

# Synthetic data generation for end-to-end thermal infrared tracking

Lichao Zhang, Abel Gonzalez-Garcia, Joost van de Weijer, Martin Danelljan, and Fahad Shahbaz Khan

**Abstract**—The usage of both off-the-shelf and end-to-end trained deep networks have significantly improved performance of visual tracking on RGB videos. However, the lack of large labeled datasets hampers the usage of convolutional neural networks for tracking in thermal infrared (TIR) images. Therefore, most state of the art methods on tracking for TIR data are still based on hand-crafted features. To address this problem, we propose to use image-to-image translation models. These models allow us to translate the abundantly available labeled RGB data to synthetic TIR data. We explore both the usage of paired and unpaired image translation models for this purpose. These methods provide us with a large labeled dataset of synthetic TIR sequences, on which we can train end-to-end optimal features for tracking. To the best of our knowledge we are the first to train end-to-end features for TIR tracking. We perform extensive experiments on VOT-TIR2017 dataset. We show that a network trained on a large dataset of synthetic TIR data obtains better performance than one trained on the available real TIR data. Combining both data sources leads to further improvement. In addition, when we combine the network with motion features we outperform the state of the art with a relative gain of over 10%, clearly showing the efficiency of using synthetic data to train end-to-end TIR trackers.

**Index Terms**—Visual tracking, thermal infrared, Deep learning, Generative Networks

## I. INTRODUCTION

VISUAL tracking aims to estimate a trajectory of an object through a video based on only one bounding box annotation at the beginning of the sequence. Tracking is important for applications in surveillance [1], video understanding [2] and robotics [3]. One of the main challenges of tracking is the limited data problem: the tracker should be able to track an object based on only a single annotated bounding box. The success of a tracker is therefore very dependent on the quality of the discriminative features which are used by the tracker.

Recently, specialized tracking subproblems have emerged. Among these is the field of tracking in thermal infrared (TIR) images, whose importance is further increasing due to improvements of thermal infrared sensors in resolution and quality [4], [5], [6]. The advantage of thermal images is that they are not influenced by the illumination variations

and shadows, and objects can be distinguished from the background as the background is normally colder. In addition, thermal infrared tracking can be used in total darkness, where visual cameras have no signal. Considering these advantages, thermal infrared tracking has a wide range of applications in car and pedestrian surveillance systems as well as various defense systems [7].

In recent years, Discriminative Correlation Filter (DCF) based methods [8], [9], [10] have shown to provide excellent tracking performance on existing benchmarks [11], [12], [13]. The DCF based trackers learn a correlation filter from example patches to discriminate between the target and background appearance. Further, the DCF based framework efficiently utilizes all spatial shifts of the training samples by exploiting the properties of circular correlation to train and apply a discriminative classifier in a sliding window fashion. Lately, the DCF based framework has been significantly advanced by employing high-dimensional visual features [9], [14], [15], powerful learning methods [16], [10], reducing boundary effects [17], and accurate scale estimation [18]. Due to their superior performance in RGB tracking, some of these methods have also been applied with success to TIR [10], [17].

Recently, deep learning has revolutionized the field of computer vision significantly advancing the state-of-the-art in many applications [19]. Generally the deep networks are trained on a large amount of labeled training data. Despite its astounding success, the impact of deep learning on generic visual tracking (RGB data) has been limited. One of the key issues when employing deep features for tracking is the unavailability of large-scale labeled tracking data for training. Further, the tracking model is desired to be learned using a single labeled frame. Therefore, most existing deep learning based DCF trackers [20], [21], [10] employ deep features pre-trained on the ImageNet dataset [22] for image classification task. Other approaches [23], [16] have investigated the integration of DCF in a deep network by adapting the end-to-end philosophy, but did not result in major improvements over features from pre-trained networks.

Even more than for RGB tracking, introducing deep learning to TIR tracking is hampered by the absence of large datasets. The datasets which are available for thermal infrared videos are relatively small. Moreover, there is no ImageNet counterpart of infrared still images on which a large network could be pre-trained. Therefore, the usage of hand-crafted features remains dominant for TIR tracking. For instance, the top three trackers in VOT-TIR2017 [6], [12] are still exploiting hand-crafted features. The winner [24] of VOT-TIR2017 challenge employs HOG [25] and motion features. Further, the other

Lichao Zhang, Abel Gonzalez-Garcia, Joost van de Weijer, are with the Computer Vision Center Barcelona, Edifici O, Campus UAB, 08193, Bellaterra, Spain.

Lichao Zhang and Joost van de Weijer are also with the Universitat Autònoma de Barcelona, Spain.

Martin Danelljan is with the Department of Electrical Engineering, Linköping University, Sweden

Fahad Shahbaz Khan is with the Inception Institute of Artificial Intelligence, Abu Dhabi, UAE and Department of Electrical Engineering, Linköping University, Sweden

top-performing methods [26], [4] are based on hand-crafted features. The success of these methods show that hand-crafted features are still the best choice for TIR tracking.

Deep learning has also resulted in fast progress in generative models which are able to generate samples from complex image distributions [27]. These models have been further extended to image-to-image translation models [28] which allow to learn mappings between image domains. A further extension of this work allows to learn mappings between unpaired domains [29], which is based on the observation that transferring an image to another domain and then transferring it back to the first domain should result in the same image which was provided as input. One of the more interesting applications of these generative networks is that they can be used to construct synthetic datasets of small data domains, such as TIR. In this work, we show that labeled data from RGB can be translated to TIR data as well as the labels.

In this work we tackle the key limited-data problem for TIR-tracking by utilizing recent developments in image-to-image translation methods [28], [29]. The idea is to automatically transfer RGB tracking videos to the TIR domain. We can then automatically transfer the labels from these RGB videos to the synthetic TIR videos. The resulting data can then be used to extract discriminative deep features for the TIR domain. The advantage is that we can generate the TIR counterpart of the available RGB tracking datasets which are much larger compared to the current TIR tracking datasets. The main contributions of the paper are:

- We address the scarcity of labeled data for TIR tracking. Therefore, we propose a framework which transfers RGB data to synthetic TIR data. The labels available for the RGB data are also transferred to the TIR data, resulting in a large synthetic TIR data set for tracking.
- We are the first to perform end-to-end training for TIR tracking, showing that this can significantly improve results (see Table III). We also show that a tracker trained on only synthetic data can outperform trackers trained on available labeled TIR data (see Fig. 1).
- We perform extensive evaluations on the latest TIR tracking challenge [6] verifying the efficiency of our different models trained on synthetic TIR datasets. We show that when combined with motion features our method obtains state of the art on the TIR tracking challenge.

The rest of the paper is organized as follows. In section II we briefly discuss related work. In section IV we introduce the standard correlation filter and the current end-to-end deep correlation filter. In section V we describe the prevalent generative adversarial networks and present our generated synthetic tracking videos. In section VI we present our experiments on standard thermal infrared tracking dataset. In section VII we conclude our work and plan our further research.

## II. RELATED WORK

### A. DCF tracking

In recent years, discriminative correlation filter (DCF) based tracking methods have shown excellent performance in terms of accuracy and robustness on benchmark tracking

datasets [12], [11]. The DCF based trackers aim at learning a correlation filter in an online fashion from example image patches to discriminate between the target and background appearance. The seminal work of [8] was restricted to a single feature channel (grayscale image). Later, the DCF framework was extended to use multi-dimensional hand-crafted features by [30], [9], [14], such as HOG [25] and Color Names [31]. Some of the recent advances in DCF frameworks can be attributed to reducing boundary effects [17], robust scale estimation [18], integrating context [32], and adding a long-term memory component [33].

Even after more than five years of flourishing, discriminative correlation filter based tracking is still the mainstream in single object tracking. Recent modifications on DCF include: Mueller et al. [32] sample four context patches around the target and incorporate these to regularize the regression function, which has the same effect as hard negative mining. Alan et al. [34] enlarge the search region and improve tracking of non-rectangular objects by using spatial maps to restrict the correlation filter. They also give the learned filter adaptive channel-wise weights, which improves the quality of the filter. Kiani et al. [35] use a mask to crop the object in the spatial domain and get a new closed-form solution of the correlation filter in the Fourier domain by embedding the mask matrix into the formulation. This yields significantly more shifted examples unaffected by boundary effects.

Compared to hand-crafted features (e.g. HOG [25], intensity and Color Names [31]), deep CNN features significantly improve the robustness of the tracker against geometric variations [36], resulting in a significant improvement of the performance. This is mainly caused by the high discriminative power of deep features, since CNNs are trained on large datasets such as ILSVRC2012 [22]. Later, Ma et al. [20] propose to encode the target appearance on several convolutional layers and each layer has a corresponding correlation filter. This hierarchical architecture locates targets by maximizing the response of each layer with different weights. They find an optimized position in a coarse to fine way. Directly using different layers may not take full advantage of the CNN features because of the discrete distribution of features. To exploit the continuity between different layers of networks, Danelljan et al. [21] propose to learn a convolution operator in the continuous spatial domain called CCOT. As CCOT is very slow, Danelljan et al. [10] propose to factorize the convolution operator to reduce the dimensions of feature maps. Then they use GMM to generate samples, which significantly accelerate the tracker, enabling the tracker to run in real-time, while still maintaining the same or higher accuracy.

### B. TIR tracking

Currently, the leading TIR trackers still employ hand-crafted features in their models. Yu et al. [37] propose structural learning on dense samples around the object. Their tracker uses edge and HOG features and transfers them into the Fourier domain, to obtain a real-time tracker. Later they extend this work, called DSLT [24], by integrating HOG [25] and motion features. With this tracker they won the VOT-TIR2017

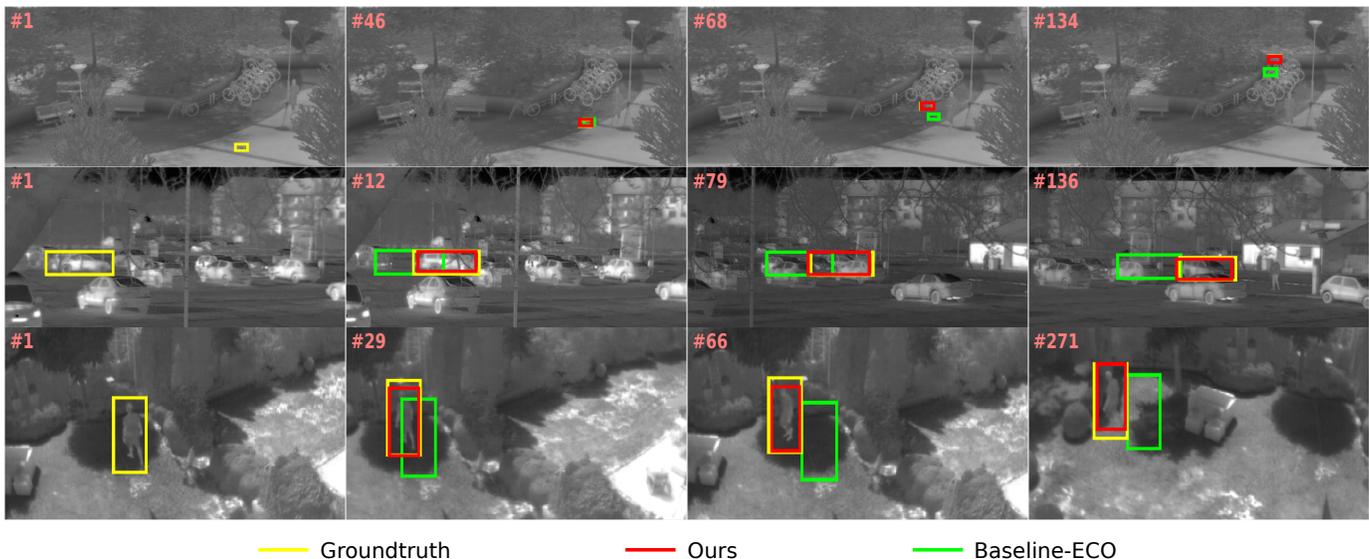


Fig. 1: Qualitative comparison of our approach trained on generated data only (red) with baseline ECO [10] (green) on the *quadcopter2*, *car2* and *garden* videos. The ground truth bounding box is provided in yellow. Owing to the synthetic TIR data our model is able to follow the object successfully in case of out-of-plane rotation, partial occlusion and scale changes.

challenge [6]. Another TIR tracker, called EBT [26], uses edge features to devise an objectness measure specific for each instance. This enables the generation of high quality object proposals and the use of richer features. Concretely, for each proposal they extract a 2640-dimensional histogram feature as well as a 5-level pyramid computed from the intensity channel. They achieve the runner-up position in the VOT-TIR2017 challenge. SRDCFir [4] extends the SRDCF [17] tracker for TIR data by adding motion features. SRDCF is a DCF-based tracker that introduced a spatial regularization function to penalize those DCF coefficients that reside outside the target region, which mitigates the damaging boundary effects present in the traditional DCF.

Another branch for TIR tracking combines the input TIR data with the visual modality, concretely with the image intensity given as a grayscale image. For example, Li et al. [38] propose an adaptive fusion scheme to incorporate information from grayscale images and TIR videos during tracking. Similarly, the approach in [39] samples a set of patches around the object and extracts a joint sparse representation in both grayscale and TIR modalities.

The usage of generating other modalities was pioneered by Hoffman et al. [40]. They used generation of depth data to improve classification on the abundant-data modality (RGB), whereas we use data generation as a source of labeled data for the scarce-data modality (TIR). Xu et al. [41] use a network which generates TIR images to pre-train the weights. These weights are then applied in a network which is used on RGB data with the aim to improve tracking of pedestrians. Other than them, we use the generation of TIR data for data augmentation; we create large synthetic labeled data sets of TIR data to be able to train end-to-end features for TIR data.

### C. Adversarial image-to-image translation

Generative Adversarial Networks (GANs) [27] have achieved promising results in several tasks such as image generation [42], image editing [43], and representation learning [44]. The conditional variants of GANs [45] enable to condition the image generation on a selected input variable, for example, an input image. In this case, the task becomes image-to-image translation, and this is the variant we use here. The general method of Isola et al. [28], pix2pix, was the first GAN-based image-to-image translation work that was not designed for a specific task (e.g. colorization [46]). The architecture is based on an encoder-decoder with skip connections [47] and it is trained using a combination of two losses: a conditional adversarial loss [27] and a loss that maps the generated image close to the corresponding target image. This method achieves excellent results, but requires matching pairs of training images, which limits the applicability of the model as such data might not be easily accessible. In order to overcome this limitation, Zhu et al. [29] extended this model to the case in which paired data is not available. Their method, called CycleGAN, relies on the assumption that mapping an image from the input domain to the target and then back to the input (i.e. the cycle) should result in the identity function. Based on this, they add a cycle consistency loss that enforces the correct reconstruction of the input image resulting of the composition of the two mappings. They demonstrate the effectiveness of their method for multiple tasks such as edges to real images or photo enhancement. In this paper, we use image-to-image translation to generate a synthetic large-scale TIR tracking dataset from a labeled RGB dataset, with the goal of learning better deep features for tracking.

## III. METHOD OVERVIEW

We aim to train end-to-end deep features for tracking in TIR data. However, to train effective deep features for TIR data, we

need a large dataset of labeled TIR videos. Unfortunately, the amount of labeled TIR data is very scarce. To the best of our knowledge, only BU-TIV dataset [48] contains a considerable amount of labeled TIR videos, but most of them depict only one object class (pedestrian). Therefore, most state of the art TIR tracking methods are still based on hand-crafted features [24], [26], [4]. On the other hand, there are vast amounts of RGB videos labeled for tracking [11], [12]. One solution could therefore be to use the pre-trained features which are optimal for tracking in RGB data for TIR data. However, this is unlikely to be optimal because TIR and RGB data differ significantly.

To illustrate the difference in nature of RGB and TIR data we measure the average activation of the 96 filters of the first layer of a pre-trained AlexNet on the KAIST dataset. This dataset contains both RGB and TIR images of the same scenes (a similar study for depth images has been performed in [49]). The pre-trained network is trained to recognize objects in RGB images (i.e. on ImageNet). We present the results in Fig. 2. The graph shows the average activation of the filters in descending order. When applied to data which is similar to that on which the network is trained, the average activations tend to produce a uniform distribution. This can be seen for RGB images where most filters yield the same average activation and only a few filters differ from this pattern. When we perform the same experiment on TIR data the pattern changes. We can now observe clear differences between filters which have a higher average activation and filters which have lower average activation. This shows that these filters are probably not optimal for TIR tracking. When we look at the exact filters which have low and high activation, we see that low frequency patterns (blobs and edges) are prevalent for the TIR data, whereas high-frequency filters are seldom found in TIR data. This is not surprising since most textures, responsible for most of the high-frequency content of images, do not appear in TIR data. In conclusion, given the different nature of the image statistics of RGB data and TIR data it is probable that a network which is trained on TIR data would outperform a network trained on RGB data.

In this paper, we aim to address the problem of data scarcity of labeled videos for tracking in TIR data. We do this by exploiting the vast amount of labelled RGB videos which are available, in combination with recent advances in image-to-image translation techniques. We will use these image-to-image translation models to transfer large labeled RGB datasets to synthetic TIR dataset together with the tracking annotations. As a result we now have a large labeled synthetic TIR dataset. We use this synthetic TIR dataset to train end-to-end deep networks to obtain optimal TIR features for tracking. Then we plug the optimal TIR feature model into a state-of-the-art tracker. Here we use ECO [10]. An overview of our method is provided in Fig. 3. In the following section we detail the various parts of our algorithm.

#### IV. DEEP LEARNING FEATURES FOR CORRELATION FILTER TRACKING

In this section, we introduce the standard correlation filter and the current end-to-end deep correlation filter. Then we

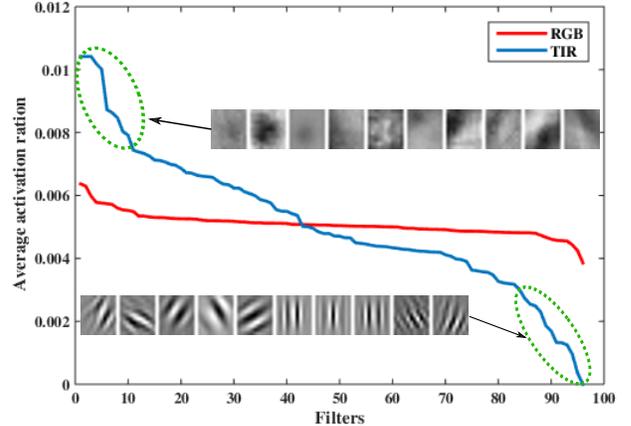


Fig. 2: Average activation of filters from the first layer of pre-trained AlexNet [19] on the test set of KAIST [50] for RGB and TIR images.

describe Efficient Convolution Operators (ECO) [10] which we will use as the correlation filter for our experiments.

##### A. Correlation Filter Tracking

The conventional discriminative correlation filters (DCF) formulation [9], learns a linear correlation filter  $f$  that discriminates the target appearance from the background. The target location is predicted by applying the correlation filter to a sample feature map. The desired filter  $f$  can be obtained by minimizing a least squares objective:

$$E(f) = \left\| \sum_{d=1}^D f^d * x^d - y \right\|^2 + \lambda \sum_{d=1}^D \|f^d\|^2. \quad (1)$$

Here  $*$  denotes circular correlation.  $x^d$  denotes feature maps of training samples  $x$ , where the layer  $d \in \{1, \dots, D\}$ .  $f^d$  denotes channel  $d$  of filter  $f$ .  $y$  is the regression target and  $\lambda$  is a regularization weight to control over-fitting. A closed-form solution is obtained in the Fourier domain,

$$\hat{f}^d = \frac{\hat{x}^d \hat{y}^*}{\sum_{d=1}^D \hat{x}^d (\hat{x}^d)^* + \lambda}. \quad (2)$$

Where the  $\hat{y}^*$  denotes the complex conjugate of the discrete Fourier transform  $\mathcal{F}(y)$ .

Recently, researchers have proposed several methods for end-to-end training of features for tracking: CFNet [23], DCFNet [51], and CFCF [52]. All use a two-branch Siamese network, of which one branch is used to compute the optimal correlation filter, which is applied in the other branch to obtain the response map (see Fig. 3). Both branches share the weights of their convolutional layers. Training is performed with paired images from same video. It backpropagates the gradients through the discriminative correlation filter layer (DCFL) with a closed-form solution [23]. Surprisingly, trackers based on end-to-end training only slightly outperformed off-the-shelf features. It should be noted that all end-to-end trackers train on RGB datasets, mainly on the ImageNet Large Scale Visual

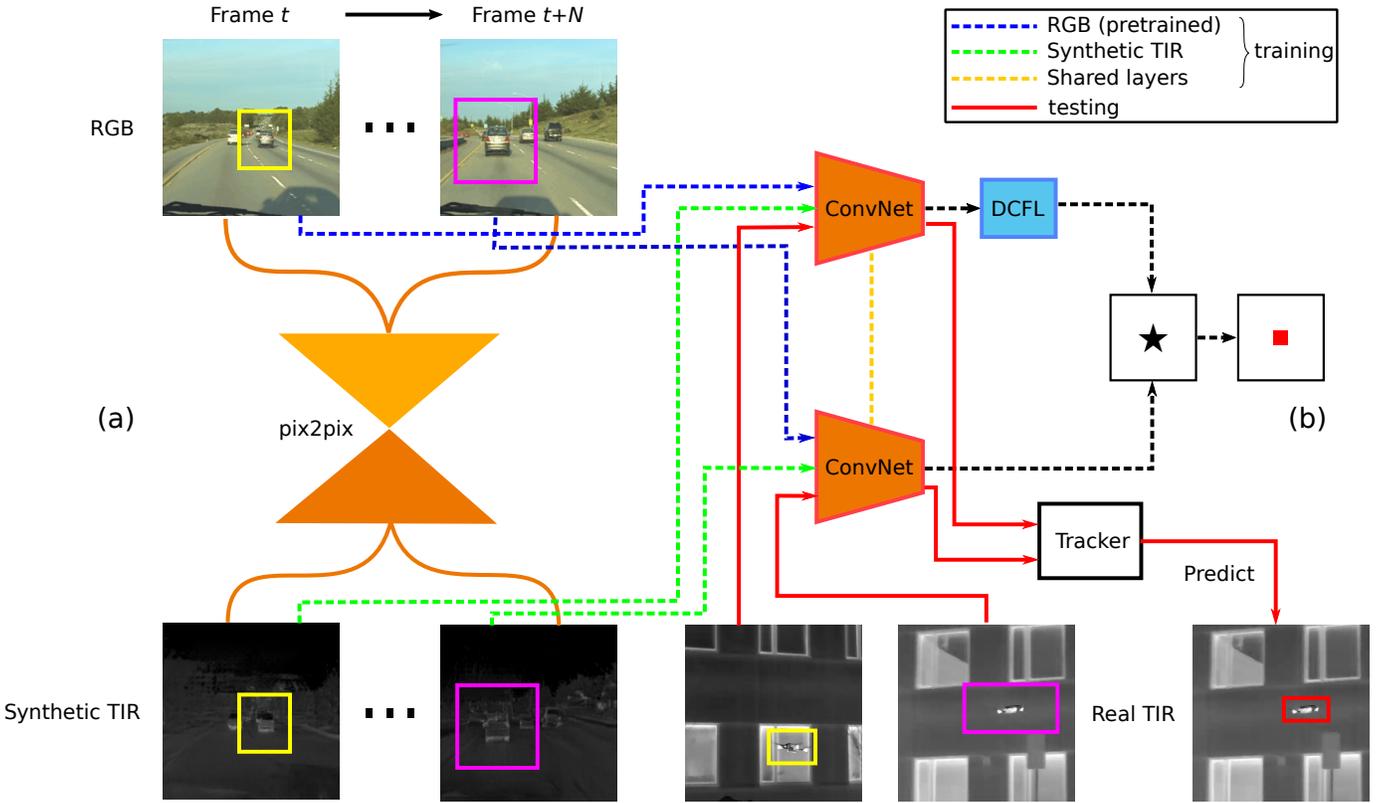


Fig. 3: Overview of our approach. (a) Image-to-image translation component (proposed in [28]) for generating a large labeled synthetic TIR tracking dataset. We use blue dashed line to represent the baseline RGB training model and the green dashed line represents our proposed synthetic data training model. After the translation of RGB data to TIR data, we acquire enough suitable data for end-to-end training networks for TIR tracking. (b) Two-branch architecture for training the network to obtain adaptive features for TIR tracking (proposed in [23]). The optimal correlation filter is computed in the discriminative correlation filter layer (DCFL) for the image processed in the upper branch. This filter is then applied on the image in the bottom branch.

Recognition Challenge (ILSVRC15) [22], and no results for end-to-end tracking on other modalities like TIR are available.

In this paper, we use the end-to-end CFNet training procedure proposed by Bertinetto et al. [23]. This method obtains stable and fast network training due to their Fourier domain implementation of the discriminative correlation filter layer. Other than them we will apply it to TIR tracking. Since current available datasets for TIR tracking are rather small, we propose in the next section our approach to generating synthetic TIR tracking data from labeled RGB tracking data.

### B. Efficient Convolution Operators

Previously, we have explained how to train end-to-end features for tracking. These features can be used in different discriminative correlation filter methods. In our work we use the Efficient Convolution Operator (ECO) [10] method, shown to obtain state-of-the-art results while being computationally efficient. However, its original implementation is based on features extracted from a pre-trained CNN model trained on the ImageNet 2012 classification dataset [22]. Even though these features are extracted from a model which is trained for image classification, ECO obtains excellent results for tracking. In our experiments we combine ECO with the end-to-end trained features for TIR tracking.

The ECO tracker aims at combining shallow and deep features by learning a multi-channel continuous convolution filter in a joint optimization scheme across all feature channels. Furthermore, it learns a projection matrix, to reduce the dimensionality of high-dimensional features. Here we briefly describe the training and inference procedures applied in ECO.

ECO learns the target model parameters based on a set of training samples  $\{x_j\}_1^M$  and corresponding labels  $\{y_j\}_1^M$ . The label function  $y_j$  consists of the desired target scores at all spatial locations in the corresponding training sample  $x_j$ . It is defined as a periodically repeated Gaussian function centered at the sample location [21]. Each training sample contains multiple feature layers  $x_j^d \in \mathbb{R}^{N_d \times N_d}$ , where  $N_d$  is the spatial resolution of layer  $d \in \{1, \dots, D\}$ . These feature layers correspond to both shallow and deep features of varying resolutions. The tracker predicts the target location using the target score operator, defined as

$$S_{f,P}\{x\} = \sum_{d=1}^D f^d * PJ_d\{x^d\}. \quad (3)$$

Here,  $x$  is the input sample and  $f$  is the learned filter that predicts the detection score function  $S_{f,P}\{x\}$  of the target. The sample  $x$  is first interpolated to the continuous domain using the operator  $J_d$ , by applying a cubic spline kernel in



Fig. 4: Results for the two image translation methods considered: pix2pix and CycleGAN. The video frames are taken from the test set of KAIST [50], and have not been seen during training.

the Fourier domain (see [21] for more details). The projection matrix  $P$  is then applied to reduce the dimensionality of the feature space.

The detection score operator is learnt via minimization of a least squares objective,

$$E(f) = \sum_{j=1}^M \alpha_j \|S_{f,P}\{x_j\} - y_j\|^2 + \sum_{d=1}^D \|w f^d\|^2 + \lambda \|P\|_F^2. \quad (4)$$

Here, the projection and filter are regularized by a constant  $\lambda$ . The spatial regularization weight function  $w$  is employed to mitigate the effects of periodic repetition [17]. Each sample  $x_j$  is weighted by  $\alpha_j$ , based on a learning rate parameter  $\gamma$ . The label functions  $\{y_j\}_1^M$  are set to Gaussian functions centered at the target location.

Using Parseval's formula an equivalent loss is obtained as,

$$E(f) = \sum_{j=1}^M \alpha_k \|S_{f,P}\{\widehat{x_j}\} - \hat{y}_j\|^2 + \sum_{d=1}^D \|\hat{w} * \hat{f}^d\|^2 + \lambda \|P\|_F^2. \quad (5)$$

Here  $\hat{\cdot}$  denotes the Fourier coefficients. We learn the projection matrix  $P$  jointly with the filter  $f$  in the first frame by applying Gauss-Newton and adopting the Conjugate Gradient method [53] for each iteration. In subsequent frames, the resulting normal equations are efficiently solved using the method of Conjugate Gradients, assuming a fixed  $P$ . For more details, we refer to [10], [21].

## V. GENERATING TIR IMAGES

In this section we discuss image-to-image translation methods and compare them for the task of transferring RGB to synthetic TIR data.

### A. Image-to-image translation methods

We use two different image-to-image translation methods to transform labeled RGB videos into labeled TIR videos. First, we use pix2pix [28], which requires paired training data. Therefore, we need matching frames in both RGB and TIR, which we can obtain from multispectral video datasets such as KAIST [50]. Second, we use CycleGAN [29], an extension on pix2pix that can be trained from unpaired data. As a consequence, any videos in the RGB and TIR modalities can be used to train CycleGAN. Despite the higher availability of unpaired training data, we expect the weaker supervision of CycleGAN to result in synthesized TIR images of lower quality. In this section, we present both translation methods and experimentally confirm this intuition. In later sections, we generate TIR data using only pix2pix given its empirically superior performance.

Both methods are based on Generative Adversarial Networks (GANs) [27] conditioned on input images. GANs consist of two networks, generator  $G$  and discriminator  $D$ , that compete against each other. The generator tries to generate samples that resemble the original data distribution, whereas the discriminator tries to detect whether samples are real or have been generated by  $G$ . When the GAN architecture is conditioned on an input image, the task becomes image-to-image translation. In our case, the input image is a color frame from an RGB video and the target is the matching frame in the TIR modality.

1) *Paired - pix2pix*: pix2pix [28] is an effective, task-agnostic method that can be applied to translate between many domain pairs, including maps to satellite pictures, edge maps to real pictures, or grayscale images to color images. The generator is based on an encoder-decoder architecture with skip connections (U-Net [47]). The discriminator is a

convolutional PatchGAN [54], which classifies each local image patch independently, making it especially suited for modifying textures or styles.

Let  $x$  be an image from the input domain  $X$  and  $y$  an image from the target domain  $Y$ . In pix2pix, both the generator and discriminator are conditioned on the input image  $x$ . The conditional GAN objective function is defined as

$$\begin{aligned} \mathcal{L}_{cGAN}(G, D) &= \mathbb{E}_{x,y}[\log D(x, y)] \\ &+ \mathbb{E}_{x,z}[\log(1 - D(x, G(x, z)))], \end{aligned} \quad (6)$$

where  $z$  is a random noise vector used as input for the generator. Additionally, pix2pix also includes an L1 loss to increase the sharpness of the output images

$$\mathcal{L}_{L1}(G) = \mathbb{E}_{x,y,z}[\|y - G(x, z)\|_1]. \quad (7)$$

The final objective function is the weighted sum of these two losses. Following the original adversarial training [27],  $G$  tries to minimize this final objective while  $D$  tries to maximize it:

$$G^* = \arg \min_G \max_D \mathcal{L}_{cGAN}(G, D) + \lambda \mathcal{L}_{L1}(G). \quad (8)$$

We translate an RGB video to TIR by applying pix2pix independently for each video frame. The original model of [28] achieves mild stochasticity in its outputs by keeping the dropout layers at test time, which are normally used only during training. In our case, this is not only unnecessary but also damaging, as it makes the output video less stable. For this reason, we only use dropout layers during training.

2) *Unpaired - CycleGAN*: Paired data might be hard to come by for particular tasks including RGB to TIR conversion, as the amount of paired videos in both modalities is rather limited. Zhu et al. [29] present CycleGAN, a method for learning to translate between image domains when paired examples are not available. The main idea consists in adding a cycle consistency loss, based on the assumption that mapping an image  $x \in X$  to domain  $Y$  and back to  $X$  should leave it unaltered. For this reason, besides the classic generator  $G: X \rightarrow Y$ , CycleGAN also learns a generator to perform the inverse mapping  $F: Y \rightarrow X$ . The method is then trained with a weighted combination of an unconditional adversarial loss [27] and the cycle consistency loss in both directions

$$\begin{aligned} \mathcal{L}_{cyc}(G, F) &= \mathbb{E}_x[\|F(G(x)) - x\|_1] \\ &+ \mathbb{E}_y[\|G(F(y)) - y\|_1], \end{aligned} \quad (9)$$

See [29] for more details. As in pix2pix, we apply CycleGAN independently per frame, and we remove the dropout layers at test time to generate a more stable video output.

## B. Datasets

We consider multiple datasets for training our image translation methods, spanning the two presented supervision levels: paired and unpaired. Table I details the number of images of all the considered datasets. Among the paired datasets, the biggest and most relevant is KAIST Multispectral Pedestrian Dataset [50], which contains a significant amount of aligned images in the RGB and TIR modalities, captured from a moving vehicle in different urban environments and under different

Type	Dataset	Number of images	
		RGB	TIR
Paired	KAIST [50]	50,184	50,184
	CVC-14 [55]	8,473	8,473
	OSU Color Thermal [56]	8,545	8,545
	VAP Trimodal [57]	5,924	5,924
	Bilodeau [58]	7,821	7,821
	LITIV2012 [59]	6,325	6,325
	total	87,088	87,088
Unpaired	VOT2016 [60]	21,455	-
	VOT2017 [6]	4,049	-
	OTB [11]	58610	-
	ASL [61]	-	6,490
	Long-term [62]	-	47,423
	InfAR [63]	-	46,121
	total	84,114	100,034

TABLE I: Datasets used for training the image-to-image translation models. We test all models using a subset of three videos from the official test set of KAIST [50].

lighting conditions. We follow the official data split [50] as in [28] and use all the frames from training videos for training. We evaluate both image translation methods using 3 randomly left out videos from the test set, amounting to 5,728 images. Train and test sets have no videos in common.

Other paired datasets include images of people captured under different conditions: pedestrians during day or night (CVC-14 [55]), static cameras at a busy intersection (OSU Color-Thermal Database [56]) or in different positions and zooms (LITIV2012 dataset [59]), interactions in indoor scenes with controlled lighting settings (VAP Trimodal People Segmentation Dataset [57]), or moving in different planes (Bilodeau et al. [58]). This amounts to a total of 87K image pairs.

We use all paired datasets to train both pix2pix and CycleGAN. Additionally, we collect an RGB-TIR unpaired dataset as extra training data for CycleGAN. As RGB data we include all the sequences from VOT2016 [60], VOT2017 [6], and OTB [11]. As TIR data we include the TIR images from ASL [61], Long-term [62], and InfAR [63]. This amounts to a total of about 230K images, almost  $5\times$  more images than the paired training dataset.

## C. Implementation details

We train all networks in pix2pix and CycleGAN from scratch, initializing the weights from a Gaussian distribution with zero mean and standard deviation of 0.02. We use the same network architectures as in the original papers [28], [29]. As in [28], we apply random jittering by slightly enlarging the input image and then randomly cropping back to the original size. We train pix2pix for 10 epochs, with batch size 4 and learning rate 0.0002. CycleGAN is trained for 3 epochs, batch size 2 and learning rate 0.0002. Note that both models are trained for an equivalent number of iterations given the size of their training sets.

## D. TIR image translation quality

In order to test the two image translation methods considered we select a random subset of the test set of KAIST [50],

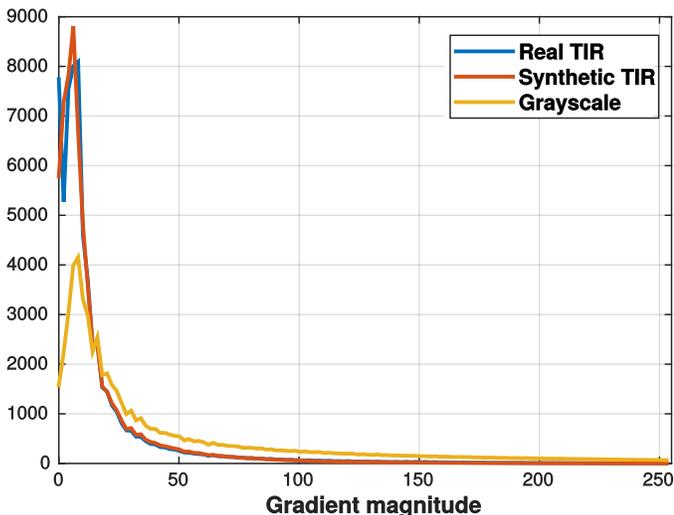


Fig. 5: Histogram of the gradient magnitude for real and synthetic TIR data computed on the test set of KAIST [50]. For comparison we have also added the gradient magnitude histogram for grayscale images from which the synthetic dataset has been generated.

amounting to about 10% of the entire dataset. We translate the RGB videos into TIR using pix2pix or CycleGAN, and then compute the Euclidean distance of the translations with the TIR ground-truth images. Finally, we average the distance for all frames. pix2pix obtains an average distance of 35.3, whereas CycleGAN obtains 69.5. This demonstrates the superiority of pix2pix for this task, showing how a paired training signal is more valuable than the unpaired counterpart, despite the bigger training dataset of the latter. Fig. 4 shows a qualitative comparison of both approaches. We can observe how the translated images using pix2pix are clearly superior to those translated by CycleGAN. Moreover, they look remarkably similar to the ground-truth TIR images, confirming the validity of the proposed data augmentation approach. Therefore, we select pix2pix as our method to generate TIR tracking data from RGB videos.

In addition we compare the statistics of the image gradients of real TIR data and synthetic TIR data generated by pix2pix. The histogram of the gradient magnitude for both datasets on the test set of KAIST is provided in Fig. 5. We have also added the gradient magnitude of the grayscale images from which the synthetic dataset is generated. The results show that the gradient magnitude of the synthetic data closely follows that of the real data. Only small variations can be seen for low magnitude gradients. The similarity of the image statistics of real and synthetic data suggests that trackers trained on the synthetic data could be successful on real TIR data.

## VI. EXPERIMENTAL RESULTS

### A. Datasets

We train several versions of our tracker using both real and generated TIR tracking data, summarized in Table II. As real TIR data we use BU-TIV [48], ASL [61], and OTCBVS [56]. The predominant class in these datasets is human/pedestrian,

Type	Dataset	Videos	Images	Bounding-boxes
Real	BU-TIV [48]	5	23,393	34,7291
	ASL [61]	13	6,490	7,773
	OTCBVS [56]	4	4861	19,944
	Total	22	34,744	375,008
Generated	VOT2016 [60]	60	21,455	21,455
	VOT2017 [6]	10	4,049	4,049
	OTB [11]	98	58,610	58,610
	Total	168	84,114	84,114

TABLE II: Datasets used for training the tracker, using real TIR data or generated TIR data from RGB images.

although BU-TIV [48] includes some vehicles and ASL [61] also contains animals like cat and horse. We select all those sequences that include annotated bounding boxes around the objects, leading to a total of 375K bounding boxes from 34K images. On the other hand, we generate synthetic TIR tracking data using the RGB videos from VOT2016 [60], VOT2017 [6], and OTB [11], which are standard tracking benchmarks used by the community. In total, we obtain 168 TIR videos with tracking annotations by translating the original RGB frames using pix2pix and transferring the corresponding bounding box annotations. The total number of bounding boxes is  $4.5\times$  greater than in the real TIR images. Furthermore, the generated TIR videos contain a wider variety of object classes than the real TIR videos. This increases the generality of the learned deep features. In both cases, we leave out around 10% of the videos during training as validation set.

We evaluate our TIR tracker on the VOT-TIR2017 dataset [6], which is identical to VOT-TIR2016 dataset [5] as the 2016 edition of this benchmark was far from being saturated. It contains 25 TIR videos of varying image resolution, with an average sequence length of 740 frames adding up to a total of 13,863 frames. Each sequence has been manually annotated with exactly one bounding box per frame around a particular object instance. There is a wide variety of object classes, including pedestrian, animals such as rhino or bird, and vehicles like quadcopter or car. Moreover, the dataset includes extra annotations in the form of attributes, either at frame level (e.g. camera motion, occlusion) or at the sequence level (e.g. blur, background clutter). This test dataset has no videos in common with the RGB modality of VOT2016-17 used for training.

### B. Evaluation measures and protocol

We follow the measures and evaluation protocol proposed by the VOT-TIR2017 benchmark [5]. The two primary measures are accuracy (A) and robustness (R), which have been shown to be highly interpretable and only weakly correlated [64]. Accuracy is computed as the overlap (intersection over union) between the predicted track region and the ground-truth bounding box, averaged over frames. The VOT protocol establishes that when the evaluated tracker fails, i.e. when the overlap is below a given threshold, it is re-initialized in the correct location five frames after the failure. In order to reduce the positive bias introduced by this protocol, the accuracy measure ignores the first ten frames after the re-initialization

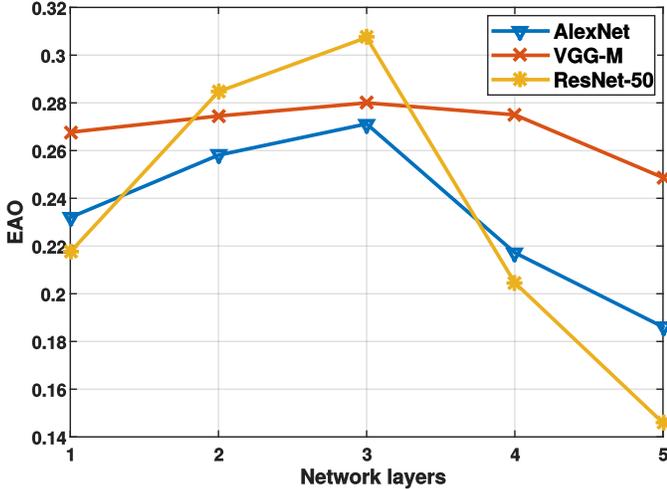


Fig. 6: The EAO on VOT-TIR2017 [6] when using deep features extracted from different layers.

when computing the average overlap. Robustness measures the number of times the tracker fails for each sequence and then takes the average over all sequences. These two measures are conflated into a third, the Expected Average Overlap (EAO), which is the main measure used to rank the trackers. The EAO estimates the expected average overlap of a tracker for a particular sequence of a fixed, short length. We refer the reader to [12] for more details.

Besides the standard VOT metrics, we also report results following the One-Pass Evaluation (OPE) protocol originally proposed in [11]. The most standard evaluation metric used with this protocol is success rate. For each frame in the test video, we compute the overlap between the predicted track and the ground-truth bounding box. A predicted track is considered successful if its overlap with the ground-truth is above a particular threshold. We obtain a success plot by evaluating the success rate at different overlap thresholds. Conventionally, the Area Under the Curve (AUC) of the success plot is reported as a summary measure. Note how this protocol does not reset the tracker in case of failure. We use the VOT toolkit [6] to compute the measure and plot the results.

### C. Implementation details

We train CFNet following [23]. We perform tests with three different networks as base model: AlexNet [19], VGG-M [65], and ResNet-50 [66]. As in [23], we reduce the total stride of the networks from 16 to 4 by changing the stride of the first and second pooling layers from 2 to 1 in AlexNet, and that of second convolutional and pooling layers in VGG-M. This allows us to obtain bigger feature maps, which benefits the correlation filters. For fairness, we apply this modification to all trained models. As training input data for the network, we randomly pick object regions from pairs of images from the same video. Specifically we crop a centered region on the object of approximately twice the object’s size, and resize it to  $125 \times 125$  pixels. We use Stochastic Gradient Descent (SGD) with momentum of 0.9 and weight decay of 0.0005 to fine-tune the network, which is pre-trained for image classification on

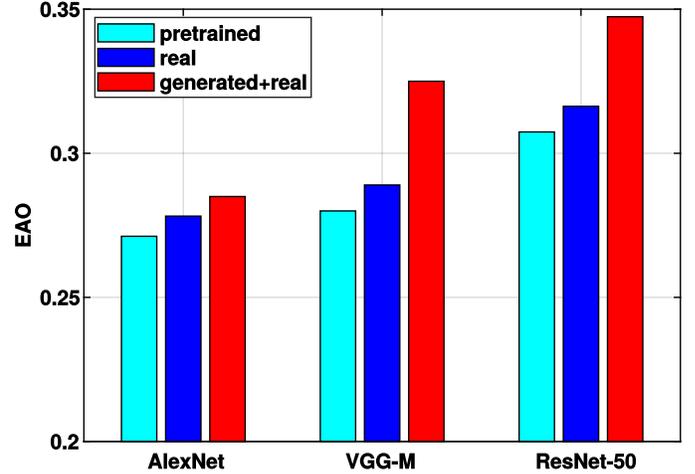


Fig. 7: The EAO on VOT-TIR2017 [6] when using deep features extracted from different networks. Our synthetic data can benefit general networks for fine-tuning.

ILSVCR12 [22]. The learning rate is decreased logarithmically at each epoch from  $10^{-4}$  to  $10^{-5}$ . The model is trained for 50 epochs with mini-batches of size 128.

For the baseline tracker ECO [10] (Fig. 3, blue dashed lines), we use the recommended settings (‘OTB\_DEEP\_settings’) detailed in the code provided by the authors [67]. ECO is an RGB tracker, so we have adapted the following parameters given the different nature of TIR data. Following [68], [69] we use the feature map of the third convolutional layer as the input of the correlation filter, (convolutional block in case of ResNet-50). We confirm these results in the following section. We reduce the learning rate used to update the correlation filter from the 0.009 used for RGB data to 0.003. A smaller learning rate is more suitable for TIR data, as TIR images have less detailed information than RGB, for example lacking texture, and thus the object appearance remains more stable during tracking. In order to optimally leverage the learned CNN features, we do not add the dimensionality reduction step at the output of each layer as in [10]. ECO uses this to increase the tracker’s efficiency, which is not a priority in our work.

Our trained models are available at [https://github.com/zhanglichao/generatedTIR\\_tracking](https://github.com/zhanglichao/generatedTIR_tracking).

### D. Network layers

Our tracker uses deep features from a particular network layer. Previous works [68], [69] selected mid-level features from the third convolutional layer as optimal for tracking in RGB videos. Here, we validate this choice for TIR data by analyzing the performance of the selected tracker across all layers for the three networks considered. We perform these experiments using only pre-trained features, i.e., we do not fine-tune the networks for tracking. Fig. 6 presents the tracking performance measured by EAO on VOT-TIR2017 [6] as a function of the network layer. Trackers that use features extracted from the third layer offer the best results, and thus we select this features for the remainder of the paper.

Tracker	without motion features			with motion features		
	EAO	A	R	EAO	A	R
handcrafted	0.235	0.60	2.74	0.361	0.62	1.12
pretrained	0.307	0.62	2.00	0.381	<b>0.69</b>	1.06
real	0.316	0.62	2.01	0.409	0.67	1.24
generated	0.321	<b>0.63</b>	2.00	0.419	0.65	0.83
generated $\rightarrow$ real	0.331	0.61	1.76	0.429	0.63	0.82
generated + real	<b>0.347</b>	<b>0.63</b>	<b>1.68</b>	<b>0.436</b>	0.65	<b>0.80</b>

TABLE III: Comparison of different tracker variants with and without motion features. Results are on the VOT-TIR2017 benchmark [6] with ResNet-50 [66] as base network. Boldface indicates the best results. In both cases, the best results are achieved when combining both real and generated TIR data.

### E. Network architectures

In this section, we experiment with all three base networks with different types of data used for training. All models use ECO [10] as base tracker, in some cases with the adaptations detailed in section IV. We consider two baselines, ‘pretrained’ and ‘real’. The first baseline uses features from the corresponding CNN pre-trained for the image classification task. On the other hand, ‘real’ is also fine-tuned using real TIR tracking datasets (sec. VI-A). Our tracker (‘generated + real’) combines both real TIR and synthesized from RGB with pix2pix for the fine-tuning process. Fig. 7 presents these results. For all base networks, fine-tuning helps when learning effective features for tracking. This shows that the generated data is complementary to the available real data, making the generated data beneficial even when a good amount of real data is available. Moreover, the gain granted by fine-tuning the network is significantly higher when augmenting the training dataset with our generated TIR data. The performance boost is especially remarkable for higher capacity models such as ResNet-50, since networks with more parameters require more data to train. For all following experiments, we use ResNet-50 as base network for the trackers.

### F. Results on real and generated data

We now present more detailed results for different configurations of our ECO tracker with ResNet-50. We include another baseline (‘handcrafted’) that employs hand-crafted features as it is prevalent in TIR tracking [24], [26], [4], and thus has not been trained using data. The variant called ‘generated’ is fine-tuned using only our generated TIR data with pix2pix. Finally, we consider another way of combining real and generated data to train the model, ‘generated  $\rightarrow$  real’, which besides uses a two-step fine-tuning mechanism, first using generated data and then real data. This is opposed to ‘generated + real’, which fine-tunes using both real and generated data simultaneously without distinction. Table III presents the results for all these models using metrics EAO, A, and R on the VOT-TIR2017 dataset [6].

First, we observe how the use of deep features is fundamental for the success of this tracker, given the low accuracy of the hand-crafted model. Simply using pre-trained features already provides a significant improvement in terms of EAO.

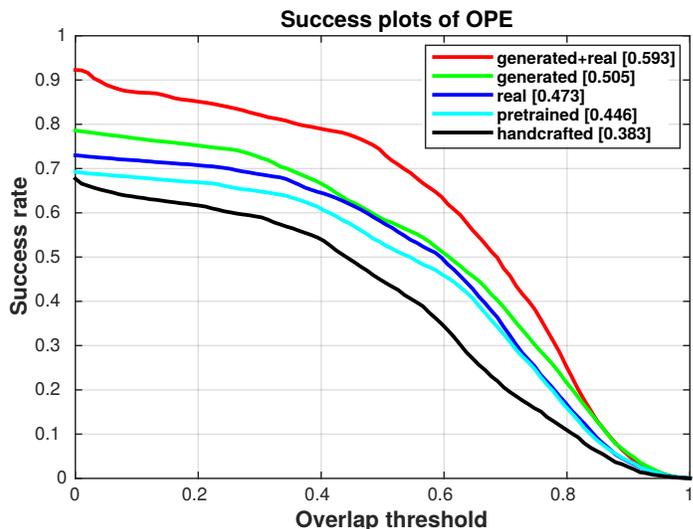


Fig. 8: The success plot of one-pass evaluation (OPE) on the the VOT-TIR2017 benchmark [6]. We show the AUC score of each tracker in the legend. The best results are obtained when using both real and generated data.

Fine-tuning the model on real data brings further benefits. Interestingly, fine-tuning only on the generated data with pix2pix results in better performance than fine-tuning on the real data; with an EAO of 0.289 on real data and 0.300 on generated data. This demonstrates our intuition that having great amounts of diverse data is very relevant when learning specialized deep features for TIR tracking. Finally, simultaneously using both real and generated data to fine-tune the network results in our best model. Moreover, training without distinguishing between the two types of data leads to better results, as opposed to a more complex two-stage fine-tuning process.

We present results using the OPE evaluation metric in Fig. 8. Also under this metric, hand-crafted features show a clearly inferior performance compared to deep features. Simple pre-trained deep features obtain higher success rates, especially for mid-range overlap thresholds. Fine-tuning on real data gives the tracker a small boost, and when fine-tuning using our generated data, the performance is further improved. Finally, the best performance is achieved when fine-tuning using both types of data simultaneously.

Finally, we analyze the performance of our generated + real tracker for different amounts of generated TIR data. Fig. 9 shows EAO as a function of the percentage of synthetic TIR data in the total training set. Interestingly, increasing the amount of synthetic data monotonically improves the tracker performance. The rightmost point, which corresponds to using all our generated data (90% of the training set), does not seem to be saturated, and thus additional generated data could bring an even further performance boost.

### G. Adding motion features

As detailed in [4], the use of hand-crafted motion features can substantially improve tracking performance for TIR data. Following the implementation of the SRDCFir tracker [4], we

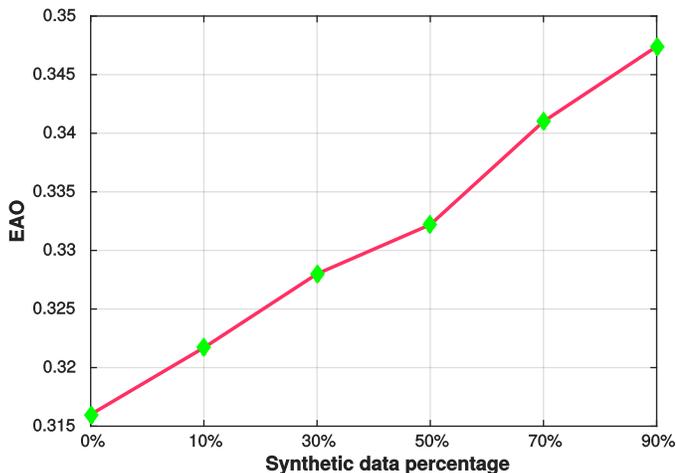


Fig. 9: Performance of our tracker (generated+real) on VOT-TIR2017 [6] for different percentages of synthetic data. The leftmost point indicates using only real data.

compute motion features by thresholding the absolute pixel-wise difference between the current and the previous frame. We then use this motion mask as an extra feature channel.

Table III presents the results of our trained models when motion features are used alongside deep features. We can see how motion features provide significant performance improvements to all models. Furthermore, the conclusions drawn in the previous experiment still hold. The models trained with generated data outperform both the pre-trained model and the model trained with real data only. Finally, the model trained with a combination of generated and real data achieves an impressive performance, surpassing other methods.

We show a qualitative comparison baseline ECO (pre-trained) and ours (generated) in Fig. 1. In challenging cases (e.g. third row), the improved deep features learned using our generated TIR data lead to a tracking model that is accurate while robust to various problematic conditions such as occlusion, scale change, and out-of-plane rotation.

#### H. State-of-the-art Comparison

Here, we compare our best model with the three top TIR trackers in the VOT-TIR2017 challenge [6], i.e. DSLT [24], EBT [26], and SRDCFir [4]. We also include in our comparison the best CNN-based tracker in VOT-TIR2016, TCNN [70]. Additionally, we compare with recently introduced CF-based (CSRDCF) [34] and spatial CF-based (CREST) [16] trackers. These trackers have shown excellent performance on VOT [12] and OTB [11] RGB datasets.

Table IV shows the comparison of our best model (generated+real) including motion mask with the state-of-the-art methods in literature on the VOT-TIR2017 benchmark [6]. Among the existing methods, SRDCFir and EBT achieve EAO scores of 0.364 and 0.368 respectively. An EAO score of 0.287 is achieved by the TCNN tracker. The recently introduced CREST and CSRDCF trackers achieve EAO scores of 0.215 and 0.248 respectively. The current state-of-the-art on this dataset is the DSLT tracker with an EAO score of 0.401.

Tracker	EAO	A	R
CREST	0.215	0.56	4.13
CSRDCF	0.248	0.57	3.49
TCNN	0.287	0.62	2.79
SRDCFir	0.364	0.63	1.10
EBT	0.368	0.44	0.82
DSLTL	0.401	0.60	0.91
Ours	<b>0.436</b>	<b>0.65</b>	<b>0.80</b>

TABLE IV: Comparison with state-of-the-art trackers on VOT-TIR2017 [6]. Boldface indicates the best results. The results are reported in terms of expected average overlap (EAO), robustness (failure rate) and accuracy. Our proposed tracker significantly outperforms the state-of-the-art by achieving an EAO score of 0.436.

Our tracker significantly outperforms DSLT by setting a new state-of-the-art with an EAO score of 0.448. Our approach also achieves superior performance in terms of accuracy and obtains second best results in terms of robustness. We further analyze the robustness of our tracker and found our approach to have promising improvements with respect to robustness in all videos except *trees2*, compared to EBT.

Fig. 10 shows a qualitative comparison of our tracker with state-of-the-art methods. Our tracker follows the target object more accurately and is robust to challenging conditions such as scale change and occlusion. Among existing methods, DSLT also provides improved tracking performance but struggles with accurate target localization. The proposed TIR-specialized deep features learned through abundant generated TIR data enable precise target localization, leading to superior tracking results. Finally, the last two rows of the figure show two example cases in which our tracker fails. In the first case, the object is rather tiny and lies on a cluttered background region, which increases the probability of confusing the tracked object with the background. In the second case, there is a considerable scale change combined with heavy occlusion, leading to a poor estimation of the object extent and the corresponding tracking failure.

#### I. TIR Data Attributes Analysis

In order to provide a more detailed analysis of the results, we present in Fig. 11 the per-attribute performance comparison of our tracker and several state-of-the-art methods. The attributes are: camera motion, dynamics change, motion change, occlusion, size change, and others. Each attribute plot indicates the expected overlap for every tracker as a function of the sequence length, computed on a particular subset of videos annotated with the corresponding data attribute. For most attributes, including the challenging scenarios of heavy camera motion, motion change, and occlusion, our proposed tracker outperforms state-of-the-art trackers. This consistent improvement on challenging attributes is likely due to specialized discriminative features, learned specifically for TIR tracking. In case of dynamics change, both TCNN and DSLT provide superior tracking performance. The TCNN tracker [70] can accurately match object proposals due to a tree structure

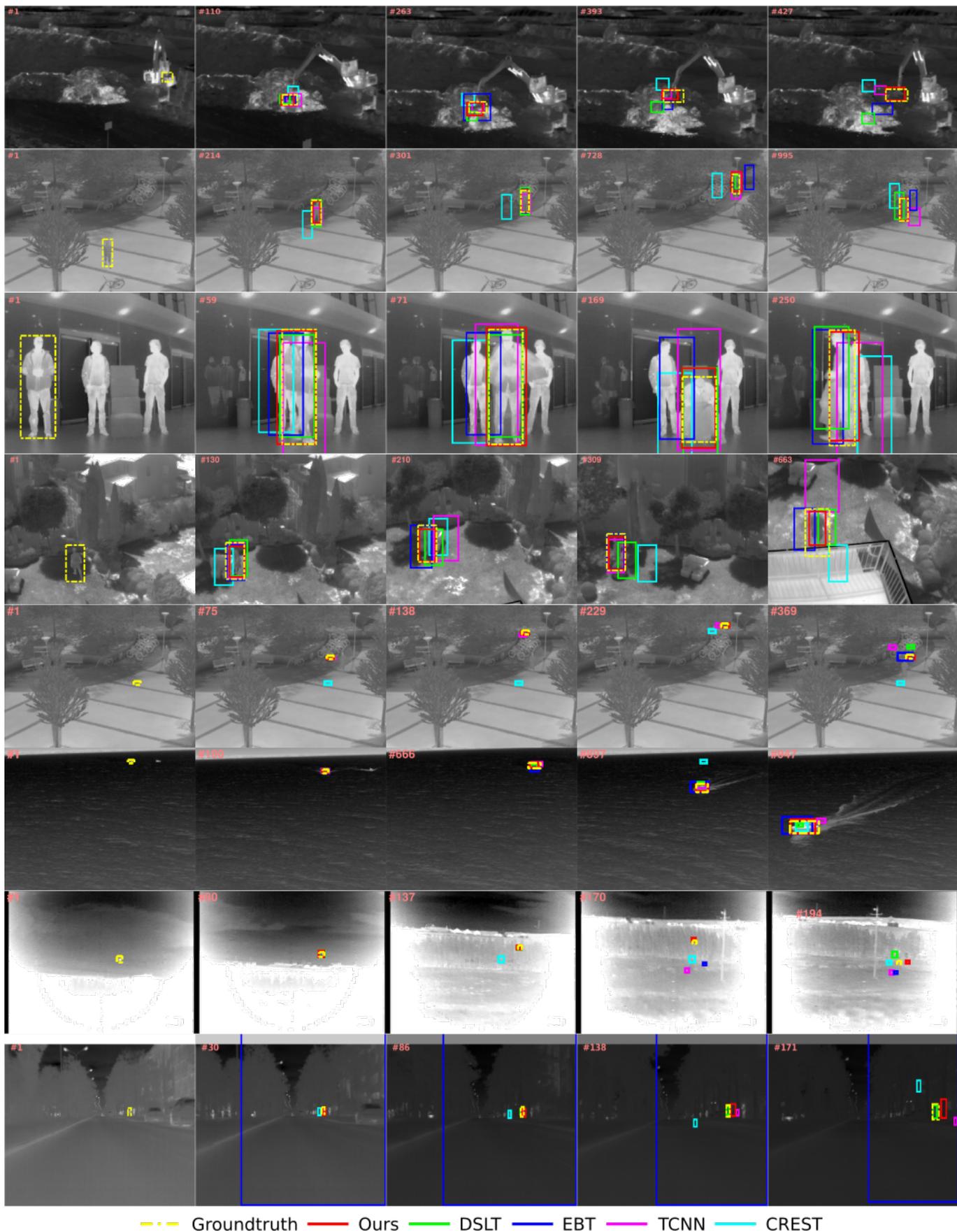


Fig. 10: Qualitative comparison of our approach trained on generated and real data with state-of-the-art trackers, CREST, TCNN, EBT and DSLT on the some challenging sequences, *excavator*, *jacket*, *mixed\_distractors*, *garden*, *quadrocopter2*, *boat2*, *bird* and *trees2* in VOT-TIR2017 [6]. Yellow dashed bounding box means Groundtruth and red solid bounding box is Ours. The last two rows show failure cases of our tracker.

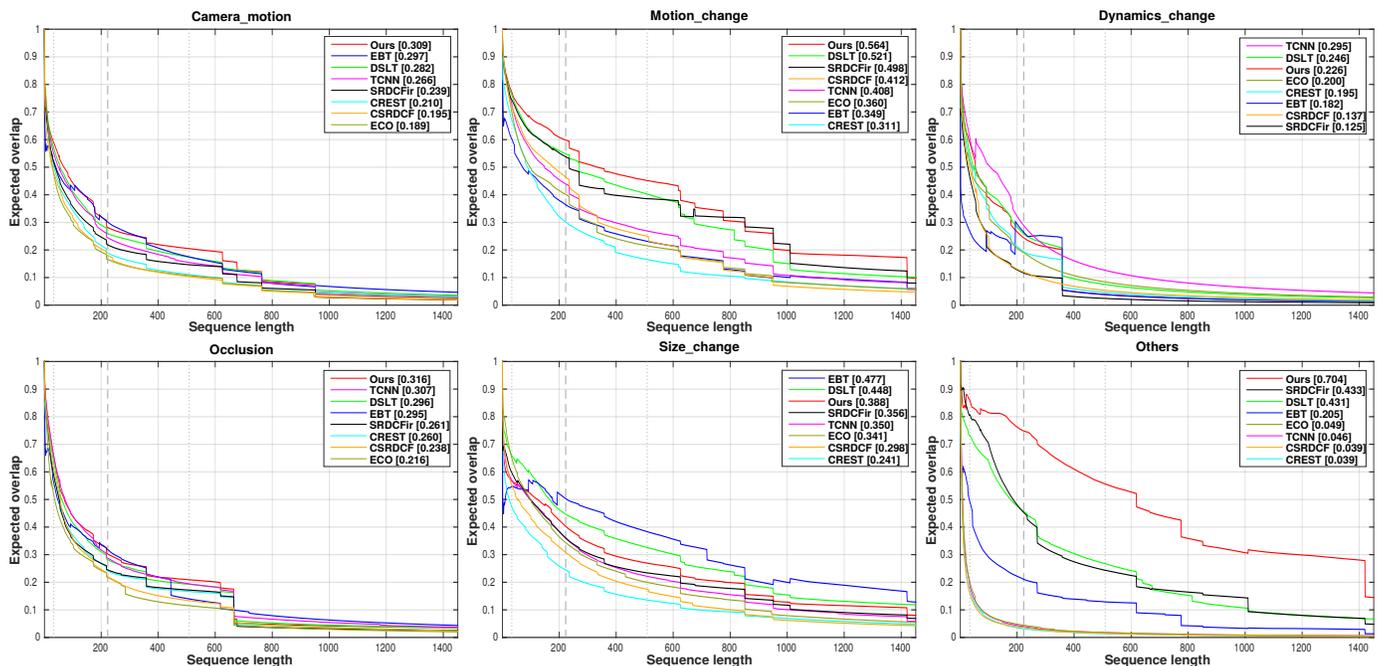


Fig. 11: Attribute-based comparison of our trackers with state of-the-art methods on VOT-TIR2017. We show expected overlap measure for four attributes: camera motion, dynamics change, motion change, occlusion, size change, and others. Our trackers provide consistent improvements in case of camera motion, motion change, occlusion and others, compared to existing methods.

encompassing multiple CNNs. The DSLT tracker [24] also uses dense proposals and structural learning classifier. In case of size change, the EBT tracker [26] and DSLT provide superior results. In this attribute, our approach provides the third best results by outperforming trackers such as SRDCFir and TCNN. Overall, our approach achieves best results on 4 out of 6 attributes.

## VII. CONCLUSION

In this paper, we have proposed a method to generate synthetic TIR data from RGB data. We use recent progress on image to image translation models for this purpose. The main advantage of this is that we can generate a large dataset of labeled TIR sequences. This dataset is far larger than datasets with real labeled sequences which are currently available for TIR tracking. These larger datasets allow us to perform end-to-end training for TIR features. To the best of our knowledge we are the first to train end-to-end features for TIR tracking. We show that our features trained on the synthetic data outperform other features for TIR tracking, including features which are computed by fine-tuning a network on real TIR sequences. In addition, we show that a combination of both real and generated data leads to a further improvement. Once we combine our feature with the motion feature we obtain state of the art results on the VOT-TIR2017.

## ACKNOWLEDGEMENTS

This work was supported by TIN2016-79717-R, and the CHISTERA project M2CR (PCIN-2015-251) of the Spanish Ministry and the ACCIO agency and CERCA Programme / Generalitat de Catalunya. This research was also supported

by CENIIT grant (18.14), and VR starting grant (2016-05543). We also acknowledge the generous GPU support from NVIDIA.

## REFERENCES

- [1] A. Emami, F. Dadgostar, A. Bigdeli, and B. C. Lovell, "Role of spatiotemporal oriented energy features for robust visual tracking in video surveillance," in *IEEE Ninth International Conference on Advanced Video and Signal-Based Surveillance*, 2012.
- [2] B. Renoust, D.-D. Le, and S. Satoh, "Visual analytics of political networks from face-tracking of news video," *IEEE Transactions on Multimedia*, vol. 18, no. 11, pp. 2184–2195, 2016.
- [3] L. Liu, J. Xing, H. Ai, and X. Ruan, "Hand posture recognition using finger geometric feature," in *Proceedings of the International Conference on Pattern Recognition*, 2012.
- [4] M. Felsberg, A. Berg, G. Hager, J. Ahlberg, M. Kristan, J. Matas, A. Leonardis, L. Cehovin, G. Fernandez, T. Vojir *et al.*, "The thermal infrared visual object tracking vot-tir2015 challenge results," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2015.
- [5] M. Felsberg, M. Kristan, J. Matas, A. Leonardis, R. Pflugfelder, G. Häger, A. Berg, A. Eldesokey, J. Ahlberg, L. Čehovin *et al.*, "The thermal infrared visual object tracking vot-tir2016 challenge results," in *Proceedings of the European Conference on Computer Vision*, 2016.
- [6] M. Kristan, A. Leonardis, J. Matas, M. Felsberg, R. Pflugfelder, L. Cehovin Zajc, T. Vojir, G. Hager, A. Lukežic, A. Eldesokey, and G. Fernandez, "The visual object tracking vot2017 challenge results," in *Proceedings of the International Conference on Computer Vision Workshops*, Oct 2017.
- [7] R. Gade and T. B. Moeslund, "Thermal cameras and applications: a survey," *Machine vision and applications*, vol. 25, no. 1, pp. 245–262, 2014.
- [8] D. S. Bolme, J. R. Beveridge, B. A. Draper, and Y. M. Lui, "Visual object tracking using adaptive correlation filters," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2010.
- [9] J. F. Henriques, R. Caseiro, P. Martins, and J. Batista, "High-speed tracking with kernelized correlation filters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 3, pp. 583–596, 2015.

- [10] M. Danelljan, G. Bhat, F. S. Khan, and M. Felsberg, "Eco: efficient convolution operators for tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.
- [11] Y. Wu, J. Lim, and M.-H. Yang, "Object tracking benchmark," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, no. 9, pp. 1834–1848, 2015.
- [12] M. Kristan, J. Matas, A. Leonardis, T. Vojir, R. Pflugfelder, G. Fernandez, G. Nebehay, F. Porikli, and L. Čehovin, "A novel performance evaluation methodology for single-target trackers," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 11, pp. 2137–2155, Nov 2016.
- [13] M. Mueller, N. Smith, and B. Ghanem, "A benchmark and simulator for uav tracking," in *Proceedings of the European Conference on Computer Vision*, 2016.
- [14] M. Danelljan, F. S. Khan, M. Felsberg, and J. van de Weijer, "Adaptive color attributes for real-time visual tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014.
- [15] M. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg, "Adaptive decontamination of the training set: A unified formulation for discriminative visual tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.
- [16] Y. Song, C. Ma, L. Gong, J. Zhang, R. W. Lau, and M.-H. Yang, "Crest: Convolutional residual learning for visual tracking," in *Proceedings of the International Conference on Computer Vision*, 2017.
- [17] M. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg, "Learning spatially regularized correlation filters for visual tracking," in *Proceedings of the International Conference on Computer Vision*, 2015.
- [18] M. Danelljan, G. Häger, F. Khan, and M. Felsberg, "Accurate scale estimation for robust visual tracking," in *Proceedings of the British Machine Vision Conference*, 2014.
- [19] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, 2012.
- [20] C. Ma, J.-B. Huang, X. Yang, and M.-H. Yang, "Hierarchical convolutional features for visual tracking," in *Proceedings of the International Conference on Computer Vision*, 2015.
- [21] M. Danelljan, A. Robinson, F. S. Khan, and M. Felsberg, "Beyond correlation filters: Learning continuous convolution operators for visual tracking," in *Proceedings of the European Conference on Computer Vision*, 2016.
- [22] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein *et al.*, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.
- [23] J. Valmadre, L. Bertinetto, J. Henriques, A. Vedaldi, and P. H. Torr, "End-to-end representation learning for correlation filter based tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.
- [24] X. Yu, Q. Yu, Y. Shang, and H. Zhang, "Dense structural learning for infrared object tracking at 200+ frames per second," *Pattern Recognition Letters*, vol. 100, pp. 152–159, 2017.
- [25] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2005.
- [26] G. Zhu, F. Porikli, and H. Li, "Beyond local search: Tracking objects everywhere with instance-specific proposals," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.
- [27] J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems*, 2014.
- [28] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.
- [29] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the International Conference on Computer Vision*, 2017.
- [30] H. K. Galoogahi, T. Sim, and S. Lucey, "Multi-channel correlation filters," in *Proceedings of the International Conference on Computer Vision*, 2013.
- [31] J. Van De Weijer, C. Schmid, J. Verbeek, and D. Larlus, "Learning color names for real-world applications," *IEEE Transactions on Image Processing*, vol. 18, no. 7, pp. 1512–1523, 2009.
- [32] M. Mueller, N. Smith, and B. Ghanem, "Context-aware correlation filter tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.
- [33] C. Ma, X. Yang, C. Zhang, and M.-H. Yang, "Long-term correlation tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015.
- [34] A. Lukezic, T. Vojir, L. C. Zajc, J. Matas, and M. Kristan, "Discriminative correlation filter with channel and spatial reliability," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.
- [35] H. Kiani Galoogahi, A. Fagg, and S. Lucey, "Learning background-aware correlation filters for visual tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.
- [36] M. Danelljan, G. Hager, F. Shahbaz Khan, and M. Felsberg, "Convolutional features for correlation filter based visual tracking," in *Proceedings of the IEEE International Conference on Computer Vision Workshops*, 2015.
- [37] X. Yu and Q. Yu, "Online structural learning with dense samples and a weighting kernel," *Pattern Recognition Letters*, vol. 105, pp. 59–66, 2018.
- [38] C. Li, H. Cheng, S. Hu, X. Liu, J. Tang, and L. Lin, "Learning collaborative sparse representation for grayscale-thermal tracking," *IEEE Transactions on Image Processing*, vol. 25, no. 12, pp. 5743–5756, 2016.
- [39] C. Li, X. Sun, X. Wang, L. Zhang, and J. Tang, "Grayscale-thermal object tracking via multitask laplacian sparse representation," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 47, no. 4, pp. 673–681, 2017.
- [40] J. Hoffman, S. Gupta, and T. Darrell, "Learning with side information through modality hallucination," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.
- [41] D. Xu, W. Ouyang, E. Ricci, X. Wang, and N. Sebe, "Learning cross-modal deep representations for robust pedestrian detection," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.
- [42] E. L. Denton, S. Chintala, R. Fergus *et al.*, "Deep generative image models using a laplacian pyramid of adversarial networks," in *Advances in Neural Information Processing Systems*, 2015.
- [43] G. Perarnau, J. van de Weijer, B. Raducanu, and J. M. Álvarez, "Invertible conditional gans for image editing," in *Advances in Neural Information Processing Systems 2016 Workshop on Adversarial Training*, 2016.
- [44] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," in *Advances in Neural Information Processing Systems*, 2016.
- [45] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014.
- [46] R. Zhang, P. Isola, and A. A. Efros, "Colorful image colorization," in *Proceedings of the European Conference on Computer Vision*, 2016.
- [47] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*, 2015.
- [48] Z. Wu, N. Fuller, D. Theriault, and M. Betke, "A thermal infrared video benchmark for visual analysis," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2014.
- [49] X. Song, L. Herranz, and S. Jiang, "Depth cnns for rgb-d scene recognition: Learning from scratch better than transferring from rgb-cnns," in *Proceedings of the Conference on Artificial Intelligence*, 2017.
- [50] S. Hwang, J. Park, N. Kim, Y. Choi, and I. S. Kweon, "Multispectral pedestrian detection: Benchmark dataset and baseline," *Integrated Comput.-Aided Eng.*, vol. 20, pp. 347–360, 2013.
- [51] Q. Wang, J. Gao, J. Xing, M. Zhang, and W. Hu, "Dcfnet: Discriminant correlation filters network for visual tracking," *arXiv preprint arXiv:1704.04057*, 2017.
- [52] E. Gundogdu and A. A. Alatan, "Good features to correlate for visual tracking," *arXiv preprint arXiv:1704.06326*, 2017.
- [53] J. Nocedal and S. J. Wright, "Numerical optimization 2nd," 2006.
- [54] C. Li and M. Wand, "Precomputed real-time texture synthesis with markovian generative adversarial networks," in *Proceedings of the European Conference on Computer Vision*, 2016.
- [55] A. González, Z. Fang, Y. Socarras, J. Serrat, D. Vázquez, J. Xu, and A. M. López, "Pedestrian detection at day/night time with visible and fir cameras: A comparison," *Sensors*, vol. 16, no. 6, p. 820, 2016.
- [56] J. W. Davis and M. A. Keck, "A two-stage template approach to person detection in thermal imagery," in *Application of Computer Vision, 2005. WACV/MOTIONS'05 Volume 1. Seventh IEEE Workshops on*, 2005.
- [57] C. Palmero, A. Clapés, C. Bahnsen, A. Møgelmoose, T. B. Moeslund, and S. Escalera, "Multi-modal rgb–depth–thermal human body segmentation," *International Journal of Computer Vision*, vol. 118, no. 2, pp. 217–239, 2016.

- [58] G.-A. Bilodeau, A. Torabi, P.-L. St-Charles, and D. Riahi, "Thermal-visible registration of human silhouettes: A similarity measure performance evaluation," *Infrared Physics & Technology*, vol. 64, pp. 79–86, 2014.
- [59] A. Torabi, G. Massé, and G.-A. Bilodeau, "An iterative integrated framework for thermal-visible image registration, sensor fusion, and people tracking for video surveillance applications," *Computer Vision and Image Understanding*, vol. 116, no. 2, pp. 210–221, 2012.
- [60] M. Kristan, R. Pflugfelder, A. Leonardis, J. Matas, F. Porikli, L. Cehovin, G. Nebel, G. Fernandez, T. Vojir, A. Gatt *et al.*, "The visual object tracking vot2016 challenge results," in *Proceedings of the European Conference on Computer Vision Workshops*, 2016.
- [61] J. Portmann, S. Lynen, M. Chli, and R. Siegwart, "People detection and tracking from aerial thermal views," in *Proceedings of IEEE International Conference on Robotics and Automation*, 2014.
- [62] R. Gade, A. Jørgensen, and T. B. Moeslund, "Long-term occupancy analysis using graph-based optimisation in thermal imagery," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013.
- [63] C. Gao, Y. Du, J. Liu, J. Lv, L. Yang, D. Meng, and A. G. Hauptmann, "Infar dataset: Infrared action recognition at different times," *Neurocomputing*, vol. 212, pp. 36–47, 2016.
- [64] L. Čehovin, A. Leonardis, and M. Kristan, "Visual object tracking performance measures revisited," *IEEE Transactions on Image Processing*, vol. 25, no. 3, pp. 1261–1274, 2016.
- [65] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: Delving deep into convolutional nets," *arXiv preprint arXiv:1405.3531*, 2014.
- [66] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.
- [67] <https://github.com/martin-danelljan/ECO>.
- [68] H. Nam and B. Han, "Learning multi-domain convolutional neural networks for visual tracking," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016.
- [69] E. Park and A. C. Berg, "Meta-tracker: Fast and robust online adaptation for visual object trackers," *arXiv preprint arXiv:1801.03049*, 2018.
- [70] H. Nam, M. Baek, and B. Han, "Modeling and propagating cnns in a tree structure for visual tracking," *arXiv preprint arXiv:1608.07242*, 2016.