.5	0.57	27.5	1.00	5.5	0.20	35
doll	0.91	19	0.95	13	0.97	10	0.94	16	0.98	4	0.29	33	0.56	31	0.85	22	0.22	35	0.82	25	0.43	32	0.99	1
girl	0.94	13	1.00	5.5	0.86	17	0.92	14.5	1.00	5.5	0.52	30	0.86	18	0.56	27	0.57	26	0.48	32	0.30	35	0.40	34
boy	0.92	23	1.00	6	1.00	6	0.99	16.5	1.00	6	1.00	12	1.00	6	0.99	15	0.34	34	0.99	16.5	0.49	30	0.18	36
dudek	0.86	9	0.87	7	0.88	6	0.75	18	0.80	15	0.46	31	0.66	24	0.35	34	0.18	36	0.41	33	0.66	23	0.67	22
crossing	1.00	10.5	1.00	10.5	1.00	10.5	1.00	10.5	1.00	10.5	1.00	10.5	1.00	10.5	1.00	10.5	1.00	10.5	0.59	28	0.68	24	0.48	29
walking2	0.73	10	1.00	2.5	0.44	16	0.99	5	0.37	32.5	0.38	31	0.40	26	0.38	29.5	0.54	12	0.37	32.5	0.40	25	0.46	15
fleetface	0.77	1	0.59	8	0.46	17	0.45	18.5	0.60	6	0.00	35.5	0.51	14.5	0.03	33	0.00	35.5	0.05	32	0.36	23.5	0.60	7
freeman1	0.87	13	0.96	5	0.39	27	0.93	10	0.93	11	0.13	35	0.98	1.5	0.66	19	0.13	35	0.45	23	0.94	8	0.40	25
freeman3	1.00	3	1.00	3	0.91	9	0.77	13	0.98	7	0.38	31	0.75	17	0.77	14.5	0.38	32	0.57	28	0.65	23	0.78	12
david3	0.90	8	0.65	16	1.00	1.5	1.00	3.5	0.99	5	0.29	29.5	0.48	22	0.87	9	0.29	29.5	0.13	34	0.75	12	0.17	32
carScale	0.68	16	0.72	10	0.81	3	0.79	4	0.64	25	0.73	9	0.30	35	0.58	29	0.77	6	0.07	36	0.65	19.5	0.98	1
dog1	0.99	14	0.94	19	1.00	4.5	1.00	4.5	0.99	12	0.63	30.5	0.54	34	0.72	29	0.63	30.5	0.52	35	0.62	32	1.00	9
suv	0.71	13	0.98	2	0.98	2	0.66	16	0.63	17	0.28	30.5	0.79	10	0.51	26	0.28	30.5	0.02	36	0.05	35	0.59	18
motorRolling	0.70	2	0.04	24.5	0.05	19	0.09	5	0.05	14	0.05	14	0.02	35	0.23	3	0.01	36	0.04	24.5	0.04	24.5	0.05	19
mountainBike	0.88	20	1.00	4.5	1.00	4.5	1.00	4.5	0.91	17	0.09	36	0.15	33.5	0.98	13	0.24	30	0.74	21	0.35	26	1.00	4.5
faceocc1	0.38	25	0.65	12	0.73	9	0.66	11	0.65	13	0.76	8	0.92	4	0.54	18	0.58	16	0.29	30	0.62	15	0.47	23
football	0.75	25.5	0.80	13	0.80	18	1.00	2.5	0.99	4	0.11	34	0.89	8	0.17	31.5	0.11	35	0.17	31.5	0.84	9	0.80	18
subway	0.55	18	1.00	4.5	1.00	4.5	1.00	4.5	1.00	4.5	0.72	15	0.24	28.5	0.71	16.5	0.71	16.5	0.07	36	1.00	4.5	0.26	22.5
freeman4	0.88	2	0.39	17	0.53	8	0.58	6	0.96	1	0.47	12	0.17	31.5	0.12	33	0.48	11	0.21	27.5	0.30	22	0.41	14.5
skiing	0.07	26	0.07	26	0.07	26	0.12	15.5	1.00	1.5	0.16	5	0.14	10	0.12	15.5	0.53	3	0.11	18.5	0.07	26	0.12	15.5
faceocc2	0.81	16	0.65	26	0.97	9	0.47	29	0.87	12	0.01	35	0.77	19	0.02	34	0.01	36	0.02	33	1.00	3	1.00	3
tiger1	0.42	10	0.15	30	0.85	2	0.28	19	0.81	3	0.35	15	0.64	5	0.27	21	0.49	8	0.27	20	0.67	4	0.17	27
tiger2	0.41	9	0.11	29.5	0.36	13	0.72	3	0.50	7	0.11	31.5	0.65	4	0.12	28	0.06	35	0.06	36	0.80	1	0.16	19
lemming	0.78	10	0.38	28	0.49	22	0.35	29	0.93	1	0.07	35	0.48	23	0.66	15	0.54	19	0.82	8.5	0.52	20	0.02	36
liquor	0.85	5	0.48	13	0.98	1	0.27	24	0.97	2	0.46	14	0.35	22	0.52	7	0.16	35	0.40	18	0.22	27	0.20	32
basketball	0.89	9	0.07	31	0.92	8	0.99	6	1.00	5	0.65	14.5	0.16	28	0.63	16	0.65	14.5	0.58	20	0.89	10	0.03	34
jumping	0.49	14	1.00	3.5	0.34	17	0.95	13	1.00	3.5	0.30	18	0.96	12	0.12	31	0.96	11	0.46	15	0.13	29.5	0.13	29.5
jogging-2	0.98	6	1.00	2.5	0.16	31.5	1.00	5	0.97	7	0.18	26.5	0.15	35	0.16	34	0.19	20	0.13	36	0.16	31.5	0.79	13
bolt	0.87	9	0.98	4	0.99	3	0.02	31	0.97	6.5	0.14	20.5	0.02	31	0.63	10	0.14	20.5	0.30	17	0.04	24	0.32	13.5
car4	1.00	3.5	1.00	3.5	0.95	10	1.00	3.5	0.83	12	0.25	26	0.22	30	0.09	33	0.26	25	0.21	31	0.26	24	0.23	27.5
shaking	0.96	4	0.01	32.5	0.02	28.5	0.97	2	0.96	3	0.00	36	0.51	12	0.05	24.5	0.02	30.5	0.02	30.5	0.83	8	0.64	10
Avg*	0.81	13.39	0.77	12.31	0.84	10.75	0.78	11.09	0.88	8.53	0.35	25.27	0.51	22.81	0.55	21.89	0.37	24.57	0.39	27.84	0.51	20.64	0.58	18.69

Table A.6: (2/3) The left column under each tracker name contains the area under the curve of **precision** for each video sequence, while the right column shows the position in the ranking with respect to the rest of the trackers.

Datasets	ASLA		L1APG		CT		TLD		IVT		MTT		CSK		SCM		LOT		CPF		OAB		SemiT	
david2	1.00	11.5	1.00	11.5	0.00	36	1.00	11.5	1.00	11.5	1.00	11.5	1.00	11.5	1.00	11.5	1.00	11.5	1.00	11.5	0.54	32	0.84	25
sylvester	0.82	18	0.52	31	0.90	10	0.95	6.5	0.68	27	0.85	14.5	0.91	9	0.95	8	0.81	21	0.86	13	0.74	24	0.65	30
mhyang	1.00	8.5	1.00	17	0.82	22	0.98	19	1.00	8.5	1.00	8.5	1.00	8.5	1.00	8.5	0.26	35	0.79	23	0.94	21	0.78	25
david	1.00	4.5	0.80	16	0.82	15	1.00	4.5	1.00	4.5	0.33	27	0.50	20.5	1.00	4.5	0.35	26	0.19	34	0.38	25	0.30	30
trellis	0.86	11	0.18	35	0.39	23.5	0.53	14	0.33	27	0.22	32	0.81	12	0.87	10	0.30	29	0.30	28	0.19	33	0.19	34
fish	1.00	6	0.05	33	0.88	14	1.00	6	1.00	6	0.04	34.5	0.04	34.5	0.86	15	0.27	28	0.11	31	0.04	36	0.13	30
carDark	1.00	9	1.00	9	0.01	36	0.64	24	0.81	18	1.00	9	1.00	9	1.00	9	0.62	28	0.17	33	1.00	9	1.00	9
matrix	0.05	31	0.12	22.5	0.02	33.5	0.16	18.5	0.02	33.5	0.34	8	0.01	35.5	0.35	7	0.15	20	0.09	25.5	0.40	4	0.21	13
ironman	0.13	12.5	0.11	17	0.10	22	0.12	14	0.05	28.5	0.11	17	0.13	12.5	0.16	8	0.10	19.5	0.05	28.5	0.03	36	0.04	34
soccer	0.12	29	0.21	20	0.22	19	0.11	32	0.17	25	0.18	23.5	0.14	27	0.27	10	0.33	7	0.26	11	0.09	34	0.08	35
deer	0.03	34.5	0.72	14	0.04	30	0.73	13	0.03	34.5	0.89	9	1.00	3.5	0.03	34.5	0.25	22	0.04	30	0.96	7	0.87	10
singer1	1.00	3.5	0.38	24	0.84	15	1.00	3.5	0.96	11	0.34	27	0.67	18	1.00	3.5	0.24	32	0.99	8.5	0.89	14	0.17	34
singer2	0.04	29	0.04	29	0.01	36	0.07	18	0.04	29	0.04	29	0.04	29	0.11	14	0.19	11	0.12	13	0.03	33.5	0.05	22
skating1	0.77	9	0.13	31	0.09	34	0.32	17	0.11	33	0.14	30	0.99	4	0.77	8	0.29	19	0.23	24	0.69	12	0.26	22
coke	0.16	21	0.26	15	0.11	27	0.68	9	0.13	25.5	0.66	10	0.87	6	0.43	13	0.13	25.5	0.39	14	0.22	17	0.19	19
woman	0.20	25	0.20	21.5	0.20	21.5	0.19	31.5	0.20	27	0.20	21.5	0.25	17	0.94	6.5	0.16	34	0.20	30	0.68	12	0.54	13
walking	1.00	11	1.00	11	1.00	11	0.96	25	1.00	11	1.00	11	1.00	11	1.00	11	1.00	11	1.00	11	1.00	11	0.41	30
couple	0.09	33	0.61	14	0.69	8	1.00	2	0.09	33	0.64	10	0.09	33	0.11	24.5	0.63	12	0.87	5	0.47	18	0.51	17
football1	0.80	22	0.84	19	0.35	33	0.55	29	0.81	21	0.77	23	0.76	24.5	0.57	27.5	1.00	5.5	1.00	5.5	0.38	32	0.16	36
doll	0.92	17	0.97	8	0.68	28	0.98	5	0.76	26	0.61	29	0.58	30	0.98	7	0.95	11	0.94	15	0.87	21	0.16	36
girl	1.00	5.5	1.00	5.5	0.61	25	0.92	14.5	0.44	33	1.00	5.5	0.55	28	1.00	5.5	0.64	23	0.74	20	1.00	5.5	0.66	22
boy	0.44	32.5	0.93	22	0.93	21	1.00	6	0.33	35	0.71	26	0.84	25	0.44	32.5	0.67	28	1.00	13	1.00	6	0.51	29
dudek	0.75	17	0.70	19	0.42	32	0.60	25	0.89	3	0.78	16	0.81	14	0.88	5	0.54	28	0.57	26	0.68	21	0.56	27
crossing	1.00	10.5	0.25	35	1.00	10.5	0.62	27	1.00	10.5	0.26	34	1.00	10.5	1.00	10.5	0.63	25	0.89	23	0.98	21.5	0.98	21.5
walking2	0.40	24	0.98	7	0.43	17	0.43	18	1.00	2.5	1.00	2.5	0.47	14	1.00	2.5	0.39	27	0.36	34	0.49	13	0.89	8
fleetface	0.30	28	0.41	22	0.44	21	0.51	14.5	0.26	29	0.45	18.5	0.57	10	0.53	13	0.55	11	0.16	30	0.44	20	0.35	25
freeman1	0.39	28	0.38	29	0.40	26	0.54	21	0.81	14	0.37	30	0.56	20	0.98	3	0.46	22	0.76	16	0.74	17	0.30	32
freeman3	1.00	3	0.75	18	0.21	34	0.77	14.5	0.76	16	0.92	8	0.57	27	1.00	3	0.59	26	0.17	35	0.74	19	0.32	33
david3	0.55	20	0.46	24	0.41	25	0.11	35	0.75	11	0.11	36	0.66	15	0.50	21	0.99	6	0.57	18	0.39	26	0.18	31
carScale	0.74	7.5	0.65	22.5	0.72	11	0.85	2	0.78	5	0.65	22.5	0.65	19.5	0.65	22.5	0.48	34	0.67	17	0.66	18	0.59	28
dog1	1.00	10	1.00	4.5	0.95	18	1.00	4.5	0.98	16	1.00	4.5	1.00	4.5	0.98	17	0.99	13	0.91	23	0.99	15	0.80	28
suv	0.57	19	0.52	24.5	0.25	32	0.91	7	0.45	28	0.52	24.5	0.57	21	0.98	4	0.81	9	0.78	11	0.76	12	0.50	27
motorRolling	0.06	10	0.04	30	0.04	30	0.12	4	0.03	33.5	0.04	24.5	0.04	24.5	0.04	30	0.05	14	0.06	10	0.03	33.5	0.08	7
mountainBike	0.90	18	0.95	15	0.18	32	0.26	29	1.00	10	1.00	4.5	1.00	4.5	0.97	14	0.71	22	0.15	33.5	0.99	12	0.60	25
faceocc1	0.18	35	0.68	10	0.33	27	0.20	34	0.64	14	0.41	24	0.95	2	0.93	3	0.26	31	0.32	28	0.25	32	0.84	7
football	0.73	27	0.80	18	0.80	13	0.80	10	0.79	22	0.80	18	0.80	13	0.77	24	1.00	2.5	0.97	6	0.43	28	0.28	30
subway	0.23	32	0.25	24.5	0.99	10	0.25	24.5	0.22	33.5	0.10	35	0.24	28.5	1.00	4.5	0.98	11.5	0.22	33.5	0.24	28.5	0.41	19
freeman4	0.22	24.5	0.63	5	0.06	36	0.41	14.5	0.35	20	0.51	9.5	0.19	30	0.51	9.5	0.56	7	0.12	34	0.17	31.5	0.40	16
skiing	0.14	10	0.09	21.5	0.09	21.5	0.12	15.5	0.11	18.5	0.14	10	0.10	20	0.14	10	0.02	36	0.06	31	0.14	10	0.06	31
faceocc2	0.79	17	0.78	18	0.68	24	0.86	15	0.99	7	0.86	14	1.00	3	0.86	13	0.64	27	0.40	30	0.71	21	0.52	28
tiger1	0.23	23	0.34	16	0.21	24	0.46	9	0.08	36	0.17	27	0.26	22	0.13	31	0.16	29	0.39	13	0.09	35	0.55	7
tiger2	0.14	23	0.27	15	0.36	11	0.39	10	0.08	34	0.25	16	0.11	31.5	0.11	29.5	0.18	18	0.11	33	0.14	24	0.24	17
lemming	0.17	32	0.17	30	0.68	14	0.86	4	0.17	33	0.39	27	0.44	24	0.17	34	0.83	5	0.88	2	0.70	12	0.17	31
liquor	0.23	25	0.21	31	0.21	28.5	0.59	6	0.21	30	0.20	33	0.22	26	0.28	23	0.94	3	0.52	9	0.41	17	0.44	15
basketball	0.60	19	0.31	24	0.30	25	0.03	35	0.50	21	0.29	26	1.00	2.5	0.66	13	1.00	2.5	0.74	12	0.02	36	0.05	32
jumping	0.45	16	0.27	19	0.10	33	1.00	3.5	0.21	22	0.16	26.5	0.05	36	0.15	28	0.99	10	0.16	26.5	0.08	34	0.07	35
jogging-2	0.18	26.5	0.19	23	0.17	29	0.86	11	0.20	19	0.17	28	0.19	23	1.00	2.5	0.16	31.5	0.84	12	0.58	14	0.92	9
bolt	0.02	31	0.02	31	0.01	36	0.31	15.5	0.01	34.5	0.02	31	0.03	25	0.03	26	0.97	6.5	0.91	8	0.05	23	0.21	19
car4	1.00	3.5	0.30	20	0.28	22	0.87	11	1.00	3.5	0.36	16	0.36	17	0.97	8	0.05	36	0.14	32	0.28	23	0.35	19
shaking	0.48	13	0.04	26	0.05	24.5	0.41	15	0.01	34	0.01	32.5	0.56	11	0.81	9	0.14	20	0.17	18	0.01	35	0.10	22
Avg*	0.64	18.73	0.59	20.17	0.50	23.75	0.71	15.17	0.59	21.38	0.55	20.31	0.62	18.16	0.75	13.85	0.63	19.45	0.60	20.64	0.60	21.5	0.42	23.69

Table A.7: (3/3) The left column under each tracker name contains the area under the curve of **precision** for each video sequence, while the right column shows the position in the ranking with respect to the rest of the trackers.

Datasets	BSBT		Frag		KMS		SMS		Struck		MIL		LSK		VTS		VTD		CXT		SCT4		CNN-SVM	
david2	0.71	28	0.31	33	0.55	31	0.26	34	1.00	11.5	0.98	23	0.74	27	1.00	11.5	1.00	11.5	1.00	11.5	1.00	11.5	1.00	11.5
sylvester	0.75	23	0.72	26	0.68	28	0.76	22	0.99	3	0.65	29	0.29	36	0.82	19	0.82	20	0.85	14.5	0.84	17	1.00	2
mhyang	0.74	30	0.74	29	0.60	31	0.76	28	1.00	8.5	0.46	34	1.00	8.5	1.00	8.5	1.00	8.5	1.00	8.5	1.00	8.5	1.00	8.5
david	0.50	22	0.17	36	0.45	24	0.25	31	0.33	28	0.70	17.5	0.70	17.5	0.96	11	0.94	14	1.00	4.5	1.00	4.5	1.00	4.5
trellis	0.25	30	0.40	22	0.34	26	0.44	18.5	0.88	9	0.23	31	0.97	7	0.50	16	0.50	17	0.97	6	0.99	4	1.00	2
fish	0.70	17	0.55	19	0.30	27	0.50	21	1.00	6	0.39	24	0.33	26	0.99	12	0.65	18	1.00	6	1.00	6	1.00	6
carDark	0.69	22	0.52	30	0.63	27	0.08	35	1.00	9	0.38	32	1.00	9	1.00	9	0.74	19	0.73	20	1.00	9	1.00	9
matrix	0.20	14.5	0.07	27	0.26	11	0.10	24	0.12	22.5	0.18	16	0.51	2	0.20	14.5	0.22	12	0.06	29	0.44	3	0.13	21
ironman	0.14	11	0.04	34	0.07	27	0.10	22	0.11	15	0.11	17	0.15	9	0.25	2.5	0.17	7	0.04	31.5	0.14	10	0.07	26
soccer	0.09	33	0.20	21	0.22	17	0.22	17	0.25	13	0.19	22	0.12	30	0.51	3	0.45	4	0.23	15	0.89	1	0.24	14
deer	0.46	17	0.21	23	0.54	16	0.14	25	1.00	3.5	0.13	26	0.34	20	0.04	30	0.04	30	1.00	3.5	0.90	8	1.00	3.5
singer1	0.21	33	0.25	31	0.29	29	1.00	7	0.64	19	0.50	21	0.48	22	1.00	3.5	1.00	3.5	0.97	10	0.91	13	0.95	12
singer2	0.06	20	0.18	12	0.10	15	0.04	29	0.04	29	0.40	9	0.04	23	0.36	10	0.45	8	0.06	19	0.90	4	0.83	5
skating1	0.18	26.5	0.18	26.5	0.26	21	0.08	36	0.47	14	0.13	32	0.70	10	0.89	6	0.90	5	0.24	23	1.00	1.5	0.44	16
coke	0.14	24	0.03	34	0.10	29	0.06	31.5	0.95	2.5	0.15	22	0.26	16	0.19	18	0.15	23	0.65	11	0.95	1	0.93	5
woman	0.28	16	0.19	31.5	0.83	10	0.09	36	1.00	1.5	0.21	19	0.20	21.5	0.20	28.5	0.20	25	0.37	15	0.94	6.5	1.00	3
walking	0.18	35	0.99	24	0.99	23	1.00	22	1.00	11	1.00	11	0.66	28	1.00	11	1.00	11	0.24	33	1.00	11	1.00	11
couple	0.17	23	0.90	4	0.11	27	0.58	16	0.74	6	0.68	9	0.09	33	0.10	29	0.11	24.5	0.64	11	0.72	7	1.00	2
football1	0.42	31	0.58	26	0.76	24.5	0.93	16	1.00	5.5	1.00	5.5	0.47	30	0.89	17	0.99	12	1.00	5.5	1.00	5.5	0.86	18
doll	0.27	34	0.88	20	0.83	24	0.84	23	0.92	18	0.73	27	0.99	2	0.95	14	0.97	9	0.99	3	0.98	6	0.95	12
girl	0.53	29	0.63	24	0.17	36	0.95	11	1.00	5.5	0.71	21	0.49	31	0.87	16	0.95	12	0.77	19	1.00	5.5	1.00	5.5
boy	0.68	27	0.48	31	0.99	14	1.00	6	1.00	6	0.85	24	1.00	6	0.98	18	0.97	19	0.94	20	1.00	6	1.00	6
dudek	0.85	10	0.50	29.5	0.50	29.5	0.22	35	0.90	2	0.69	20	0.82	13	0.87	8	0.88	4	0.82	12	0.84	11	0.90	1
crossing	0.33	33	0.40	32	1.00	10.5	1.00	10.5	1.00	10.5	1.00	10.5	0.13	36	0.42	31	0.44	30	0.63	26	1.00	10.5	1.00	10.5
walking2	0.32	36	0.34	35	0.38	29.5	0.38	28	0.98	6	0.41	22.5	0.67	11	0.41	20	0.41	20	0.41	22.5	0.41	20	0.86	9
fleetface	0.54	12	0.34	26	0.07	31	0.01	34	0.64	5	0.36	23.5	0.31	27	0.64	4	0.66	3	0.57	9	0.49	16	0.67	2
freeman1	0.28	33	0.90	12	0.44	24	0.13	35	0.80	15	0.94	9	0.37	31	0.97	4	0.95	6.5	0.73	18	0.95	6.5	0.98	1.5
freeman3	0.39	30	0.64	24	0.53	29	0.73	20	0.79	11	0.05	36	0.64	25	0.70	22	0.72	21	1.00	3	0.83	10	0.98	6
david3	0.37	27	0.77	10	0.98	7	0.64	17	0.34	28	0.74	14	0.48	23	0.74	13	0.56	19	0.15	33	1.00	1.5	1.00	3.5
carScale	0.52	33	0.69	15	0.55	30.5	0.62	27	0.65	22.5	0.63	26	0.69	14	0.54	32	0.55	30.5	0.74	7.5	0.71	12	0.70	13
dog1	0.83	26	0.84	24	0.61	33	0.14	36	1.00	11	0.92	21.5	0.92	20	0.81	27	0.83	25	1.00	4.5	1.00	4.5	0.92	21.5
suv	0.71	15	0.71	14	0.33	29	0.14	33	0.57	20	0.12	34	0.82	8	0.54	22	0.52	23	0.91	6	0.98	2	0.94	5
motorRolling	0.05	19	0.06	10	0.05	14	0.07	8	0.09	6	0.04	24.5	0.05	14	0.05	19	0.05	19	0.04	30	0.04	30	0.88	1
mountainBike	0.34	27	0.15	35	0.68	23	0.23	31	0.92	16	0.67	24	0.89	19	1.00	10	1.00	10	0.28	28	1.00	4.5	1.00	4.5
faceocc1	0.85	6	0.98	1	0.53	20	0.51	21	0.58	17	0.22	33	0.12	36	0.49	22	0.53	19	0.34	26	0.86	5	0.31	29
football	0.35	29	0.99	5	0.16	33	0.01	36	0.75	25.5	0.79	23	0.80	11	0.80	18	0.80	18	0.80	18	1.00	1	0.96	7
subway	0.25	26	0.74	14	0.30	20	0.29	21	0.98	11.5	0.99	9	0.98	13	0.24	28.5	0.23	31	0.26	22.5	1.00	4.5	1.00	4.5
freeman4	0.31	21	0.21	27.5	0.22	26	0.08	35	0.37	18	0.20	29	0.25	23	0.22	24.5	0.37	19	0.43	13	0.71	4	0.85	3
skiing	0.07	26	0.04	34.5	0.14	10	0.38	4	0.04	34.5	0.07	26	0.05	33	0.06	31	0.14	10	0.15	6	0.07	26	1.00	1.5
faceocc2	0.68	23	0.69	22	0.36	31	0.28	32	1.00	3	0.74	20	0.66	25	0.94	11	0.98	8	1.00	3	0.95	10	1.00	6
tiger1	0.19	25	0.29	18	0.32	17	0.42	11.5	0.17	27	0.09	34	0.42	11.5	0.12	32.5	0.12	32.5	0.37	14	0.98	1	0.56	6
tiger2	0.13	26	0.13	27	0.14	25	0.15	22	0.63	5	0.41	8	0.36	12	0.16	20.5	0.16	20.5	0.34	14	0.73	2	0.58	6
lemming	0.40	26	0.40	25	0.82	8.5	0.86	3	0.63	17	0.82	6	0.82	7	0.55	18	0.51	21	0.73	11	0.64	16	0.69	13
liquor	0.44	16	0.35	21	0.49	12	0.50	10	0.39	19	0.20	34	0.12	36	0.36	20	0.52	8	0.21	28.5	0.49	11	0.93	4
basketball	0.15	29	0.77	11	0.61	18	0.63	17	0.12	30	0.28	27	0.46	22	1.00	2.5	1.00	2.5	0.04	33	0.92	7	0.34	23
jumping	0.19	24.5	1.00	8	0.20	23	0.19	24.5	1.00	3.5	1.00	8	0.11	32	0.24	20	0.21	21	1.00	8	1.00	3.5	1.00	3.5
jogging-2	0.86	10	0.56	15	0.21	18	0.96	8	0.25	17	0.19	23	0.54	16	0.19	23	0.19	23	0.16	31.5	1.00	2.5	1.00	2.5
bolt	0.32	13.5	0.28	18	0.34	12	0.46	11	0.02	28	0.01	34.5	0.98	5	0.09	22	0.31	15.5	0.03	27	1.00	1.5	1.00	1.5
car4	0.23	27.5	0.22	29	0.29	21	0.08	34.5	0.99	7	0.35	18	0.08	34.5	0.36	15	0.36	14	0.38	13	0.97	9	1.00	3.5
shaking	0.03	27	0.09	23	0.15	19	0.02	28.5	0.19	17	0.28	16	0.47	14	0.92	7	0.93	6	0.13	21	0.94	5	0.97	1
Avg*	0.46	24.05	0.57	22.59	0.52	22.42	0.48	22.89	0.74	13.6	0.52	21.72	0.61	19.73	0.69	16.71	0.70	15.86	0.68	16.17	0.87	7.74	0.87	7.94