Queensland University of Technology
Brisbane Australia

This may be the author's version of a work that was submitted/accepted for publication in the following source:

Choi, Jongwon, Chang, Hyung Jin, Yun, Sangdoo, Fischer, Tobias, Demiris, Yiannis, & Choi, Jin Young
(2017)
Attentional correlation filter network for adaptive visual tracking.
In *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*
Institute of Electrical and Electronics Engineers Inc., United States of America, pp. 4828-4837.

This file was downloaded from: https://eprints.qut.edu.au/196766/

© 2017 IEEE

2017 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.

License: Free-to-read at publisher

Notice: *Please note that this document may not be the Version of Record (i.e. published version) of the work. Author manuscript versions (as Submitted for peer review or as Accepted for publication after peer review) can be identified by an absence of publisher branding and/or typeset appearance. If there is any doubt, please refer to the published source.*

https://doi.org/10.1109/CVPR.2017.513

Preprint version; final version available at http://ieeexplore.ieee.org
IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
Published by: IEEE

Attentional Correlation Filter Network for Adaptive Visual Tracking

Jongwon Choi[1] Hyung Jin Chang[2] Sangdoo Yun[1] Tobias Fischer[2]
Yiannis Demiris[2] Jin Young Choi[1]
[1]ASRI, Dept. of Electrical and Computer Eng., Seoul National University, South Korea
[2]Personal Robotics Laboratory, Department of Electrical and Electronic Engineering
Imperial College London, United Kingdom
jwchoi.pil@gmail.com {hj.chang,t.fischer,y.demiris}@imperial.ac.uk {yunsd101,jychoi}@snu.ac.kr

Abstract

We propose a new tracking framework with an attentional mechanism that chooses a subset of the associated correlation filters for increased robustness and computational efficiency. The subset of filters is adaptively selected by a deep attentional network according to the dynamic properties of the tracking target. Our contributions are manifold, and are summarised as follows: (i) Introducing the Attentional Correlation Filter Network which allows adaptive tracking of dynamic targets. (ii) Utilising an attentional network which shifts the attention to the best candidate modules, as well as predicting the estimated accuracy of currently inactive modules. (iii) Enlarging the variety of correlation filters which cover target drift, blurriness, occlusion, scale changes, and flexible aspect ratio. (iv) Validating the robustness and efficiency of the attentional mechanism for visual tracking through a number of experiments. Our method achieves similar performance to non real-time trackers, and state-of-the-art performance amongst real-time trackers.

1. Introduction

Humans rely on various cues when observing and tracking objects, and the selection of attentional cues highly depends on knowledge-based expectation according to the dynamics of the current scene [13, 14, 28]. Similarly, in order to infer the accurate location of the target object, a tracker needs to take changes of several appearance (illumination change, blurriness, occlusion) and dynamic (expanding, shrinking, aspect ratio change) properties into account. Although visual tracking research has achieved remarkable advances in the past decades [21–23, 32, 38–40], and thanks to deep learning especially in the recent years [6, 8, 29, 35, 36, 41], most methods employ only a subset of these properties, or are too slow to perform in real-time.

The deep learning based approaches can be divided into two large groups. Firstly, online deep learning based trackers [29, 33, 35, 36, 41] which require frequent fine-tuning of the network to learn the appearance of the target. These approaches show high robustness and accuracy, but are too slow to be applied in real-world settings. Secondly, correlation filter based trackers [6, 8, 26, 30] utilising deep convolutional features have also shown state-of-the-art performance. Each correlation filter distinguishes the target from nearby outliers in the Fourier domain, which leads to high robustness even with small computational time. However, in order to cover more features and dynamics, more diverse correlation filters need to be added, which slows the overall tracker.

As previous deep-learning based trackers focus on the changes in the appearance properties of the target, only limited dynamic properties can be considered. Furthermore, updating the entire network for online deep learning based trackers is computationally demanding, although the deep network is only sparsely activated at any time [29, 33, 35, 36,

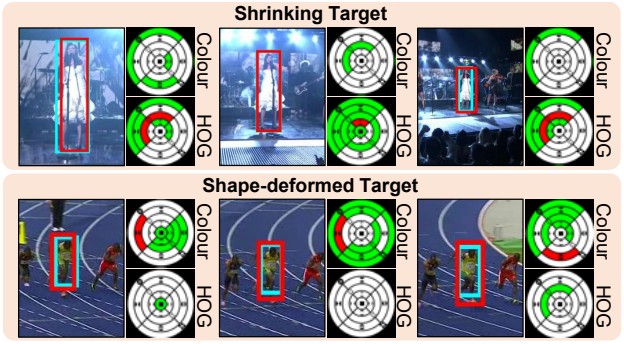


Figure 1. **Attentional Mechanism for Visual Tracking.** The tracking results of the proposed framework (red) are shown along with the ground truth (cyan). The circles represent the attention at that time, where one region represents one tracking module. When the target shrinks as in the first row, the attention is on modules with scale-down changes in the left-top region of the circles. If the target suffers from shape deformation as in the second row, modules with colour features are chosen because they are robust to shape deformation.

Preprint version; final version available at http://ieeexplore.ieee.org
IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
Published by: IEEE

41]. Similarly, for correlation filter based trackers, only some of the convolutional features are useful at a time [6, 8, 26, 30]. Therefore, by introducing an adaptive selection of attentional properties, additional dynamic properties can be considered for increased accuracy and robustness while keeping the computational time constant.

In this paper, we propose an attentional mechanism to adaptively select the best fitting subset of all available correlation filters, as shown in Fig. 1. This allows increasing the number of overall filters while simultaneously keeping the computational burden low. Furthermore, the robustness of the tracker increases due to the exploitation of previous experience, which allows concentrating on the expected appearance and dynamic changes and ignoring irrelevant properties of a scene. The importance of attentional mechanisms of the human visual system is highlighted by recent trends in neuroscience research [13, 14, 28], as well as by theoretical work on the computational aspects of visual attention [12, 34]. Similarly, we find that our framework using the attentional mechanism outperforms a network which utilises all module trackers while being significantly faster.

2. Related Research

Deep learning based trackers: Recent works based on online deep learning trackers have shown high performance [29, 33, 35, 36]. Wang *et al*. [35] proposed a framework which fuses shallow convolutional layers with deep convolutional layers to simultaneously consider detailed and contextual information of the target. Nam and Han [29] introduced a multi-domain convolutional neural network which determines the target location from a large set of candidate patches. Tao *et al*. [33] utilised a Siamese network to estimate the similarities between the previous target and the candidate patches. Wang *et al*. [36] proposed a sequential training method of convolutional neural networks for visual tracking, which utilises an ensemble strategy to avoid overfitting the network. However, because these trackers learn the appearance of the target, the networks require frequent fine-tuning which is slow and prohibits real-time tracking.

Correlation filter based trackers: Correlation filter based approaches have recently become increasingly popular due to the rapid speed of correlation filter calculations [2, 3, 5, 7, 16, 18, 27]. Henriques *et al*. [16] improved the performance of correlation filter based trackers by extending them to multi-channel inputs and kernel based training. Danelljan *et al*. [5] developed a correlation filter covering scale changes of the target, and Ma *et al*. [27] and Hong *et al*. [18] used the correlation filter as a short-term tracker with an additional long-term memory system. Choi *et al*. [3] proposed an integrated tracker with various correlation filters weighted by a spatially attentional weight map. Danelljan *et al*. [7] developed a regularised correlation filter which can extend the training region for the correlation filter by applying spatially regularised weights to suppress the background.

Deep learning + correlation filter trackers: Correlation filter based trackers show state-of-the-art performance when rich features such as deep convolution features are utilised [6, 8, 26, 30]. Danelljan *et al*. [6] extended the regularised correlation filter to use deep convolution features. Danelljan *et al*. [8] also proposed a novel correlation filter to find the target position in the continuous domain, while incorporating features of various resolutions. Their framework showed state-of-the-art performance with deep convolution features. Ma *et al*. [30] estimated the position of the target by fusing the response maps obtained from the convolution features of various resolutions in a coarse-to-fine scheme. Qi *et al*. [26] tracked the target by utilising an adaptive hedge algorithm applied to the response maps from deep convolution features. However, even though each correlation filter works fast, deep convolutional features have too many dimensions to be handled in real-time. Furthermore, to recognise scale changes of the target, correlation filter based algorithms need to train scale-wise filters, or apply the same filter repeatedly. As this increases the computation time significantly with deep convolutional features, several approaches do not consider scale changes, including the methods proposed by Ma *et al*. [30] and Qi *et al*. [26].

Adaptive module selection frameworks: In the action recognition area, Spatio-Temporal Attention RElocation (STARE) [25] provides an information-theoretic approach for attending the activity with the highest uncertainty amongst multiple activities. However, STARE determined the attention following a predefined strategy. In the robotics area, Hierarchical, Attentive, Multiple Models for Execution and Recognition (HAMMER) [9–11] is selecting modules based on their prediction error. A method to predict the performance of thousands of different robotic behaviours based on a pre-computed performance map was presented in [4]. In our framework, we focus on enhancing the ability to adapt to dynamic changes.

3. Methodology

The overall scheme of the proposed Attentional Correlation Filter Network (ACFN) is depicted in Fig. 2. ACFN consists of two networks: the correlation filter network and the attention network. The correlation filter network has a lot of tracking modules, which estimate the validation scores as their precisions. In the attention network, the prediction sub-network predicts the validation scores of all modules for the current frame, and the selection sub-network selects the active modules based on the predicted scores. The active module with the highest estimated validation score, the best module, is used to determine the position and scale of the target. Finally, the estimated validation scores of the active modules are used along with the predicted validation scores of the inactive modules to generate the final validation scores

Preprint version; final version available at http://ieeexplore.ieee.org
IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
Published by: IEEE

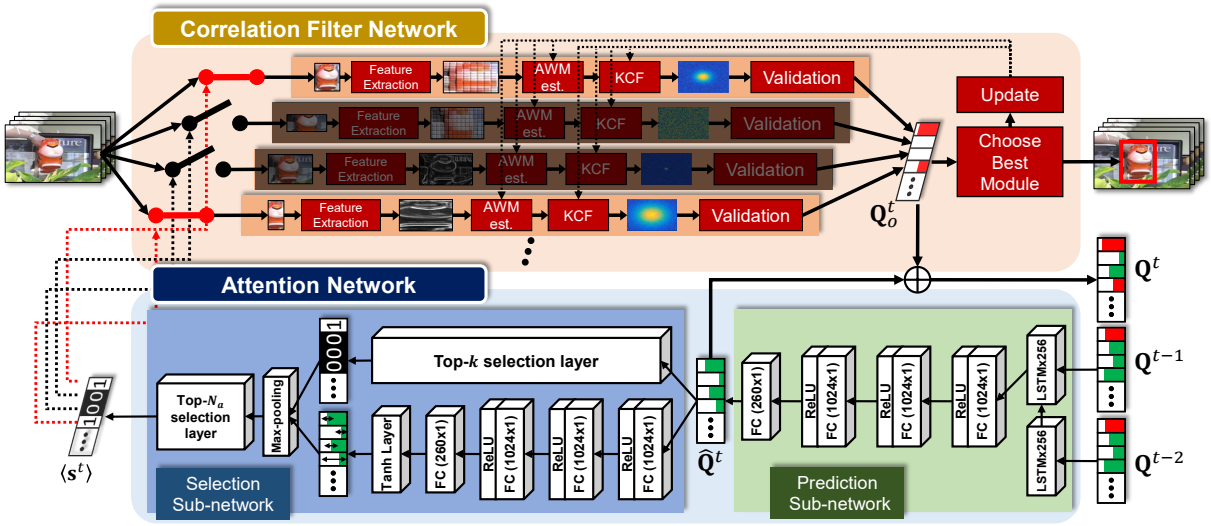


Figure 2. **Proposed Algorithm Scheme.** The proposed framework consists of the correlation filter network and the attention network. According to the results from the attention network obtained by the previous validation scores, an adaptive subset of tracking modules in the correlation filter network is operated. The target is determined based on the tracking module with the best validation score among the subset, and the tracking modules are updated accordingly.

used for the next frames. Every tracking module is updated according to the tracking result of the best module. As the attention network learns the general expectation of dynamic targets rather than scene or target specific properties, it can be pre-trained and does not need updating while tracking.

3.1. Correlation Filter Network

The correlation filter network incorporates a large variety of tracking modules, each covering a specific appearance or dynamic change, including changes due to blurring, structural deformation, scale change, and occlusion. The attentional feature-based correlation filter (AtCF) [3] is utilised as tracking module, which is composed of the attentional weight map (AWM) and the Kernelized Correlation Filter (KCF) [16]. As only a subset of all tracking modules is active at a time, we can increase the total number of modules which allows us to consider novel property types: variable aspect ratios and the delayed update for drifting targets.

3.1.1 Tracking module types

The tracking modules are based on a combination of four different types of target properties and dynamics: two feature types, two kernel types, thirteen relative scale changes, and five steps of the delayed update. Thus, the correlation filter network contains 260 ($2\times2\times13\times5$) different tracking modules in total.

Feature types: We utilise two feature types: a colour feature and a histogram of oriented gradients (HOG) feature. We discretise the tracking box into a grid with cell size $N_g \times N_g$. For colour images, we build a 6-dimensional colour feature vector by averaging the R, G, B and L, a, b values in the RGB and *Lab* space respectively. For grey images, we average the intensity and Laplacian values along the x and y-direction as a 3-dimensional colour feature vector. The HOG feature of a cell is extracted with N_h dimensions.

Kernel types: As the correlation filters of the tracking modules, KCF [16] is used. It allows varying the kernel type, and we utilise a Gaussian kernel and a polynomial kernel.

Relative scale changes: In order to handle shape deformations as well as changes in the viewing directions, we use flexible aspect ratios. The target scale is changed from the previous target size in four steps (±1 cell and ±2 cells) along the x-axis, along the y-axis, and along both axes simultaneously, which leads to 13 possible variations of scale including the static case.

Delayed update: To handle target drift, partial occlusions, and tiny scale changes (too small to be detected in a frame-to-frame basis), we introduce tracking modules with delayed updates. For these modules, the module update is delayed such that the tracking module has access to up to four previous frames, *i.e.* before the drift / occlusion / scale change occurred.

3.1.2 Tracking module

Feature map extraction: The region of interest (ROI) of each tracking module is centred at the previous target's location. To cover nearby areas, the size of the ROI is β times the previous target's size. For tracking modules with scale changes, we normalise the ROI's image size into the initial ROI's size to conserve the correlation filter's size (see Fig. 2).

Preprint version; final version available at http://ieeexplore.ieee.org
IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
Published by: IEEE

From the resized ROI, the feature map with size $W \times H$ is obtained by the specific feature type of the tracking module.

Attentional weight map estimation: The attentional weight map (AWM) $\mathbf{W} \in \mathbb{R}^{W \times H}$ is the weighted sum of the target confidence regression map $\mathbf{W}_s$ and the centre bias map $\mathbf{W}_w$ as obtained in [3]. However, to estimate $\mathbf{W}^t$ at time t we weight $\mathbf{W}_s^t$ and $\mathbf{W}_w$ differently:

$$\mathbf{W}^t(\mathbf{p}) = \lambda_s \mathbf{W}_w(\mathbf{p}) \mathbf{W}_s^t(\mathbf{p}) + (1 - \lambda_s)\mathbf{W}_w(\mathbf{p}), \quad (1)$$

where $\mathbf{p} = (p, q)$ with $p \in \{1, ..., W\}$ and $q \in \{1, ..., H\}$. Contrary to [3], $\mathbf{W}_s^t$ is weighted by $\mathbf{W}_w$ in the first term to give more weight to features in the centre, which results in higher performance if $\mathbf{W}_s^t$ is noisy. As [3] estimated the weight factor λ_s dynamically from the previous frames, the tracker shows low robustness for the abrupt changes of the target appearance. Thus, we fix λ_s to provide a stable weight to the target confidence regression map.

Target position adjustment: The response map $\mathbf{R}^t \in \mathbb{R}^{W \times H}$ at time t is obtained by applying the associated correlation filter to the feature map weighted by the AWM $\mathbf{W}^t$. The position of the target is quantised due to the grid of the feature map with cell size $N_g \times N_g$. This can lead to a critical drift problem when the cell size increases due to an increased target size. Thus, we find the target position $\hat{\mathbf{p}}^t$ within the ROI by interpolating the response values near the peak position $\mathbf{p}'^t$ with interpolation range N_p as follows:

$$\hat{\mathbf{p}}^t = \mathbf{p}'^t + \sum_{i=-N_p}^{N_p} \sum_{j=-N_p}^{N_p} (i, j)\mathbf{R}^t\big(\mathbf{p}' + (i, j)\big). \quad (2)$$

Using the interpolated position within the ROI, the position of the target on the image is estimated by

$$(x^t, y^t) = (x^{t-1}, y^{t-1}) + \lfloor N_g \times \hat{\mathbf{p}}^t \rfloor. \quad (3)$$

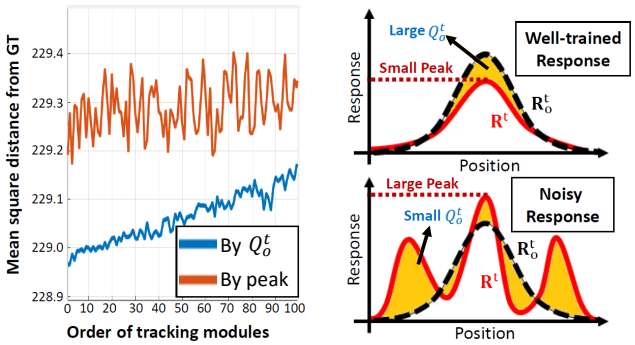
Validation:

The precision of the tracking module is estimated by the validation score. Previous correlation filter-based trackers [5, 18] determine the scale change by comparing the peak values of the correlation filter response maps obtained with various scale changes. However, Fig. 3(a) shows that this measure is not informative enough in the proposed network due to the significantly varying intensity ranges of the response map according to the various characteristics of the correlation filters (feature type, kernel).

We thus select the filter with the least number of noisy peaks as it most likely represents the target as shown in Fig. 3(b). Based on this intuition, our novel validation score Q_o^t is estimated by the difference between the response map $\mathbf{R}^t$ and the ideal response map $\mathbf{R}_o^t$:

$$Q_o^t = \exp(-\|\mathbf{R}^t - \mathbf{R}_o^t\|_2^2), \quad (4)$$

where $\mathbf{R}_o^t = \mathcal{G}\big(\mathbf{p}'^t, \sigma_G^2\big)_{W \times H}$ is a two-dimensional Gaussian window with size $W \times H$ centred at $\mathbf{p}'^t$, and variance σ_G^2.



(a) Reliability of validation scores (b) Validation score estimation

Figure 3. **Validation Score Estimation.** (a) Comparison of the mean square distance errors on the positions of the tracking modules with respect to the order of the tracking modules based on the peak values of the correlation filter response map and the proposed validation scores. Contrary to the peak values, the order obtained by the new estimation method shows high correlation with the distance errors. (b) New estimation method for the validation score, which shows better reliability than using peak values.

3.1.3 Tracking module update

Out of the 260 tracking modules, we only update four basic tracking modules; one per feature type and kernel type. Modules with scale changes can share the correlation filter with the basic module without scale change, as the ROI of the scaled modules is resized to be the same size as the basic tracking module's ROI. Modules with the delayed update can re-use the correlation filter of the previous frame(s). In case a module with the delayed update is best performing, the basic tracking modules with the same delayed update are used as the update source. The basic tracking modules are updated by the feature map weighted by the attentional weight map, as detailed in [3] and [16].

3.2. Attention Network

3.2.1 Prediction Sub-network

We employ a deep regression network to predict the validation scores $\widehat{\mathbf{Q}}^t \in \mathbb{R}^{260}$ of all modules at the current frame t based on the previous validation scores $\{\mathbf{Q}^{t-1}, \mathbf{Q}^{t-2}, ...\}$, where $\mathbf{Q}^t \in \mathbb{R}^{260}$. As long short-term memory (LSTM) [17] can model sequential data with high accuracy, we use it to consider the dynamic changes of the validation scores.

We first normalise the validation scores obtained at the previous frame $\mathbf{Q}^{t-1}$ from zero to one as

$$\widetilde{\mathbf{Q}}^{t-1} = \frac{\mathbf{Q}^{t-1} - \min(\mathbf{Q}^{t-1})}{\max(\mathbf{Q}^{t-1}) - \min(\mathbf{Q}^{t-1})}, \quad (5)$$

where $\min$ and $\max$ provide the minimum and maximum values among all elements of the input vector. Then the normalised scores $\widetilde{\mathbf{Q}}^{t-1}$ are sequentially fed in the LSTM, and the four following fully connected layers estimate the

Preprint version; final version available at http://ieeexplore.ieee.org
IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
Published by: IEEE

normalised validation scores of the current frame $\widetilde{\mathbf{Q}}_*^t$. The detailed network architecture is described in Fig. 2. Finally, based on the assumption that the range of the predicted validation scores is identical to the range of the previous validation scores, we transform the normalised scores back and obtain the predicted validation score $\widehat{\mathbf{Q}}^t$:

$$\widehat{\mathbf{Q}}^t = \widetilde{\mathbf{Q}}_*^t \Big(\max(\mathbf{Q}^{t-1}) - \min(\mathbf{Q}^{t-1}) \Big) + \min(\mathbf{Q}^{t-1}). \quad (6)$$

3.2.2 Selection Sub-network

Based on the predicted validation scores $\widehat{\mathbf{Q}}^t$, the selection sub-network selects the tracking modules which are activated for the current frame. The role of the selection sub-network is twofold. On the one hand, it should select tracking modules which are likely to perform well. On the other hand, if a tracking module is not activated for a long time, it is hard to estimate its performance as the prediction error accumulates over time, so modules should be activated from time to time.

Therefore, the selection sub-network consists of two parts fulfilling these roles. The first part is a top-k selection layer which selects the k modules with the highest predicted validation score, resulting in a binary vector. The second part consists of four fully connected layers followed by a $\tanh$ layer to estimate the prediction error, resulting in a vector with values ranging between -1 and 1. The results of both parts are integrated by $\max$-pooling, resulting in the attentional scores $\mathbf{s}^t \in [0, 1]$. The binary attentional vector $\langle \mathbf{s}^t \rangle$ is obtained by selecting the N_a tracking modules with the highest values within $\mathbf{s}^t$, where $\langle \cdot \rangle$ is used to denote vectors containing binary values. As N_a is bigger than k and the results of the $\tanh$ layer being smaller than one, $\langle \mathbf{s}^t \rangle$ essentially includes all modules of the top-k part and $N_a - k$ modules with the highest estimated prediction error.

At the current frame, the modules within the correlation filter network which should be activated are chosen according to $\langle \mathbf{s}^t \rangle$, so the validation scores of the active modules $\mathbf{Q}_o^t \in \mathbb{R}^{260}$ can be obtained from the correlation filter network as shown in Fig. 2 ($\mathbf{Q}_o^t$ contains zeros for the modules which are not activated). Then, the final validation scores $\mathbf{Q}^t$ are formulated as

$$\mathbf{Q}^t = (\mathbf{1} - \langle \mathbf{s}^t \rangle) * \widehat{\mathbf{Q}}^t + \langle \mathbf{s}^t \rangle * \mathbf{Q}_o^t, \quad (7)$$

where $*$ represents the element-wise multiplication.

3.2.3 Training

Training Data: We randomly choose the training sample i out of all frames within the training sequences. Then the ground truth validation score $\mathbf{Q}_{GT}(i)$ is obtained by setting the target position to the ground truth given in the dataset and operating all correlation filters.

To train the LSTM layer, the attention network is sequentially fed by the validation scores of the previous ten frames. After feeding the attention network, we obtain the predicted validation scores $\widehat{\mathbf{Q}}(i)$ and the attentional binary vector $\langle \mathbf{s}(i) \rangle$. The final validation scores from the i-th training sample is then defined following Eq.(7): $\mathbf{Q}(i) = (\mathbf{1} - \langle \mathbf{s}(i) \rangle) * \widehat{\mathbf{Q}}(i) + \langle \mathbf{s}(i) \rangle * \mathbf{Q}_{GT}(i)$.

Loss function: We develop a sparsity based loss function which minimises the error between the final validation scores $\mathbf{Q}(i)$ and the ground truth validation scores $\mathbf{Q}_{GT}(i)$ while using the least number of active modules:

$$E = \sum_{i=1}^{N} \left\{ \|\mathbf{Q}(i) - \mathbf{Q}_{GT}(i)\|_2^2 + \lambda \, \|\langle \mathbf{s}(i) \rangle\|_0 \right\}, \quad (8)$$

where N is the number of training samples. However, as we need to estimate the gradient of the loss function, the discrete variable $\langle \mathbf{s}(i) \rangle$ is substituted with the continuous attentional scores $\mathbf{s}(i)$, resulting in

$$E = \sum_{i=1}^{N} \left\{ \left\| (\mathbf{1} - \mathbf{s}(i)) * (\widehat{\mathbf{Q}}(i) - \mathbf{Q}_{GT}(i)) \right\|_2^2 + \lambda \|\mathbf{s}(i)\|_0 \right\}. \quad (9)$$

Training sequence: We train the network in two steps, *i.e.* we first train the prediction sub-network and subsequently the selection sub-network. We found that training the network as a whole leads to the selection of the same modules by the selection sub-network each time, which in turn also prohibits the prediction sub-network to learn the accuracy of the selected tracking modules.

For training the prediction sub-network, the sparsity term is removed by setting all values of $\mathbf{s}(i)$ to zeros, such that the objective becomes to minimise the prediction error:

$$E = \sum_{i=1}^{N} \left\{ \left\| \widehat{\mathbf{Q}}(i) - \mathbf{Q}_{GT}(i) \right\|_2^2 \right\}. \quad (10)$$

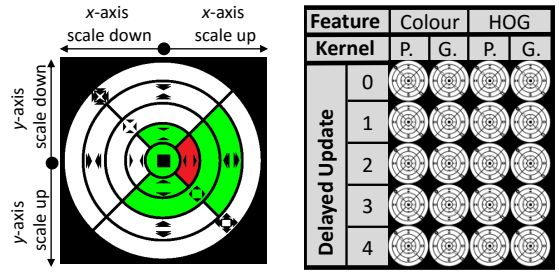
The subsequent training of the selection sub-network should then be performed with the original loss function as in Eq.(8). However, we found that the error is not sufficiently back-propagated to the fully connected layers of the selection sub-network because of the $\max$-pooling and $\tanh$ layer. If the prediction is assumed to be fixed, the output of the top-k part can be regarded as constant. Furthermore, the $\tanh$ layer only squashes the output of the last fully connected layer $\mathbf{h}$, but does not change the sparsity. Therefore, the loss function can employ $\mathbf{h}(i)$ obtained by the i-th training sample for the sparsity term:

$$E = \sum_{i=1}^{N} \left\{ \left\| (\mathbf{1} - \mathbf{s}(i)) * (\widehat{\mathbf{Q}}(i) - \mathbf{Q}_{GT}(i)) \right\|_2^2 + \lambda \ln\big(1 + \|\mathbf{h}(i)\|_1\big) \right\}, \quad (11)$$

where the sparsity norm is approximated by a sparsity-aware penalty term as described in [37].

Optimisation: We use the Adam optimiser [20] to optimise the prediction sub-network, and gradient descent [24] for the optimisation of the selection sub-network.

Preprint version; final version available at http://ieeexplore.ieee.org
IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
Published by: IEEE



(a) Attention map (b) Global attention map

Figure 4. **Attention map.** (a) Each region within the attention map represents one tracking module, each covering another scale change. The green colour indicates the active modules to be operated at a time, and the best module which is used to determine the tracking result is coloured red. (b) Multiple attention maps with different properties of feature types, kernel types, and delayed updates.

3.3. Handling full occlusions

A full occlusion is assumed if the score $\mathbf{Q}^t_{max} = \max(\mathbf{Q}^t)$ of the best performing tracking module drops suddenly as described by $\mathbf{Q}^t_{max} < \lambda_r \overline{\mathbf{Q}}^{t-1}_{max}$ with $\overline{\mathbf{Q}}^t_{max} = (1-\gamma)\overline{\mathbf{Q}}^{t-1}_{max} + \gamma \mathbf{Q}^t_{max}$ and $\overline{\mathbf{Q}}^0_{max} = \mathbf{Q}^1_{max}$. λ_r is the detection ratio threshold and γ is an interpolation factor. If a full occlusion is detected at time t, we add and activate four additional basic tracking modules for the period of N_r frames without updating them. The ROI of these modules is fixed to the target position at time t. If one of the re-detection modules is selected as the best module, all tracking modules are replaced by the modules saved at time t.

4. Experimental Result

4.1. Implementation

$20\%(N_a = 52)$ among all modules were selected as active modules. One quarter of them $(k = 13)$ were chosen by the top-k layer. The weight factor for the attentional weight map estimation was set to $\lambda_s = 0.9$, and the interpolation range to $N_p = 2$. The sparsity weight for training the attention network was set to $\lambda = 0.1$. The parameters for full occlusion handling, λ_r and N_r, were experimentally set to 0.7 and 30 using scenes containing full occlusions. The other parameters were set as mentioned in [3, 16]: $N_g = 4$, $N_h = 31$, $\beta = 2.5$, $\sigma_G = \sqrt{WH}/10$, and $\gamma = 0.02$. The parameters were fixed for all training and evaluation sequences. The input image was resized such that the minimum length of the initial bounding box equals 40 pixels. To initialise the LSTM layer, all modules were activated for the first ten frames.

We used MATLAB to implement the correlation filter network, and TensorFlow [1] to implement the attention network. The two networks communicated to each other via a TCP-IP socket. The tracking module update and the attention network ran in parallel for faster execution. The computational speed was 15.0 FPS in the CVPR2013 dataset [38], and the attention network only took 3ms per frame. The prediction sub-network and selection sub-network were each trained for 1000K iterations, which took about 10 hours. The computational environment had an Intel i7-6900K CPU @ 3.20GHz, 32GB RAM, and a NVIDIA GTX1070 GPU. We release the source code for tracking and training along with the attached experimental results.[1]

Table 1. Quantitative results on the CVPR2013 dataset [38]

	Algorithm	Pre. score	Mean FPS	Scale
Proposed	ACFN	86.0%	15.0	O
	CFN+predNet	82.3%	14.4	O
	CFN	81.3%	6.9	O
	CFN+simpleSel.	79.4%	15.7	O
	CFN-	78.4%	15.5	O
Real-time	SCT [3]	84.5%	40.0	X
	MEEM [42]	81.4%	19.5	X
	KCF [16]	74.2%	**223.8**	X
	DSST [5]	74.0%	25.4	O
	Struck [15]	65.6%	10.0	O
	TLD [19]	60.8%	21.7	O
Non Real-time	C-COT [8]	**89.9%**	<1.0	O
	MDNet-N [29]	87.7%	<1.0	O
	MUSTer [18]	86.5%	3.9	O
	FCNT [35]	85.6%	3.0	O
	D-SRDCF [6]	84.9%	<1.0	O
	SRDCF [7]	83.8%	5	O
	STCT [36]	78.0%	2.5	O

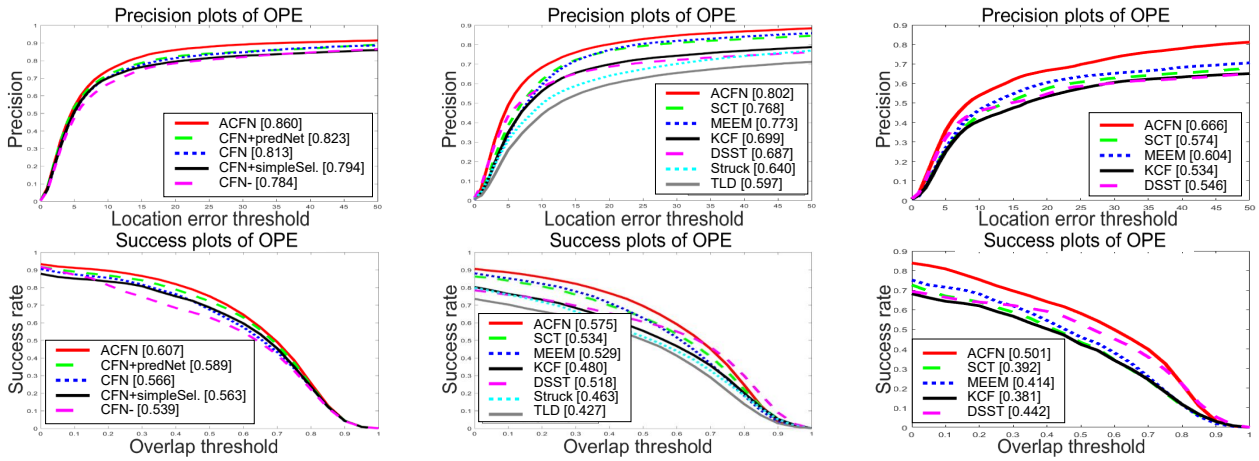
4.2. Dataset

To evaluate the proposed framework, we used the CVPR2013 [38] (51 targets, 50 videos), TPAMI2015 [39] (100 targets, 98 videos), and VOT2014 [22] datasets (25 targets, 25 videos), which contain the ground truth of the target bounding box at every frame. These datasets have been frequently used [3, 8, 15, 16, 18, 29, 42] as they include a large variety of environments to evaluate the general performance of visual trackers.

To train the attention network for evaluating on the CVPR2013 and TPAMI2015 datasets, the VOT2014 [22] and VOT2015 [21] datasets were used. After removing scenes which overlap with the CVPR2013 and TPAMI2015 datasets, 44 sequences remained for training. For the evaluation on the VOT2014 dataset, we trained the attention network by 39 sequences of the CVPR2013 dataset after eliminating overlapping scenes. For both cases, we obtained additional training data by slightly moving the ground truth of the target position in eight directions (10% of the target size to the left, right, top, bottom, top-left, top-right, bottom-left, and bottom-right), as well as by changing the size of the tracking box (10% enlarging and shrinking). Therefore, we had 11

[1] https://sites.google.com/site/jwchoivision/

Preprint version; final version available at http://ieeexplore.ieee.org
IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
Published by: IEEE



(a) Self-comparison on CVPR2013 dataset (b) Evaluation plots on TPAMI2015 dataset (c) Evaluation plots on VOT2014 dataset

Figure 5. **Evaluation Results.** ACFN showed the best performance within the self-comparison, and the state-of-the-art performance amongst real-time trackers in TPAMI2015 [39] and VOT2014 [22] dataset. The numbers within the legend are the average precisions when the centre error threshold equals 20 pixels (top row), or the area under the curve of the success plot (bottom row).

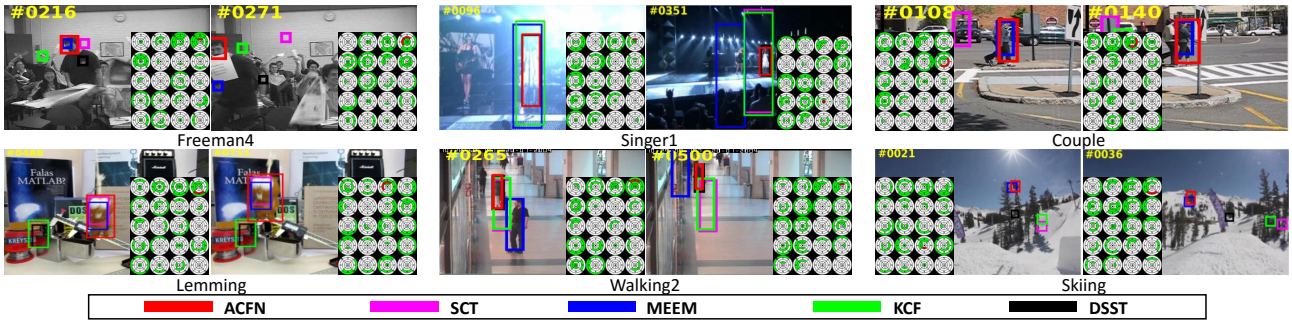


Figure 6. **Qualitative Results.** The used sequences are Freeman4, Singer1, Couple, Lemming, Walking2, and Skiing.

times the training data available compared to the original sequences without augmentations.

4.3. Evaluation

As performance measure, we used the average precision curve of one-pass evaluation (OPE) as proposed in [38]. The average precision curve was estimated by averaging the precision curves of all sequences, which was obtained using two bases: location error threshold and overlap threshold. The precision curve based on the location error threshold (precision plot) shows the percentage of correctly tracked frames on the basis of a distance between the centres of the tracked box and ground truth. The precision curve based on the overlap threshold (success plot) indicates the percentage of correctly tracked frames on the basis of the overlap region between the tracked box and ground truth. As representative scores of trackers, the average precisions when the centre error threshold equals 20 pixels and the area under curve of the success plot were used.

To describe the attention on the tracking modules qualitatively, we built an attention map which is depicted in Fig. 4. In the attention map, one area represents one tracking module of a specific scale change, and within the global attention map there are many maps with different feature types, kernel types, and delayed updates.

4.4. Self-comparison

To analyse the effectiveness of the attention network, we compared the full framework with four additional trackers. The correlation filter network (CFN) operated all the associated tracking modules. CFN is similar to SCT [3], but uses all 260 filters of ACFN instead of just 4. The limited correlation filter network (CFN-) constantly operated 20% of the modules which are most frequently selected as the best module by ACFN in the CVPR2013 dataset. CFN with simple selection mechanism (CFN+simpleSel.) utilised the previous validation scores as the predicted validation scores, and the top-k modules with high validation scores were selected as the active modules while the other active modules were selected randomly. CFN with the prediction sub-network (CFN+predNet) used the score prediction network to predict the current validation scores, but selected the modules with high prediction errors randomly. The prediction network of CFN+predNet was extracted from the trained ACFN.

Preprint version; final version available at http://ieeexplore.ieee.org
IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
Published by: IEEE

The results of the comparison with these four trackers are shown in Fig. 5(a) and Table 1. As CFN performs worse than SCT, it shows that it is not sufficient to simply increase the number of filters to obtain a high performance. Therefore, ACFN provides a meaningful solution to the problem of integrating correlation filters with various characteristics. The attentional mechanisms result both in higher accuracy and higher efficiency.

In addition, ACFN outperformed the trackers which only contain a subset of the full framework. Interestingly, CFN- showed worse performance than CFN, which confirmed that a large variety of tracking modules was essential to track the target. By comparing the performance of CFN and CFN+simpleSel., one can see that the performance of the tracker was reduced when the active modules were chosen without considering the dynamic changes of the target. From the performance of CFN+predNet and ACFN, it can be confirmed that the selection sub-network has an important role for the performance of the tracker.

4.5. Experiments on the benchmark datasets

The results of the state-of-the-art methods, including FCNT [35], STCT [36], SRDCF-Deep [6], SRDCF [7], DSST [5], and C-COT [8] were obtained from the authors. In addition, the results of MDNet-N [29], MUSTer [18], MEEM [42], KCF [16], SCT [3], Struck [15], and TLD [19] were estimated using the authors' implementations. MDNet-N is a version of MDNet [29], which is trained by the images of Image-Net [31] as described in VOT2016 [23].

In Table 1, the precision scores on the CVPR2013 dataset are presented along with the computational speed of the algorithms and their ability to recognise scale changes. The proposed algorithm ran sufficiently fast to be used in real time. Among real-time trackers, ACFN showed the state-of-the-art performance for both benchmark datasets. Especially, among the real-time trackers considering scale changes, ACFN improved the relative performance by 12% compared to DSST [5] which was the previous state-of-the-art algorithm in the CVPR2013 dataset. Fig. 5 shows the performances of the real-time trackers, where ACFN demonstrates state-of-the-art performance in both the TPAMI2015 and the VOT2014 datasets. Some qualitative results of ACFN are shown in Fig. 6.

4.6. Analysis on Attention Network

To analyse the results of the attention network, the frequency map was obtained by normalising the frequency that each module was selected as the active module or the best module in the CVPR2013 dataset [38]. Fig. 7 shows the frequency maps for different tracking situations. In the frequency map obtained across all sequences, the module trackers with HOG and Gaussian kernel were most frequently selected as the best module, while diverse modules were selected as active modules in various situations. The tracking modules which detect increasing scale were selected more often than others in scenes containing enlarging targets, while the modules detecting scale-down changes were chosen most often in scenes with shrinking targets. Interestingly, when we estimated the frequency map from the tracking failure scenes of the CVPR2013 dataset (Matrix, MotorRolling, IronMan), the distribution of the active modules was relatively identical, which meant that the attention became scattered as the target was missed.

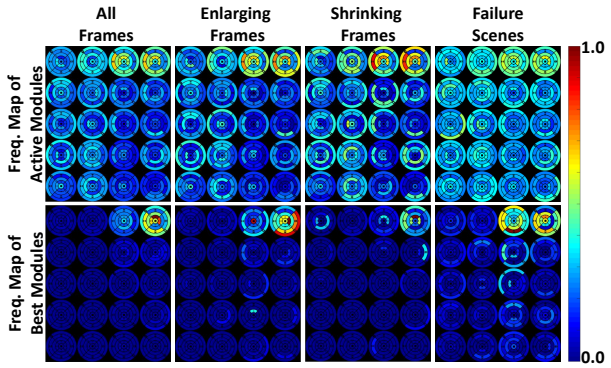


Figure 7. **Frequency map for different tracking situations.** The distribution of the attention is determined by the dynamic changes of the target.

5. Conclusion

In this paper, a new visual tracking framework using an attentional mechanism was proposed. The proposed framework consists of two major networks: the correlation filter network and the attention network. Due to the attentional mechanism which reduces the computational load, the correlation filter network can consider significantly more status and dynamic changes of the target, including novel properties such as flexible aspect ratios and delayed updates. The attention network was trained by the general expectation of dynamic changes to adaptively select the attentional subset of all tracking modules. The effectiveness of the attentional mechanism for visual tracking was validated by its high robustness even with rapid computation. In the experiments based on several tracking benchmark datasets, the proposed framework performed comparable to deep learning-based trackers which cannot be operated in real-time, and showed state-of-the-art performance amongst the real-time trackers. As a future work, we will extend the attentional mechanism to correlation filters employing deep convolution features.

Acknowledgement: This work was partly supported by the ICT R&D program of MSIP/IITP (No.B0101-15-0552, Development of Predictive Visual Intelligence Technology), the SNU-Samsung Smart Campus Research Centre, Brain Korea 21 Plus Project, and EU FP7 project WYSIWYD under Grant 612139. We thank the NVIDIA Corporation for their GPU donation.

Preprint version; final version available at http://ieeexplore.ieee.org
IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
Published by: IEEE

References

[1] M. Abadi et al. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org. 6

[2] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui. Visual object tracking using adaptive correlation filters. In *CVPR*, pages 2544–2550, 2010. 2

[3] J. Choi, H. J. Chang, J. Jeong, Y. Demiris, and J. Y. Choi. Visual tracking using attention-modulated disintegration and integration. In *CVPR*, pages 4321–4330, 2016. 2, 3, 4, 6, 7, 8

[4] A. Cully, J. Clune, D. Tarapore, and J.-B. Mouret. Robots that can adapt like animals. *Nature*, 521(7553):503–507, 2015. 2

[5] M. Danelljan, G. Hager, F. S. Khan, and M. Felsberg. Discriminative scale space tracking. *IEEE Trans. on PAMI*. to be published. 2, 4, 6, 8

[6] M. Danelljan, G. Hager, F. S. Khan, and M. Felsberg. Convolutional features for correlation filter based visual tracking. In *ICCV workshop*, pages 58–66, 2016. 1, 2, 6, 8

[7] M. Danelljan, G. Hger, F. Khan, and M. Felsberg. Learning spatially regularized correlation filters for visual tracking. In *ICCV*, pages 4310–4318, 2015. 2, 6, 8

[8] M. Danelljan, A. Robinson, F. S. Khan, and M. Felsberg. Beyond correlation filters: Learning continuous convolution operators for visual tracking. In *ECCV*, pages 472–488, 2016. 1, 2, 6, 8

[9] Y. Demiris. Imitation as a dual-route process featuring predictive and learning components: a biologically-plausible computational model. In *Imitation in Animals and Artifacts*, pages 327–361. MIT Press, 2002. 2

[10] Y. Demiris, L. Aziz-Zadeh, and J. Bonaiuto. Information Processing in the Mirror Neuron System in Primates and Machines. *Neuroinformatics*, 12(1):63–91, 2014. 2

[11] Y. Demiris and B. Khadhouri. Hierarchical attentive multiple models for execution and recognition of actions. *Robotics and Autonomous Systems*, 54(5):361–369, 2006. 2

[12] S. Frintrop, E. Rome, and H. I. Christensen. Computational visual attention systems and their cognitive foundations: A survey. *ACM Trans. Appl. Percept.*, 7(1):6:1–6:39, 2010. 2

[13] C. D. Gilbert and W. Li. Top-down influences on visual processing. *Nature Reviews Neuroscience*, 14(5):350–363, 2013. 1, 2

[14] R. M. Haefner, P. Berkes, and J. Fiser. Perceptual decision-making as probabilistic inference by neural sampling. *Neuron*, 90(3):649–660, 2016. 1, 2

[15] S. Hare, S. Golodetz, A. Saffari, V. Vineet, M. M. Cheng, S. L. Hicks, and P. H. S. Torr. Struck: Structured output tracking with kernels. *IEEE Trans. on PAMI*, 38(10):2096–2109, 2016. 6, 8

[16] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista. High-speed tracking with kernelized correlation filters. *IEEE Trans. on PAMI*, 37(3):583–596, 2015. 2, 3, 4, 6, 8

[17] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997. 4

[18] Z. Hong, Z. Chen, C. Wang, X. Mei, D. Prokhorov, and D. Tao. Multi-store tracker (muster): a cognitive psychology inspired approach to object tracking. In *CVPR*, pages 749–758, 2015. 2, 4, 6, 8

[19] Z. Kalal, K. Mikolajczyk, and J. Matas. Tracking-learning-detection. *IEEE Trans. on PAMI*, 34(7):1409–1422, 2012. 6, 8

[20] D. Kingma and J. Ba. Adam: A Method for Stochastic Optimization. In *International Conference for Learning Representations*, 2015. 5

[21] M. Kristan et al. The Visual Object Tracking VOT2015 Challenge Results. In *ICCV Workshops*, pages 1–23, 2015. 1, 6

[22] M. Kristan et al. The visual object tracking vot2014 challenge results. In *ECCV 2014 Workshops Proceedings Part II*, pages 191–217, 2015. 1, 6, 7

[23] M. Kristan et al. The Visual Object Tracking VOT2016 Challenge Results. In *ICCV Workshops*, 2016. 1, 8

[24] Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller. Efficient backprop. In *Neural networks: Tricks of the trade*, pages 9–48. Springer, 2012. 5

[25] K. Lee, D. Ognibene, H. J. Chang, T.-K. Kim, and Y. Demiris. Stare: Spatio-temporal attention relocation for multiple structured activities detection. *Transactions on Image Processing*, 24(12):5916–5927, 2015. 2

[26] C. Ma, J.-B. Huang, X. Yang, and M.-H. Yang. Hierarchical convolutional features for visual tracking. In *ICCV*, pages 3074–3082, 2015. 1, 2

[27] C. Ma, X. Yang, C. Zhang, and M.-H. Yang. Long-term correlation tracking. In *CVPR*, pages 5388–5396, 2015. 2

[28] H. Makino and T. Komiyama. Learning enhances the relative impact of top-down processing in the visual cortex. *Nature Neuroscience*, 18:1116–1122, 2015. 1, 2

[29] H. Nam and B. Han. Learning Multi-Domain Convolutional Neural Networks for Visual Tracking. In *CVPR*, pages 4293–4302, 2016. 1, 2, 6, 8

[30] Y. Qi, S. Zhang, L. Qin, H. Yao, Q. Huang, J. Lim, and M.-H. Yang. Hedged deep tracking. In *CVPR*, pages 4303–4311, 2016. 1, 2

[31] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. Imagenet large scale visual recognition challenge. *IJCV*, 115(3):211–252, 2015. 8

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

[33] R. Tao, E. Gavves, and A. W. Smeulders. Siamese instance search for tracking. In *CVPR*, pages 1420–1429, 2016. 1, 2

[34] J. K. Tsotsos. *A computational perspective on visual attention*. MIT Press, 2011. 2

[35] L. Wang, W. Ouyang, X. Wang, and H. Lu. Visual tracking with fully convolutional networks. In *ICCV*, pages 3119–3127, 2015. 1, 2, 6, 8

[36] L. Wang, W. Ouyang, X. Wang, and H. Lu. Stct: Sequentially training convolutional networks for visual tracking. In *CVPR*, pages 1373–1381, 2016. 1, 2, 6, 8

[37] J. Weston, A. Elisseeff, B. Schölkopf, and M. Tipping. Use of the zero norm with linear models and kernel methods. *J. Mach. Learn. Res.*, 3:1439–1461, 2003. 5

[38] Y. Wu, J. Lim, and M. H. Yang. Online object tracking: A benchmark. In *CVPR*, pages 2411–2418, 2013. 1, 6, 7, 8

Preprint version; final version available at http://ieeexplore.ieee.org
IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
Published by: IEEE

[39] Y. Wu, J. Lim, and M. H. Yang. Object tracking benchmark. *IEEE Trans. on PAMI*, 37(9):1834–1848, 2015. 1, 6, 7

[40] H. Yang, L. Shao, F. Zheng, L. Wang, and Z. Song. Recent advances and trends in visual tracking: A review. *Neurocomputing*, 74(18):3823 – 3831, 2011. 1

[41] S. Yun, J. Choi, Y. Yoo, K. Yun, and J. Young Choi. Action-decision networks for visual tracking with deep reinforcement learning. In *CVPR*, 2017. 1

[42] J. Zhang, S. Ma, and S. Sclaroff. Meem: Robust tracking via multiple experts using entropy minimization. In *ECCV*, pages 188–203, 2014. 6, 8