Beyond Correlation Filters: Learning Continuous Convolution Operators for Visual Tracking

Martin Danelljan(✉), Andreas Robinson, Fahad Shahbaz Khan, and Michael Felsberg

CVL, Department of Electrical Engineering, Linköping University, Linköping, Sweden
{martin.danelljan,andreas.robinson,fahad.khan,michael.felsberg}@liu.se

Abstract. Discriminative Correlation Filters (DCF) have demonstrated excellent performance for visual object tracking. The key to their success is the ability to efficiently exploit available negative data by including all shifted versions of a training sample. However, the underlying DCF formulation is restricted to single-resolution feature maps, significantly limiting its potential. In this paper, we go beyond the conventional DCF framework and introduce a novel formulation for training *continuous* convolution filters. We employ an implicit interpolation model to pose the learning problem in the continuous spatial domain. Our proposed formulation enables efficient integration of multi-resolution deep feature maps, leading to superior results on three object tracking benchmarks: OTB-2015 (+5.1 % in mean OP), Temple-Color (+4.6 % in mean OP), and VOT2015 (20 % relative reduction in failure rate). Additionally, our approach is capable of sub-pixel localization, crucial for the task of accurate feature point tracking. We also demonstrate the effectiveness of our learning formulation in extensive feature point tracking experiments.

1 Introduction

Visual tracking is the task of estimating the trajectory of a target in a video. It is one of the fundamental problems in computer vision. Tracking of objects or feature points has numerous applications in robotics, structure-from-motion, and visual surveillance. In recent years, Discriminative Correlation Filter (DCF) based approaches have shown outstanding results on object tracking benchmarks [30,46]. DCF methods train a correlation filter for the task of predicting the target classification scores. Unlike other methods, the DCF efficiently utilize all spatial shifts of the training samples by exploiting the discrete Fourier transform.

Deep convolutional neural networks (CNNs) have shown impressive performance for many tasks, and are therefore of interest for DCF-based tracking. A CNN consists of several layers of convolution, normalization and pooling operations. Recently, activations from the last convolutional layers have been successfully employed for image classification. Features from these deep convolutional

Electronic supplementary material The online version of this chapter (doi:10.1007/978-3-319-46454-1_29) contains supplementary material, which is available to authorized users.

B. Leibe et al. (Eds.): ECCV 2016, Part V, LNCS 9909, pp. 472–488, 2016.
DOI: 10.1007/978-3-319-46454-1_29

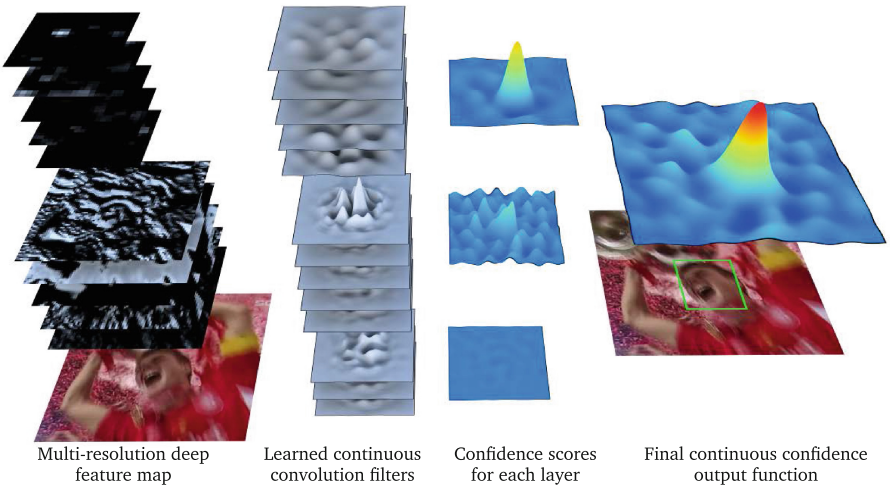


Fig. 1. Visualization of our continuous convolution operator, applied to a multi-resolution deep feature map. The feature map (*left*) consists of the input RGB patch along with the first and last convolutional layer of a pre-trained deep network. The second column visualizes the continuous convolution filters learned by our framework. The resulting continuous convolution outputs for each layer (third column) are combined into the final continuous confidence function (*right*) of the target (green box). (Color figure online)

layers are discriminative while preserving spatial and structural information. Surprisingly, in the context of tracking, recent DCF-based methods [10,35] have demonstrated the importance of shallow convolutional layers. These layers provide higher spatial resolution, which is crucial for accurate target localization. However, fusing multiple layers in a DCF framework is still an open problem.

The conventional DCF formulation is limited to a single-resolution feature map. Therefore, all feature channels must have the same spatial resolution, as in e.g. the HOG descriptor. This limitation prohibits joint fusion of multiple convolutional layers with different spatial resolutions. A straightforward strategy to counter this restriction is to explicitly resample all feature channels to the same common resolution. However, such a resampling strategy is both cumbersome, adds redundant data and introduces artifacts. Instead, a principled approach for integrating multi-resolution feature maps in the learning formulation is preferred.

In this work, we propose a novel formulation for learning a convolution operator in the *continuous* spatial domain. The proposed learning formulation employs an implicit interpolation model of the training samples. Our approach learns a set of convolution filters to produce a *continuous-domain* confidence map of the target. This enables an elegant fusion of multi-resolution feature maps in a joint learning formulation. Figure 1 shows a visualization of our continuous convolution operator, when integrating multi-resolution deep feature maps. We validate the effectiveness of our approach on three object tracking benchmarks:

OTB-2015 [46], Temple-Color [32] and VOT2015 [29]. On the challenging OTB-2015 with 100 videos, our object tracking framework improves the state-of-the-art from 77.3 % to 82.4 % in mean overlap precision.

In addition to multi-resolution fusion, our continuous domain learning formulation enables accurate sub-pixel localization. This is achieved by labeling the training samples with sub-pixel precise continuous confidence maps. Our formulation is therefore also suitable for accurate feature point tracking. Further, our learning-based approach is discriminative and does not require explicit interpolation of the image to achieve sub-pixel accuracy. We demonstrate the accuracy and robustness of our approach by performing extensive feature point tracking experiments on the popular MPI Sintel dataset [7].

2 Related Work

Discriminative Correlation Filters (DCF) [5,11,24] have shown promising results for object tracking. These methods exploit the properties of circular correlation for training a regressor in a sliding-window fashion. Initially, the DCF approaches [5,23] were restricted to a single feature channel. The DCF framework was later extended to multi-channel feature maps [4,13,17]. The multi-channel DCF allows high-dimensional features, such as HOG and Color Names, to be incorporated for improved tracking. In addition to the incorporation of multi-channel features, the DCF framework has been significantly improved lately by, e.g., including scale estimation [9,31], non-linear kernels [23,24], a long-term memory [36], and by alleviating the periodic effects of circular convolution [11,15,18].

With the advent of deep CNNs, fully connected layers of the network have been commonly employed for image representation [38,43]. Recently, the last (deep) convolutional layers were shown to be more beneficial for image classification [8,33]. On the other hand, the first (shallow) convolutional layer was shown to be more suitable for visual tracking, compared to the deeper layers [10]. The deep convolutional layers are discriminative and possess high-level visual information. In contrast, the shallow layers contain low-level features at high spatial resolution, beneficial for localization. Ma et al. [35] employed multiple convolutional layers in a hierarchical ensemble of independent DCF trackers. Instead, we propose a novel continuous formulation to fuse multiple convolutional layers with different spatial resolutions in a *joint* learning framework.

Unlike object tracking, feature point tracking is the task of accurately estimating the motion of distinctive key-points. It is a core component in many vision systems [1,27,39,48]. Most feature point tracking methods are derived from the classic Kanade-Lucas-Tomasi (KLT) tracker [34,44]. The KLT tracker is a generative method, that is based on minimizing the squared sum of differences between two image patches. In the last decades, significant effort has been spent on improving the KLT tracker [2,16]. In contrast, we propose a discriminative learning based approach for feature point tracking.

Our approach: Our main contribution is a theoretical framework for learning discriminative convolution operators in the continuous spatial domain. Our

formulation has two major advantages compared to the conventional DCF framework. Firstly, it allows a natural integration of multi-resolution feature maps, e.g. combinations of convolutional layers or multi-resolution HOG and color features. This property is especially desirable for object tracking, detection and action recognition applications. Secondly, our continuous formulation enables accurate sub-pixel localization, crucial in many feature point tracking problems.

3 Learning Continuous Convolution Operators

In this section, we present a theoretical framework for learning continuous convolution operators. Our formulation is generic and can be applied for supervised learning tasks, such as visual tracking and detection.

3.1 Preliminaries and Notation

In this paper, we utilize basic concepts and results in continuous Fourier analysis. For clarity, we first formulate our learning method for data defined in a one-dimensional domain, i.e. for functions of a single spatial variable. We then describe the generalization to higher dimensions, including images, in Sect. 3.5.

We consider the space $L^2(T)$ of complex-valued functions $g : \mathbb{R} \to \mathbb{C}$ that are periodic with period $T > 0$ and square Lebesgue integrable. The space $L^2(T)$ is a Hilbert space equipped with an inner product $\langle \cdot, \cdot \rangle$. For functions $g, h \in L^2(T)$,

$$\langle g, h \rangle = \frac{1}{T} \int_0^T g(t)\overline{h(t)}\,\mathrm{d}t\,, \qquad g * h(t) = \frac{1}{T} \int_0^T g(t-s)h(s)\,\mathrm{d}s. \tag{1}$$

Here, the bar denotes complex conjugation. In (1) we have also defined the circular convolution operation $* : L^2(T) \times L^2(T) \to L^2(T)$.

In our derivations, we use the complex exponential functions $e_k(t) = e^{i\frac{2\pi}{T}kt}$ since they are eigenfunctions of the convolution operation (1). The set $\{e_k\}_{-\infty}^{\infty}$ further forms an orthonormal basis for $L^2(T)$. We define the Fourier coefficients of $g \in L^2(T)$ as $\hat{g}[k] = \langle g, e_k \rangle$. For clarity, we use square brackets for functions with discrete domains. Any $g \in L^2(T)$ can be expressed in terms of its Fourier series $g = \sum_{-\infty}^{\infty} \hat{g}[k]e_k$. The Fourier coefficients satisfy Parseval's formula $\|g\|^2 = \|\hat{g}\|_{\ell^2}^2$, where $\|g\|^2 = \langle g, g \rangle$ and $\|\hat{g}\|_{\ell^2}^2 = \sum_{-\infty}^{\infty} |\hat{g}[k]|^2$ is the squared ℓ^2-norm. Further, the Fourier coefficients satisfy the two convolution properties $\widehat{g * h} = \hat{g}\hat{h}$ and $\widehat{gh} = \hat{g} * \hat{h}$, where $\hat{g} * \hat{h}[k] := \sum_{l=-\infty}^{\infty} \hat{g}[k-l]\hat{h}[l]$.

3.2 Our Continuous Learning Formulation

Here we formulate our novel learning approach. The aim is to train a continuous convolution operator based on training samples x_j. The samples consist of feature maps extracted from image patches. Each sample x_j contains D feature channels $x_j^1, \ldots, x_j^D$, extracted from the same image patch. Conventional DCF formulations [11,17,24] assume the feature channels to have the same spatial

resolution, i.e. have the same number of spatial sample points. Unlike previous works, we eliminate this restriction in our formulation and let N_d denote the number of spatial samples in x_j^d. In our formulation, the feature channel $x_j^d \in \mathbb{R}^{N_d}$ is viewed as a function $x_j^d[n]$ indexed by the discrete spatial variable $n \in \{0, \dots, N_d - 1\}$. The sample space is expressed as $\mathcal{X} = \mathbb{R}^{N_1} \times \dots \times \mathbb{R}^{N_D}$.

To pose the learning problem in the continuous spatial domain, we introduce an implicit interpolation model of the training samples. We regard the continuous interval $[0, T) \subset \mathbb{R}$ to be the spatial support of the feature map. Here, the scalar T represents the size of the support region. In practice, however, T is arbitrary since it represents the scaling of the coordinate system. For each feature channel d, we define the interpolation operator $J_d : \mathbb{R}^{N_d} \to L^2(T)$ of the form,

$$J_d\{x^d\}(t) = \sum_{n=0}^{N_d-1} x^d[n] b_d \left(t - \frac{T}{N_d} n \right). \tag{2}$$

The interpolated sample $J_d\{x^d\}(t)$ is constructed as a superposition of shifted versions of an interpolation function $b_d \in L^2(T)$. In (2), the feature values $x^d[n]$ act as weights for each shifted function. Similar to the periodic assumption in the conventional discrete DCF formulation, a periodic extension of the feature map is also performed here in (2).

As discussed earlier, our objective is to learn a linear convolution operator $S_f : \mathcal{X} \to L^2(T)$. This operator maps a sample $x \in \mathcal{X}$ to a target confidence function $s(t) = S_f\{x\}(t)$, defined on the continuous interval $[0, T)$. Here, $s(t) \in \mathbb{R}$ is the confidence score of the target at the location $t \in [0, T)$ in the image. Similar to other discriminative methods, the target is localized by maximizing the confidence scores in an image region. The key difference in our formulation is that the confidences are defined on a continuous spatial domain. Therefore, our formulation can be used to localize the target with higher accuracy.

In our continuous formulation, the operator S_f is parametrized by a set of convolution filters $f = (f^1, \dots, f^D) \in L^2(T)^D$. Here, $f^d \in L^2(T)$ is the continuous filter for feature channel d. We define the convolution operator as,

$$S_f\{x\} = \sum_{d=1}^{D} f^d * J_d\{x^d\}, \quad x \in \mathcal{X}. \tag{3}$$

Here, each feature channel is first interpolated using (2) and then convolved with its corresponding filter. Note that the convolutions are performed in the continuous domain, as defined in (1). In the last step, the convolution responses from all filters are summed to produce the final confidence function.

In the standard DCF, each training sample is labeled by a discrete function that represents the desired convolution output. In contrast, our samples $x_j \in \mathcal{X}$ are labeled by confidence functions $y_j \in L^2(T)$, defined in the continuous spatial domain. Here, y_j is the desired output of the convolution operator $S_f\{x_j\}$ applied to the training sample x_j. This enables sub-pixel accurate information to be incorporated in the learning. The filter f is trained, given a set of m training sample pairs $\{(x_j, y_j)\}_1^m \subset \mathcal{X} \times L^2(T)$, by minimizing the functional,

$$E(f) = \sum_{j=1}^{m} \alpha_j \left\| S_f\{x_j\} - y_j \right\|^2 + \sum_{d=1}^{D} \left\| w f^d \right\|^2 . \tag{4}$$

Here, the weights $\alpha_j \geq 0$ control the impact of each training sample. We additionally include a spatial regularization term, similar to [11], determined by the penalty function w. This regularization enables the filter to be learned on arbitrarily large image regions by controlling the spatial extent of the filter f. Spatial regions typically corresponding to background features are assigned a large penalty in w, while the target region has small penalty values. Thus, w encodes the prior reliability of features depending on their spatial location. Unlike [11], the penalty function w is defined on the whole continuous interval $[0, T)$ and periodically extended to $w \in L^2(T)$. Hence, $\left\| w f^d \right\| < \infty$ is required in (4). This is implied by our later assumption of w having finitely many non-zero Fourier coefficients $\hat{w}[k]$. Next, we derive the procedure to train the continuous filter f, using the proposed formulation (4).

3.3 Training the Continuous Filter

To train the filter f, we minimize the functional (4) in the Fourier domain. By using results from Fourier analysis it can be shown[1] that the Fourier coefficients of the interpolated feature map are given by $\widehat{J_d\{x^d\}}[k] = X^d[k]\hat{b}_d[k]$. Here, $X^d[k] := \sum_{n=0}^{N_d-1} x^d[n] e^{-i\frac{2\pi}{N_d}nk}$, $k \in \mathbb{Z}$ is the discrete Fourier transform (DFT) of x^d. By using linearity and the convolution property in Sect. 3.1, the Fourier coefficients of the output confidence function (3) are derived as

$$\widehat{S_f\{x\}}[k] = \sum_{d=1}^{D} \hat{f}^d[k] X^d[k] \hat{b}_d[k] , \quad k \in \mathbb{Z}. \tag{5}$$

By applying Parseval's formula to (4) and using (5), we obtain

$$E(f) = \sum_{j=1}^{m} \alpha_j \left\| \sum_{d=1}^{D} \hat{f}^d X_j^d \hat{b}_d - \hat{y}_j \right\|_{\ell^2}^2 + \sum_{d=1}^{D} \left\| \hat{w} * \hat{f}^d \right\|_{\ell^2}^2 . \tag{6}$$

Hence, the functional $E(f)$ can equivalently be minimized with respect to the Fourier coefficients $\hat{f}^d[k]$ for each filter f^d. We exploit the Fourier domain formulation (6) to minimize the original loss (4).

For practical purposes, the filter f needs to be represented by a finite set of parameters. One approach is to employ a parametric model to represent an infinite number of coefficients. In this work, we instead obtain a finite representation by minimizing (6) over the finite–dimensional subspace $V = \operatorname{span}\{e_k\}_{-K_1}^{K_1} \times \ldots \times \operatorname{span}\{e_k\}_{-K_D}^{K_D} \subset L^2(T)^D$. That is, we minimize (6) with respect to the coefficients $\{\hat{f}^d[k]\}_{-K_d}^{K_d}$, while assuming $\hat{f}^d[k] = 0$ for $|k| > K_d$.

[1] See the supplementary material for a detailed derivation.

In practice, K_d determines the number of filter coefficients $\hat{f}^d[k]$ to be computed for feature channel d during learning. Increasing K_d leads to a better estimate of the filter f^d at the cost of increased computations and memory consumption. In our experiments, we set $K_d = \left\lfloor \frac{N_d}{2} \right\rfloor$ such that the number of stored filter coefficients for channel d equals the spatial resolution N_d of the training sample x^d.

To derive the solution to the minimization problem (6) subject to $f \in V$, we introduce the vector of non-zero Fourier coefficients $\hat{\mathbf{f}}^d = (\hat{f}^d[-K_d] \cdots \hat{f}^d[K_d])^{\mathrm{T}} \in \mathbb{C}^{2K_d+1}$ and define the coefficient vector $\hat{\mathbf{f}} = \left[(\hat{\mathbf{f}}^1)^{\mathrm{T}} \cdots (\hat{\mathbf{f}}^D)^{\mathrm{T}}\right]^{\mathrm{T}}$. Further, we define $\hat{\mathbf{y}}_j = (\hat{y}_j[-K] \cdots \hat{y}_j[K])^{\mathrm{T}}$ be the vectorization of the $K := \max_d K_d$ first Fourier coefficients of y_j. To simplify the regularization term in (6), we let L be the number of non-zero coefficients $\hat{w}[k]$, such that $\hat{w}[k] = 0$ for all $|k| > L$. We further define W_d to be the $(2K_d + 2L + 1) \times (2K_d + 1)$ Toeplitz matrix corresponding to the convolution operator $W_d \hat{\mathbf{f}}^d = \operatorname{vec} \hat{w} * \hat{f}^d$. Finally, let W be the block-diagonal matrix $W = W_1 \oplus \cdots \oplus W_D$. The minimization of the functional (6) subject to $f \in V$ is equivalent to the following least squares problem,

$$E_V(\hat{\mathbf{f}}) = \sum_{j=1}^{m} \alpha_j \left\| A_j \hat{\mathbf{f}} - \hat{\mathbf{y}}_j \right\|_2^2 + \left\| W \hat{\mathbf{f}} \right\|_2^2 . \tag{7}$$

Here, the matrix $A_j = [A_j^1 \cdots A_j^D]$ has $2K + 1$ rows and contains one diagonal block A_j^d per feature channel d with $2K_d + 1$ columns containing the elements $\{X_j^d[k]\hat{b}_d[k]\}_{-K_d}^{K_d}$. In (7), $\|\cdot\|_2$ denotes the standard Euclidian norm in $\mathbb{C}^M$.

To obtain a simple expression of the normal equations, we define the sample matrix $A = [A_1^{\mathrm{T}} \cdots A_m^{\mathrm{T}}]^{\mathrm{T}}$, the diagonal weight matrix $\Gamma = \alpha_1 I \oplus \cdots \oplus \alpha_m I$ and the label vector $\hat{\mathbf{y}} = [\hat{\mathbf{y}}_1^{\mathrm{T}} \cdots \hat{\mathbf{y}}_m^{\mathrm{T}}]^{\mathrm{T}}$. The minimizer of (7) is found by solving the normal equations,

$$\left(A^{\mathrm{H}} \Gamma A + W^{\mathrm{H}} W\right) \hat{\mathbf{f}} = A^{\mathrm{H}} \Gamma \hat{\mathbf{y}}. \tag{8}$$

Here, $^{\mathrm{H}}$ denotes the conjugate-transpose of a matrix. Note that (8) forms a sparse linear equation system if w has a small number of non-zero Fourier coefficients $\hat{w}[k]$. In our object tracking framework, presented in Sect. 4.2, we employ the Conjugate Gradient method to iteratively solve (8). For our feature point tracking approach, presented in Sect. 4.3, we use a single-channel feature map and a constant penalty function w for improved efficiency. This results in a diagonal system (8), which can be efficiently solved by a direct computation.

3.4 Desired Confidence and Interpolation Function

Here, we describe the choice of the desired convolution output y_j and the interpolation function b_d. We construct both y_j and b_d by periodically repeating functions defined on the real line. In general, the T-periodic repetition of a function g is defined as $g_T(t) = \sum_{-\infty}^{\infty} g(t - nT)$. In the derived Fourier domain formulation (6), the functions y_j and b_d are represented by their respective Fourier

coefficients. The Fourier coefficients of a periodic repetition g_T can be retrieved from the continuous Fourier transform $\hat{g}(\xi)$ of $g(t)$ as $\hat{g}_T[k] = \frac{1}{T}\hat{g}(\frac{k}{T})$.[2] We use this property to compute the Fourier coefficients of y_j and b_d.

To construct the desired convolution output y_j, we let $u_j \in [0, T)$ denote the estimated location of the target object or feature point in sample x_j. We define y_j as the periodic repetition of the Gaussian function $\exp\big(-\frac{(t-u_j)^2}{2\sigma^2}\big)$ centered at u_j. This provides the following expression for the Fourier coefficients,

$$\hat{y}_j[k] = \frac{\sqrt{2\pi\sigma^2}}{T} \exp\left(-2\sigma^2 \left(\frac{\pi k}{T}\right)^2 - i\frac{2\pi}{T} u_j k\right). \tag{9}$$

The variance σ^2 is set to a small value to obtain a sharp peak. Further, this ensures a negligible spatial aliasing. In our work, the functions b_d are constructed based on the cubic spline kernel $b(t)$. The interpolation function b_d is set to the periodic repetition of a scaled and shifted version of the kernel $b\big(\frac{N_d}{T}\big(t - \frac{T}{2N_d}\big)\big)$, to preserve the spatial arrangement of the feature pyramid. The Fourier coefficients of b_d are then obtained as $\hat{b}_d[k] = \frac{1}{N_d} \exp\big(-i\frac{\pi}{N_d}k\big)\hat{b}\big(\frac{k}{N_d}\big)$. (see footnote 2)

3.5 Generalization to Higher Dimensions

The proposed formulation can be extended to domains of arbitrary number of dimensions. For our tracking applications we specifically consider the two-dimensional case, but higher-dimensional spaces can be treated similarly. For images, we use the space $L^2(T_1, T_2)$ of square-integrable periodic functions of two variables $g(t_1, t_2)$. The complex exponentials are then given by $e_{k_1,k_2}(t_1, t_2) = e^{i\frac{2\pi}{T_1}k_1 t_1} e^{i\frac{2\pi}{T_2}k_2 t_2}$. For the desired convolution output y_j, we employ a 2-dimensional Gaussian function. Further, the interpolation functions are obtained as a separable combination of the cubic spline kernel, i.e. $b(t_1, t_2) = b(t_1)b(t_2)$. The derivations presented in Sect. 3.3 also hold for the higher dimensional cases.

4 Our Tracking Frameworks

We apply our continuous learning formulation for two problems: visual object tracking and feature point tracking. We first present the localization procedure, which is based on maximizing the continuous confidence function. This is shared for both the object and feature point tracking frameworks.

4.1 Localization Step

Here, the aim is to localize the tracked target or feature point using the learned filter f. This is performed by first extracting a feature map $x \in \mathcal{X}$ from the region of interest in an image. The Fourier coefficients of the confidence score function

[2] Further details are given in the supplementary material.

$s = S_f\{x\}$ are then calculated using (5). We employ a two-step approach for maximizing the score $s(t)$ on the interval $t \in [0, T)$. To find a rough initial estimate, we first perform a *grid search*, where the score function is evaluated at the discrete locations $s\big(\frac{Tn}{2K+1}\big)$ for $n = 0, \dots, 2K$. This is efficiently implemented as a scaled inverse DFT of the non-zero Fourier coefficients $\hat{s}[k], k = -K, \dots, K$. The maximizer obtained in the grid search is then used as the initialization for an iterative optimization of the Fourier series expansion $s(t) = \sum_{-K}^{K} \hat{s}[k] e_k(t)$. We employ the standard Newton's method for this purpose. The gradient and Hessian are computed by analytic differentiation of $s(t)$.

4.2 Object Tracking Framework

We first present the object tracking framework based on our continuous learning formulation introduced in Sect. 3.2. We employ multi-resolution feature maps x_j extracted from a pre-trained deep network.[3] Similar to DCF based trackers [11,13,24], we extract a single training sample x_j in each frame. The sample is extracted from an image region centered at the target location and the region size is set to 5^2 times the area of the target box. Its corresponding importance weight is set to $\alpha_j = \frac{\alpha_{j-1}}{1-\lambda}$ using a learning rate parameter $\lambda = 0.0075$. The weights are then normalized such that $\sum_j \alpha_j = 1$. We store a maximum of $m = 400$ samples by replacing the sample with the smallest weight. The Fourier coefficients $\hat{w}$ of the penalty function w are computed as described in [11]. To detect the target, we perform a multi-scale search strategy [11,31] with 5 scales and a relative scale factor 1.02. The extracted confidences are maximized using the grid search followed by five Newton iterations, as described in Sect. 4.1.

The training of our continuous convolution filter f is performed by iteratively solving the normal equations (8). The work of [11] employed the Gauss-Seidel method for this purpose. However, this approach suffers from a quadratic complexity $\mathcal{O}(D^2)$ in the number of feature channels D. Instead, we employ the Conjugate Gradient (CG) [37] method due to its computational efficiency. Our numerical optimization scales linearly $\mathcal{O}(D)$ and is therefore especially suitable for high-dimensional deep features. In the first frame, we use 100 iterations to find an initial estimate of the filter coefficients $\hat{\mathbf{f}}$. Subsequently, 5 iterations per frame are sufficient by initializing CG with the current filter (see footnote 2).

4.3 Feature Point Tracking Framework

Here, we describe the feature point tracking framework based on our learning formulation. For computational efficiency, we assume a single-channel feature map ($D = 1$), e.g. a grayscale image, and a constant penalty function $w(t) = \beta$. Under these assumptions, the normal equations (8) form a diagonal system of equations. The filter coefficients are directly obtained as,

$$\hat{f}[k] = \frac{\sum_{j=1}^{M} \alpha_j \overline{X_j[k]\hat{b}[k]} \hat{y}_j[k]}{\sum_{j=1}^{M} \alpha_j \big|X_j[k]\hat{b}[k]\big|^2 + \beta^2}\,, \quad k = -K, \dots, K. \tag{10}$$

[3] We use imagenet-vgg-m-2048, available at: http://www.vlfeat.org/matconvnet/.

Here, we have dropped the feature dimension index for the sake of clarity. In this case (single feature channel and constant penalty function), the training equation (10) resembles the original MOSSE filter [5]. However, our continuous formulation has several advantages compared to the original MOSSE. Firstly, our formulation employs an implicit interpolation model, given by $\hat{b}$. Secondly, each sample is labeled by a continuous-domain confidence y_j, that enables sub-pixel information to be incorporated in the learning. Thirdly, our convolution operator outputs continuous confidence functions, allowing accurate sub-pixel localization of the feature point. In our experiments, we show that the advantages of our continuous formulation are crucial for accurate feature point tracking.

5 Experiments

We validate our learning framework for two applications: tracking of objects and feature points. For object tracking, we perform comprehensive experiments on three datasets: OTB-2015 [46], Temple-Color [32], and VOT2015 [29]. For feature point tracking, we perform extensive experiments on the MPI Sintel dataset [7].

Table 1. A baseline comparison when using different combinations of convolutional layers in our object tracking framework. We report the mean OP (%) and AUC (%) on the OTB-2015 dataset. The best results are obtained when combining all three layers in our framework. The results clearly show the importance of multi-resolution deep feature maps for improved object tracking performance.

	Layer 0	Layer 1	Layer 5	Layers 0, 1	Layers 0, 5	Layers 1, 5	Layers 0, 1, 5
Mean OP	58.8	78.0	60.0	77.8	70.7	*81.8*	82.4
AUC	49.9	65.8	51.1	65.7	59.0	*67.8*	68.2

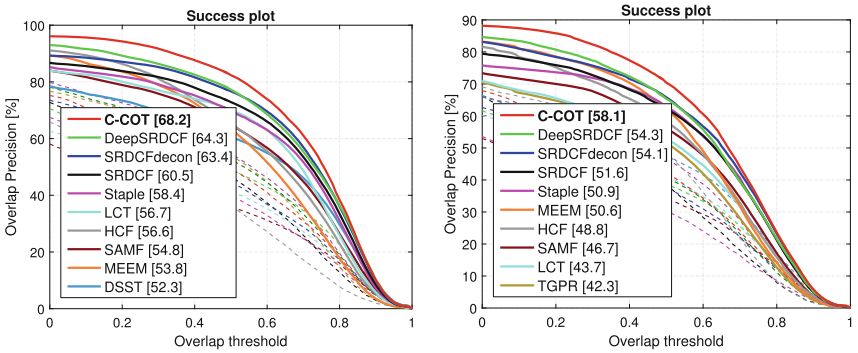
5.1 Baseline Comparison

We first evaluate the impact of fusing multiple convolutional layers from the deep network in our object tracking framework. Table 1 shows the tracking results, in mean overlap precision (OP) and area-under-the-curve (AUC), on the OTB-2015 dataset. OP is defined as the percentage of frames in a video where the intersection-over-union overlap exceeds a threshold of 0.5. AUC is computed from the success plot, where the mean OP over all videos is plotted over the range of thresholds $[0, 1]$. For details about the OTB protocol, we refer to [45].

In our experiments, we investigate the impact of the input RGB image layer (layer 0), the first convolutional layer (layer 1) and the last convolutional layer (layer 5). No significant gain in performance was observed when adding intermediate layers. The shallow layer (layer 1) alone provides superior performance compared to using only the deep convolutional layer (layer 5). Fusing the shallow and deep layers provides a large improvement. The best results are obtained

Table 2. A Comparison with state-of-the-art methods on the OTB-2015 and Temple-Color datasets. We report the mean OP (%) for the top 10 methods on each dataset. Our approach outperforms DeepSRDCF by 5.1 % and 5.0 % respectively.

	DSST	SAMF	TGPR	MEEM	LCT	HCF	Staple	SRDCF	SRDCFdecon	DeepSRDCF	**C-COT**
OTB-2015	60.6	64.7	54.0	63.4	70.1	65.5	69.9	72.9	76.7	*77.3*	**82.4**
Temple-Color	47.5	56.1	51.6	62.2	52.8	58.2	63.0	62.2	*65.8*	65.4	**70.4**



(a) OTB-2015 (b) Temple-Color

Fig. 2. Success plots showing a comparison with state-of-the-art on the OTB-2015 (a) and Temple-Color (b) datasets. Only the top 10 trackers are shown for clarity. Our approach improves the state-of-the-art by a significant margin on both these datasets.

when combining all three convolutional layers in our learning framework. We employ this three-layer combination for all further object tracking experiments.

We also compare our continuous formulation with the discrete DCF formulation by performing explicit resampling of the feature layers to a common resolution. For a fair comparison, all shared parameters are left unchanged. The layers (0, 1 and 5) are resampled with bicubic interpolation such that the data size of the training samples is preserved. On OTB-2015, the discrete DCF with resampling obtains an AUC score of 47.7 %, compared to 68.2 % for our continuous formulation. This dramatic reduction in performance is largely attributed to the reduced resolution in layer 1. To mitigate this effect, we also compare with only resampling layers 0 and 5 to the resolution of layer 1. This improves the result of the discrete DCF to 60.8 % in AUC, but at the cost of a 5-fold increase in data size. Our continuous formulation still outperforms the discrete DCF as it avoids artifacts introduced by explicit resampling.

5.2 OTB-2015 Dataset

We validate our Continuous Convolution Operator Tracker (C-COT) in a comprehensive comparison with 20 state-of-the-art methods: ASLA [25], TLD [26], Struck [21], LSHT [22], EDFT [14], DFT [41], CFLB [18], ACT [13], TGPR [19], KCF [24], DSST [9], SAMF [31], MEEM [47], DAT [40], LCT [36], HCF [35],

Staple [3] and SRDCF [11]. We also compare with SRDCFdecon, which integrates the adaptive decontamination of the training set [12] in SRDCF, and DeepSRDCF [10] employing activations from the first convolutional layer.

State-of-the-art Comparison: Table 2 (first row) shows a comparison with state-of-the-art methods on the OTB-2015 dataset.[4] The results are reported as mean OP over all the 100 videos. The HCF tracker, based on hierarchical convolutional features, obtains a mean OP of 65.5 %. The DeepSRDCF employs the first convolutional layer, similar to our baseline "Layer 1" in Table 1, and obtains a mean OP of 77.3 %. Our approach achieves the best results with a mean OP of 82.4 %, significantly outperforming DeepSRDCF by 5.1 %.

Figure 2a shows the success plot on the OTB-2015 dataset. We report the AUC score for each tracker in the legend. The DCF-based trackers HCF and Staple obtain AUC scores of 56.6 % and 58.4 % respectively. Among the compared methods, the SRDCF and its variants SRDCFdecon and DeepSRDCF provide the best results, all obtaining AUC scores above 60 %. Overall, our tracker achieves the best results, outperforming the second best method by 3.9 %.

Robustness to Initialization: We evaluate the robustness to initializations using the protocol provided by [46]. Each tracker is evaluated using two different initialization strategies: spatial robustness (SRE) and temporal robustness (TRE). The SRE criteria initializes the tracker with perturbed boxes, while the TRE criteria starts the tracker at 20 frames. Figure 3 provides the SRE and TRE success plots. Our approach obtains consistent improvements in both cases.

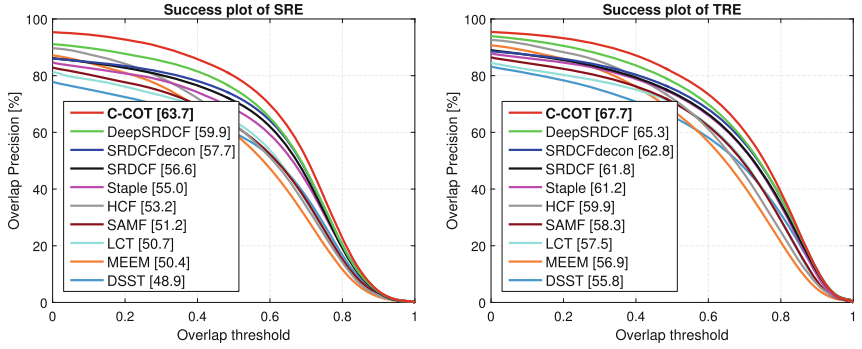


Fig. 3. An evaluation of the spatial (*left*) and temporal (*right*) robustness to initializations on the OTB-2015 dataset. We compare the top 10 trackers. Our approach demonstrates superior robustness compared to state-of-the-art methods.

5.3 Temple-Color Dataset

Here, we evaluate our approach on the Temple-Color dataset [32] containing 128 videos. The second row of Table 2 shows a comparison with state-of-the-art

[4] Detailed results are provided in the supplementary material.

methods. The DeepSRDCF tracker provides a mean OP score of 65.4 %. MEEM and SRDCFdecon obtain mean OP scores of 62.2 % and 65.8 % respectively. Different from these methods, our C-COT does not explicitly manage the training set to counter occlusions and drift. Our approach still improves the start-of-the-art by a significant margin, achieving a mean OP score of 70.4 %. A further gain in performance is expected by incorporating the unified learning framework [12] to handle corrupted training samples. In the success plot in Fig. 2b, our method obtains an absolute gain of 3.8 % in AUC compared to the previous best method.

5.4 VOT2015 Dataset

The VOT2015 dataset [29] consists of 60 challenging videos compiled from a set of more than 300 videos. Here, the performance is measured both in terms of accuracy (overlap with the ground-truth) and robustness (failure rate). In VOT2015, a tracker is restarted in the case of a failure. We refer to [29] for details. Table 3 shows the comparison of our approach with the top 10 participants in the challenge according to the VOT2016 rules [28]. Among the compared methods, RAJSSC achieves favorable results in terms of accuracy, at the cost of a higher failure rate. EBT achieves the best robustness among the compared methods. Our approach improves the robustness with a 20 % reduction in failure rate, without any significant degradation in accuracy.

Table 3. Comparison with state-of-the-art methods on the VOT2015 dataset. The results are presented in terms of robustness and accuracy. Our approach provides improved robustness with a significant reduction in failure rate.

	S3Tracker	RAJSSC	Struck	NSAMF	SC-EBT	sPST	LDP	SRDCF	EBT	DeepSRDCF	**C-COT**
Robustness	1.77	1.63	1.26	1.29	1.86	1.48	1.84	1.24	1.02	1.05	0.82
Accuracy	0.52	0.57	0.47	0.53	0.55	0.55	0.51	0.56	0.47	0.56	0.54

5.5 Feature Point Tracking

We validate our approach for robust and accurate feature point tracking. Here, the task is to track distinctive local image regions. We perform experiments on the MPI Sintel dataset [7], based on the 3D-animated movie "Sintel". The dataset consists of 23 sequences, featuring naturalistic and dynamic scenes with realistic lighting and camera motion blur. The ground-truth dense optical flow and occlusion maps are available for each frame. Evaluation is performed by selecting approximately 2000 feature points in the first frame of each sequence. We use the Good Features to Track (GFTT) [42] feature selector, but discard points at motion boundaries due to their ambiguous motion. The ground-truth tracks are then generated by integrating flow vectors over the sequence. The flow vectors are obtained by a bilinear interpolation of the dense ground-truth flow. We terminate the ground-truth tracks using the provided occlusion maps.

We compare our approach to MOSSE [5] and KLT [34,44]. The OpenCV implementation of KLT, used in our experiments, employs a pyramidal search [6]

to accommodate for large translations. For a fair comparison, we adopt a similar pyramid approach for our method and MOSSE, by learning an independent filter for each pyramid level. Further, we use the window size of 31×31 pixels and 3 pyramid levels for all methods. For both our method and MOSSE we use a learning rate of $\lambda = 0.1$ and set the regularization parameter to $\beta = 10^{-4}$. For the KLT we use the default settings in OpenCV. Unlike ours and the MOSSE tracker, the KLT tracks feature points frame-to-frame without memorizing earlier appearances. In addition to our standard tracker, we also evaluate a frame-to-frame version (Ours-FF) of our method by setting the learning rate to $\lambda = 1$.

For quantitative comparisons, we use the endpoint error (EPE), defined as the Euclidian distance between the tracked point and its corresponding ground-truth location. Tracked points with an EPE smaller than 3 pixels are regarded as inliers. Figure 4 (left) shows the distribution of EPE computed over all sequences and tracked points. We also report the average inlier EPE for each method in the legend. Our approach achieves superior accuracy, with an inlier error of 0.449 pixels. We also provide the precision plot (Fig. 4, center), where the fraction of points with an EPE smaller than a threshold is plotted. The legend shows the inlier ratio for each method. Our tracker achieves superior robustness in comparison to the KLT, with an inlier ratio of 0.886. Compared to MOSSE, our method obtains significantly improved precision at sub-pixel thresholds (< 1 pixel). This clearly demonstrates that our continuous formulation enables accurate sub-pixel feature point tracking, while being robust. Unlike the frame-to-frame KLT, our method provides a principled procedure for updating the tracking model, while memorizing old samples. The experiments show that already our frame-to-frame variant (Ours-FF) provides a spectacular improvement compared to the KLT. Hence, our gained performance is due to both the model update *and* the proposed continuous formulation. On a desktop machine, our Matlab code achieves real-time tracking of 300 points at a single scale, utilizing only a single CPU.

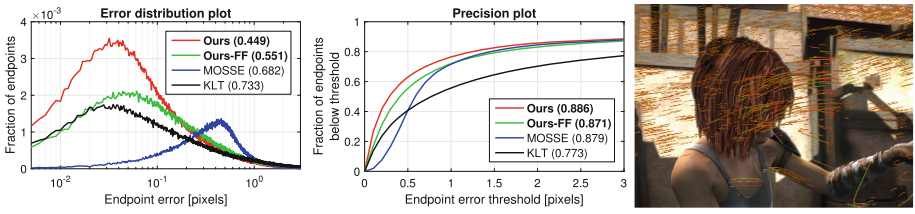


Fig. 4. Feature point tracking results on the MPI Sintel dataset. We report the endpoint error (EPE) distribution (*left*) and precision plot (*center*) over all sequences and points. In the legends, we display the average inlier EPE and the inlier ratio for the error distribution and precision plot respectively. Our approach provides consistent improvements, both in terms of accuracy and robustness, compared to existing methods. The example frame (*right*) from the Sintel dataset visualizes inlier trajectories obtained by our approach (red) along with the ground-truth (green). (Color figure online)

6 Conclusions

We propose a generic framework for learning discriminative convolution operators in the continuous spatial domain. We validate our framework for two problems: object tracking and feature point tracking. Our formulation enables the integration of multi-resolution feature maps. In addition, our approach is capable of accurate sub-pixel localization. Experiments on three object tracking benchmarks demonstrate that our approach achieves superior performance compared to the state-of-the-art. Further, our method obtains substantially improved accuracy and robustness for real-time feature point tracking.

Note that, in this work, we do not use any video data to learn an application specific deep feature representation. This is expected to further improve the performance of our object tracking framework. Another research direction is to incorporate motion-based deep features into our framework, similar to [20].

Acknowledgments. This work has been supported by SSF (CUAS), VR (EMC^2), CENTAURO, the Wallenberg Autonomous Systems Program, NSC and Nvidia.

References

1. Badino, H., Yamamoto, A., Kanade, T.: Visual odometry by multi-frame feature integration. In: ICCV Workshop (2013)
2. Baker, S., Matthews, I.A.: Lucas-kanade 20 years on: a unifying framework. IJCV **56**(3), 221–255 (2004)
3. Bertinetto, L., Valmadre, J., Golodetz, S., Miksik, O., Torr, P.H.S.: Staple: complementary learners for real-time tracking. In: CVPR (2016)
4. Boddeti, V.N., Kanade, T., Kumar, B.: Correlation filters for object alignment. In: CVPR (2013)
5. Bolme, D.S., Beveridge, J.R., Draper, B.A., Lui, Y.M.: Visual object tracking using adaptive correlation filters. In: CVPR (2010)
6. Bouguet, J.Y.: Pyramidal implementation of the lucas kanade feature tracker. Technical report Microprocessor Research Labs, Intel Corporation (2000)
7. Butler, D.J., Wulff, J., Stanley, G.B., Black, M.J.: A naturalistic open source movie for optical flow evaluation. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part VI. LNCS, vol. 7577, pp. 611–625. Springer, Heidelberg (2012)
8. Cimpoi, M., Maji, S., Vedaldi, A.: Deep filter banks for texture recognition and segmentation. In: CVPR (2015)
9. Danelljan, M., Häger, G., Shahbaz Khan, F., Felsberg, M.: Accurate scale estimation for robust visual tracking. In: BMVC (2014)
10. Danelljan, M., Häger, G., Shahbaz Khan, F., Felsberg, M.: Convolutional features for correlation filter based visual tracking. In: ICCV Workshop (2015)
11. Danelljan, M., Häger, G., Shahbaz Khan, F., Felsberg, M.: Learning spatially regularized correlation filters for visual tracking. In: ICCV (2015)
12. Danelljan, M., Häger, G., Shahbaz Khan, F., Felsberg, M.: Adaptive decontamination of the training set: a unified formulation for discriminative visual tracking. In: CVPR (2016)

13. Danelljan, M., Shahbaz Khan, F., Felsberg, M., van de Weijer, J.: Adaptive color attributes for real-time visual tracking. In: CVPR (2014)
14. Felsberg, M.: Enhanced distribution field tracking using channel representations. In: ICCV Workshop (2013)
15. Fernandez, J.A., Boddeti, V.N., Rodriguez, A., Kumar, B.V.K.V.: Zero-aliasing correlation filters for object recognition. TPAMI **37**(8), 1702–1715 (2015)
16. Fusiello, A., Trucco, E., Tommasini, T., Roberto, V.: Improving feature tracking with robust statistics. Pattern Anal. Appl. **2**(4), 312–320 (1999)
17. Galoogahi, H.K., Sim, T., Lucey, S.: Multi-channel correlation filters. In: ICCV (2013)
18. Galoogahi, H.K., Sim, T., Lucey, S.: Correlation filters with limited boundaries. In: CVPR (2015)
19. Gao, J., Ling, H., Hu, W., Xing, J.: Transfer learning based visual tracking with gaussian processes regression. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014. LNCS, vol. 8691, pp. 188–203. Springer, Heidelberg (2014). doi:10.1007/978-3-319-10578-9_13
20. Gladh, S., Danelljan, M., Shahbaz Khan, F., Felsberg, M.: Deep motion features for visual tracking. In: ICPR (2016)
21. Hare, S., Saffari, A., Torr, P.: Struck: structured output tracking with kernels. In: ICCV (2011)
22. He, S., Yang, Q., Lau, R., Wang, J., Yang, M.H.: Visual tracking via locality sensitive histograms. In: CVPR (2013)
23. Henriques, J.F., Caseiro, R., Martins, P., Batista, J.: Exploiting the circulant structure of tracking-by-detection with kernels. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part IV. LNCS, vol. 7575, pp. 702–715. Springer, Heidelberg (2012)
24. Henriques, J.F., Caseiro, R., Martins, P., Batista, J.: High-speed tracking with kernelized correlation filters. TPAMI **37**(3), 583–596 (2015)
25. Jia, X., Lu, H., Yang, M.H.: Visual tracking via adaptive structural local sparse appearance model. In: CVPR (2012)
26. Kalal, Z., Matas, J., Mikolajczyk, K.: P-N learning: bootstrapping binary classifiers by structural constraints. In: CVPR (2010)
27. Klein, G., Murray, D.: Parallel tracking and mapping for small AR workspaces. In: ISMAR (2007)
28. Kristan, M., Leonardis, A., Matas, J., Felsberg, M., Pflugfelder, R., Čehovin, L., Vojír, T., Häger, G., Lukěžič, A., Fernández, G.: The visual object tracking VOT 2016 challenge results. In: ECCV Workshop (2016)
29. Kristan, M., Matas, J., Leonardis, A., Felsberg, M., Čehovin, L., Fernández, G., Vojír, T., Nebehay, G., Pflugfelder, R., Häger, G.: The visual object tracking VOT 2015 challenge results. In: ICCV Workshop (2015)
30. Kristan, M., et al.: The visual object tracking VOT 2014 challenge results. In: Agapito, L., Bronstein, M.M., Rother, C. (eds.) ECCV 2014. LNCS, vol. 8926, pp. 191–217. Springer, Heidelberg (2015). doi:10.1007/978-3-319-16181-5_14
31. Li, Y., Zhu, J.: A scale adaptive kernel correlation filter tracker with feature integration. In: Agapito, L., Bronstein, M.M., Rother, C. (eds.) ECCV 2014. LNCS, vol. 8926, pp. 254–265. Springer, Heidelberg (2015). doi:10.1007/978-3-319-16181-5_18
32. Liang, P., Blasch, E., Ling, H.: Encoding color information for visual tracking: algorithms and benchmark. TIP **24**(12), 5630–5644 (2015)
33. Liu, L., Shen, C., van den Hengel, A.: The treasure beneath convolutional layers: cross-convolutional-layer pooling for image classification. In: CVPR (2015)

34. Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: IJCAI (1981)
35. Ma, C., Huang, J.B., Yang, X., Yang, M.H.: Hierarchical convolutional features for visual tracking. In: ICCV (2015)
36. Ma, C., Yang, X., Zhang, C., Yang, M.H.: Long-term correlation tracking. In: CVPR (2015)
37. Nocedal, J., Wright, S.J.: Numerical Optimization, 2nd edn. Springer, New York (2006)
38. Oquab, M., Bottou, L., Laptev, I., Sivic, J.: Learning and transferring mid-level image representations using convolutional neural networks. In: CVPR (2014)
39. Ovren, H., Forssén, P.: Gyroscope-based video stabilisation with auto-calibration. In: ICRA (2015)
40. Possegger, H., Mauthner, T., Bischof, H.: In defense of color-based model-free tracking. In: CVPR (2015)
41. Sevilla-Lara, L., Learned-Miller, E.G.: Distribution fields for tracking. In: CVPR (2012)
42. Shi, J., Tomasi, C.: Good features to track. In: CVPR (1994)
43. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: ICLR (2015)
44. Tomasi, C., Kanade, T.: Detection and Tracking of Point Features. Technical report (1991)
45. Wu, Y., Lim, J., Yang, M.H.: Online object tracking: a benchmark. In: CVPR (2013)
46. Wu, Y., Lim, J., Yang, M.H.: Object tracking benchmark. TPAMI **37**(9), 1834–1848 (2015)
47. Zhang, J., Ma, S., Sclaroff, S.: MEEM: robust tracking via multiple experts using entropy minimization. In: Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T. (eds.) ECCV 2014, Part VI. LNCS, vol. 8694, pp. 188–203. Springer, Heidelberg (2014)
48. Zografos, V., Lenz, R., Ringaby, E., Felsberg, M., Nordberg, K.: Fast segmentation of sparse 3D point trajectories using group theoretical invariants. In: Cremers, D., Reid, I., Saito, H., Yang, M.-H. (eds.) ACCV 2014. LNCS, vol. 9006, pp. 675–691. Springer, Heidelberg (2015)