ED-DCFNet: an unsupervised encoder-decoder neural model for event-driven feature extraction and object tracking

Raz Ramon Hadar Cohen-Duwek Elishai Ezra Tsur

Neuro-Biomorphic Engineering Lab (NBEL)
Department of Mathematics and Computer Science, The Open University of Israel

elishai@nbel-lab.com

Abstract

Neuromorphic cameras feature asynchronous event-based pixel-level processing and are particularly useful for object tracking in dynamic environments. Current approaches for feature extraction and optical flow with high-performing hybrid RGB-events vision systems require large computational models and supervised learning, which impose challenges for embedded vision and require annotated datasets. In this work, we propose ED-DCFNet, a small and efficient ($< 72k$) unsupervised multi-domain learning framework, which extracts events-frames shared features without requiring annotations, with comparable performance. Furthermore, we introduce an open-sourced event and frame-based dataset that captures indoor scenes with various lighting and motion-type conditions in realistic scenarios, which can be used for model building and evaluation. The dataset is available at https://github.com/NBELab/UnsupervisedTracking.

1. Introduction

Neuromorphic (brain-inspired) event-driven cameras communicate transients in luminance as events. They are characterized by high temporal resolution, high dynamic range, sparse data representation, and low power consumption of up to 10 μW [1], and are therefore well suited to embedded vision. Their asynchronous event-based pixel-level processing allows for fast movement detection in dynamic environments with applications ranging from robotics to star tracking [2, 7, 8, 18].

Feature detection and object tracking with an event camera that generates an event stream are key for a wide range of vision tasks [8]. Some approaches adapted conventional frame-based feature extraction and tracking through the integration of global and local object detection and online learning [13, 14]. Others used a parametric model to detect and track moving objects that did not conform to the camera's motion compensation model [10, 12, 27]. Contemporary methodologies leveraged deep learning for feature extraction, with architectures ranging from convolutional neural networks (CNNs) [5, 6, 21] to transformers [23].

Several studies exploited the complementary features of frames and events to improve frame-based vision tracking by integrating event data and frames [3]. For example, [25] utilized a spiking neural network (SNN) to extract event-driven features and a deep CNN to extract RGB-based features to capture complementary visual constructs. In a follow-up work, [24] used those cross-domain visual constructs to enhance object tracking performance in challenging conditions. In a later study [21], a cross-modality transformer was developed to further facilitate event-frames feature fusion. Those hybrid dual-modality frameworks leverage the strengths of frame-based vision, which excels at capturing texture details and slow-motion sequences, and event-based cameras, which are less sensitive to motion blur and have a wider dynamic range thus leading to a more reliable object tracking.

The main disadvantages of these frameworks are the underlying model size and the required computational resources, making them unsuitable for embedded systems. Moreover, most of these methods necessitate supervised learning, and thus require annotated datasets.

In [20], DCFNet was introduced, demonstrating simultaneous learning of convolutional features and feature correlation for object tracking using a lightweight neural network. In a follow-up study, [19] extended DCFNet to support unsupervised training in a framework termed unsupervised deep tracking (UDT). UDT utilizes bidirectional tracking in which the model learns to localize and backtrack a target object in successive frames using a siamese correlation filter network (two identical networks that share the

same features). UDT demonstrated comparable real-time tracking performance. This unsupervised learning approach frames DCFNet into a Siamese framework in which a series of shared-weight branches are used to extract feature representations. These representations are utilized to train correlation filters for each feature, which are subsequently convolved with the succeeding feature in the sequence to produce the result. A key benefit of the Siamese DCF network lies in its ability to incorporate both the feature extraction CNN and the correlation filter within an end-to-end framework.

In this work, we extended DCFNet [20] and UDT [19] for both event-based and hybrid tracking. We used unsupervised multi-domain learning to extract shared features from event and frame data without requiring annotations. This allows us to leverage a single online (real-time) tracker capable of operating in both domains: frames and events, as well as in a hybrid setting. Our real-time tracker seamlessly integrates event features, frame features, and shared features to enhance tracking performance while using a lightweight neural model. Furthermore, we introduce an open-sourced event and frame-based dataset that captures realistic indoor scenes in various conditions, such as low light, bright light, different motion types, and blur levels. Our dataset contains various sequence lengths for testing long- and short-term tracking.

2. Methods

2.1. Input representation

We adapted DCFNet, originally crafted for frame-based input, by converting event data into a voxel grid representation [15, 16, 26]. We modified DCFNet-tracking to apply to both grayscale frames and events by incorporating event voxels along with their corresponding grayscale frames. Considering N events between two successive frames, each event e_i was represented by four values: $e_i = \{x_i, y_i, t_i, p_i\}_{i=1..,N}$, where x_i, y_i are the location of the event in the sensor, t_i is the event's timestamp (in milliseconds) and p_i is its polarity (a binary output correlated to the increase or decrease in the intensity of the pixels). We define t_i^* as the normalized timestamp of the i^{th} event by $t_i^* = \frac{t_i - t_{min}}{t_{max} - t_{min}}(B-1)$. An Event Voxel V has the dimensions of $H \times W \times B$ and is defined using temporal interpolation using:

$$V_{t_n} = \sum_{i=0}^{N} p_i max(0, 1 - |t_n - t_i^*|) \quad (1)$$

where H, W are the sensor's height and width, n is the temporal bin index, and B is the number of temporal bins.

2.2. Network architecture

The proposed tracking architecture (Figure 1) comprises the following stages: 1. A lightweight encoder-decoder network processes a template input (an object to track, which can be in frames, events, or in a hybrid modality) and a searched input (the subsequent voxel or frame for object tracking) to extract a shared feature space; 2. The template feature vector is converted to the Fourier domain to produce a discriminative correlation filter (DCF); 3. The correlation response between the DCF of the template and the searched feature vector is calculated to derive a correlation response map. In the correlation response map, the peak response signifies the template's location relative to the searched input. During unsupervised training, both forward and backward tracking takes place (Figure 2). During forward tracking, the correlation between the template and 'search1' (the subsequent input in the sequence of frames or voxels) is calculated. Subsequently, forward tracking is performed between the objects in 'search1' and 'search2' (the next frame or voxel in the sequence). Finally, backward tracking is conducted between 'search2' and the template. Ideally, the correlation between 'search2' and the template yields a peak at the center of the correlation response map (the original object location). The network is trained using consistency loss to minimize the distance between forward and backward tracking.

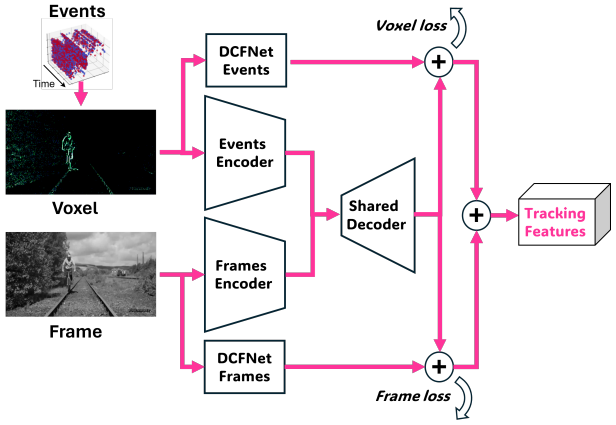

Figure 1. ED-DCFNet network architecture.

To extract shared features from both event voxels and grayscale frames, we used an Encoder-Decoder neural network, where the event voxels have one encoder, the grayscale frames have another encoder, and they share the same decoder (Figure 1). While the first layer's size of the event encoder is $B \times 3 \times 32 \times 32$, the size of the corresponding layer in the grayscale frames is $1 \times 3 \times 32 \times 32$ (the number of color channels). In both encoders, the second convolution layer $32 \times 3 \times 32 \times 64$ is followed by a max pooling layer. The decoder consists of two $2 \times 2 \times 64 \times 32$

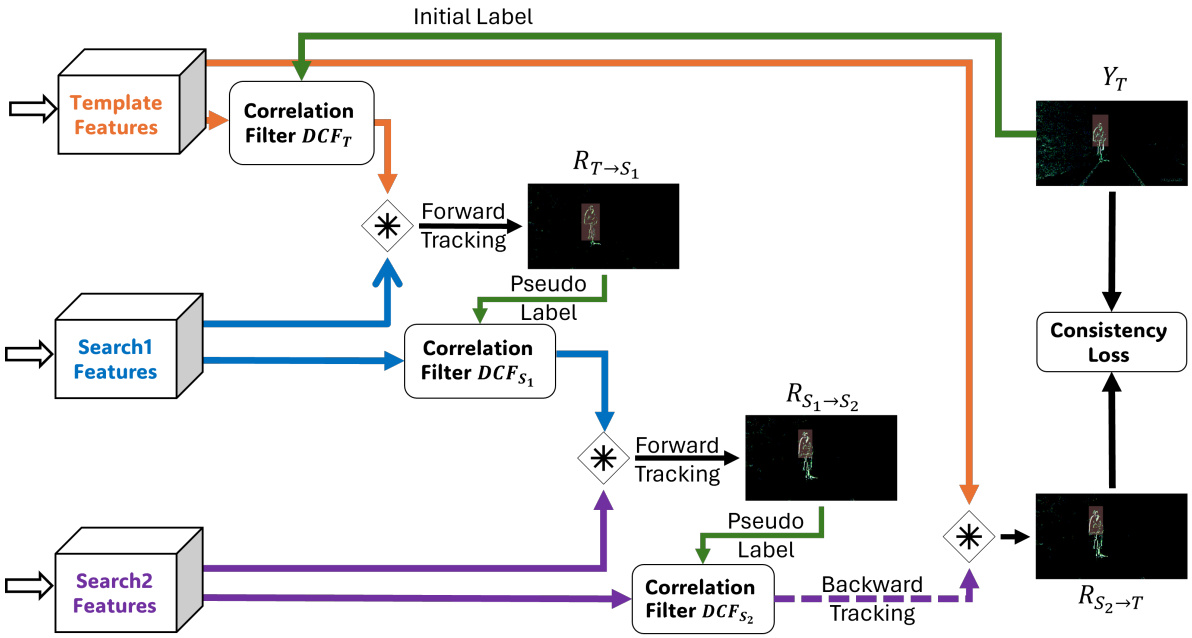

Figure 2. ED-DCFNet unsupervised learning architecture.

and $2 \times 2 \times 32 \times 32$ convolution layers. In total, the network features only 70K parameters.

DCFNet [20] is a shallow CNN that contains two $3 \times 3 \times 3 \times 32$ and $3 \times 3 \times 32 \times 32$ convolution layers. We incorporated the original DCFNet network with our Encoder-Decoder network, as shown in Figure 1. This architecture, we termed ED-DCFNet (Encoder-Decoder DCFNet) uses two DCFNet networks as skip connections. It amplifies the extracted features by merging the features derived from each input domain via the DCFNet networks, with the encoder-decoder network-derived shared features.

2.3. Unsupervised deep tracking

Annotating event-camera data, which is required for supervised learning, presents a prominent challenge for the design of neural trackers. Here, we extended DCFNet [20] and UDT [19] to provide an unsupervised neural model. To train the network we used sequences of event voxels along with their corresponding grayscale frames. Each sequence S contained both a sequence of event voxels V and a sequence of corresponding grayscale frames G, where

$$S = [V_T, V_{S_1}, V_{S_2}], [G_T, G_{S_1}, G_{S_2}] \tag{2}$$

In this context, T denotes the selected template input (the tracked object), while S_1 and S_2 donate the selected search inputs (search1 and search2), the subsequent frames or voxels where the object will be searched. We introduced each input into its corresponding encoder, to derive an encoder vector:

$$E_V : \forall V : E_V = Encoder_{Events}(V) \tag{3}$$

$$E_G : \forall G : E_G = Encoder_{Grayscale}(G) \tag{4}$$

The encoder vectors (E_V and E_G) were introduced into the Decoder to extract shared feature vectors:

$$F_{V_{Dec}} = Deccoder(E_V) = Decoder(E_G) = F_{G_{Dec}} \tag{5}$$

Each input was also introduced to a DCFNet network to derive feature vectors:

$$\forall V : F_V = F_{V_{Dec}} + F_{V_{DCF}} \tag{6}$$

$$\forall G : F_G = F_{G_{Dec}} + F_{G_{DCF}} \tag{7}$$

Finally, the derived features were combined into a single feature:

$$F = F_G + F_V \tag{8}$$

creating the merged features sequence $[F_T, F_{S_1}, F_{S_2}]$.

A DCF was calculated using the template feature vector within the Fourier domain, as follows:

$$DCF(X, Y) = \mathscr{F}^{-1}\left(\frac{\mathscr{F}(X) \odot \mathscr{F}^{\star}(Y)}{\mathscr{F}^{\star}(X) \odot \mathscr{F}(X) + \lambda}\right) \tag{9}$$

where X is the feature vector to correlate with, Y is the label, $\odot$ is the element-wise product, $\mathscr{F}(\cdot)$ is the Discrete

Fourier Transform (DFT), $\mathscr{F}^{-1}(\cdot)$ is the inverse DFT, $\star$ denotes complex-conjugate operation, and λ is a regularization parameter. We used (9) to calculate $DCF_T = DCF(F_T, Y_T)$, where Y_T is a pseudo-sharp Gaussian label centered at the middle of the input, frame or voxel. The correlation response between the DCF of the template and the S_1 feature vector ($R_{T\to S_1}$) was then calculated. The discriminative correlation response [4] was calculated using:

$$R(X,Y) = X * Y = \mathscr{F}^{-1}(\mathscr{F}^{\star}(X) \odot \mathscr{F}(Y)) \quad (10)$$

where $*$ denotes a circular convolution. We used (10) to calculate $R_{T\to S_1} = R(DCF_T, F_{S_1})$. Using Equations (9), (10) we calculated $DCF_{S_1} = DCF(F_{S_1}, R_{T\to S_1})$ and the discriminative correlation response between S_1 and S_2:$R_{S_1\to S_2} = R(DCF_{S_1}, F_{S_2})$.We termed $R_{T\to S_1}$ and $R_{S_1\to S_2}$ as forward tracking.

During backward tracking, we calculated the discriminative correlation response opposite to the sequence order, using Equations (9), (10): $DCF_{S_2} = DCF(F_{S_2}, R_{s_1\to S_2})$ and $R_{S_2\to T} = R(DCF_{S_2}, F_T)$. The backward response $R_{S_2\to T}$ is expected to be a sharp Gaussian profile, aligning with the expectation that the tracker returns to its original location in the center of the voxel. To achieve this, the model was trained to minimize the consistency loss, as described below.

2.4. Consistency loss

With the expectation that the backward tracking will backtrack the tracker to its original location, the consistency loss, $\mathscr{L}_C$, was calculated using:

$$\mathscr{L}_C = ||R_{S_2\to T} - Y_T||_2^2 \quad (11)$$

This loss enables unsupervised feature extraction. Since the loss was computed using the discriminative correlation responses rather than with the actual inputs, we could train the model and use it to track objects within multi-domain input, as well as within only grayscale frames or event voxels. Multi-domain training allows the definition of weighted average loss, one for each domain, while using the $R_{S_1\to S_2}$ and the separated feature vectors, F_G, F_V to perform backward tracking, as follows:

$$\begin{aligned} DCF_{G[S_2]} &= DCF(F_{G[S_2]}, R_{S_1\to S_2}) \\ DCF_{V[S_2]} &= DCF(F_{V[S_2]}, R_{S_1\to S_2}) \end{aligned} \quad (12)$$

$$\begin{aligned} R_{G[S_2]\to G[T]} &= R(DCF_{G[S_2]}, F_{G[T]}) \\ R_{V[S_2]\to V[T]} &= R(DCF_{V[S_2]}, F_{V[T]}) \end{aligned} \quad (13)$$

And finally:

$$\begin{aligned} \mathscr{L}_C &= \alpha||R_{S_2\to T} - Y_T||_2^2 \\ &+\beta||R_{G[S_2]\to G[T]} - Y_T||_2^2 \\ &+\gamma||R_{V[S_2]\to V[T]} - Y_T||_2^2 \end{aligned} \quad (14)$$

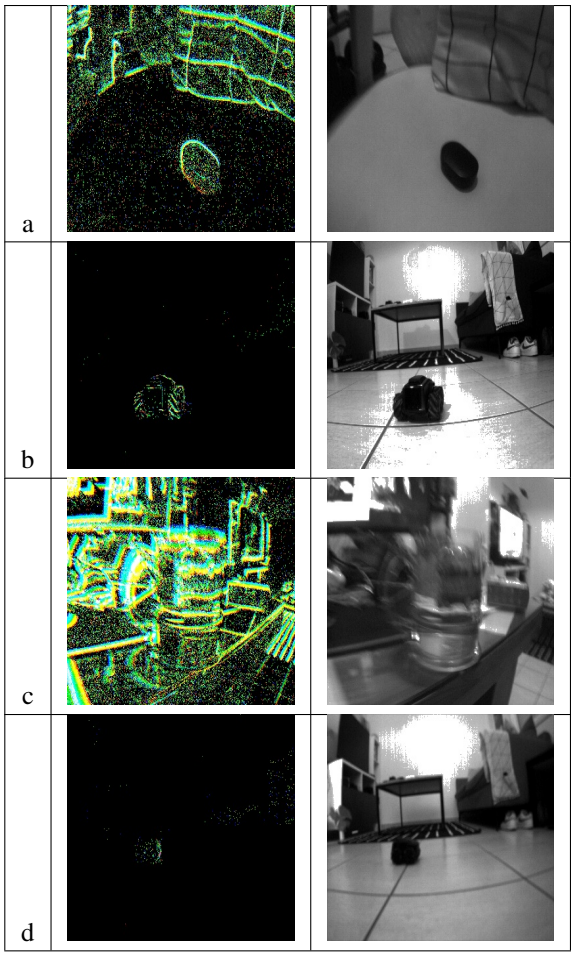

Figure 3. Examples of the real-event-camera-indoor dataset; a) Regular light, static object, and a moving camera; b) Moving object with a static camera; c) Overexposed imagery with motion blur; d) Underexposed imagery.)

where here we chose $\alpha = 0.5, \beta = 0.3, \gamma = 0.2$.

2.5. Training data

We used ESIM [11] to synthetically generate events from frame-based video, where ESIM provides both events and grayscale frames for the same video. The process involves creating a virtual 3D scene and simulating the movement of a camera within it to generate a stream of events, intensity frames, and depth maps. Thousands of rendered images were generated along the specified trajectory, ensuring minimal motion between consecutive images. Each pixel retained the timestamp of the last event triggered at that location, allowing for time interpolation of image intensities and detecting brightness changes between images. This method effectively provides continuous timestamps, simulating asynchronous event generation. Like [16], we used objects from the COCO [9] dataset to simulate the move-

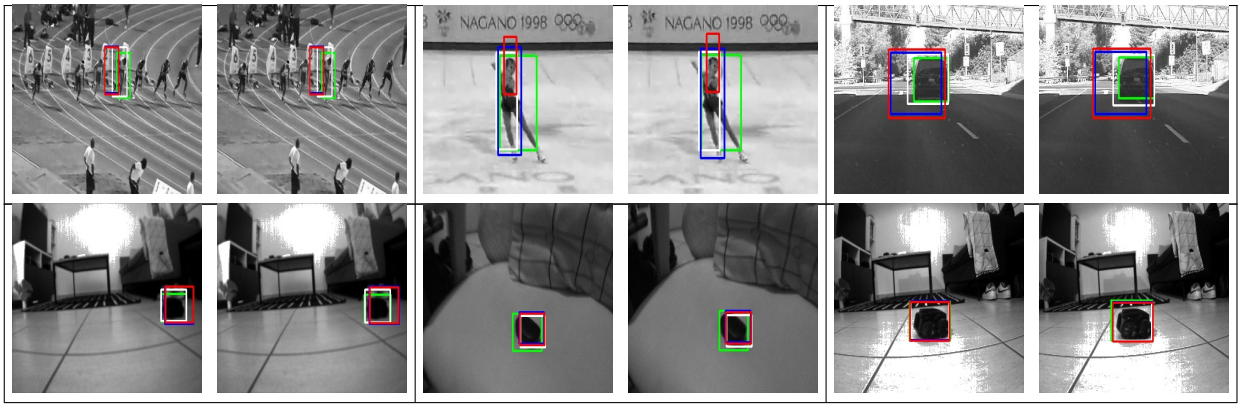

Figure 4. Selected results from the grayscale frames datasets, where the first row showcases scenes from OTB2015 and the second row showcases scenes from our real-event-camera-indoor dataset

ment of trackable objects, with the objects being randomly selected from the dataset. In the generated videos (111 videos, each 10 seconds in length), the objects' motion was restricted to affine transformations such as translations, rotations, and dilations at different velocities. During training, a wide range of scene dynamics was introduced to the network, which promoted the network's generalization from static images to arbitrary camera motions. During the training process, sequence augmentation (described below) was used to increase the veracity of the data.

2.6. Sequence augmentations

We employed event-voxel augmentations (AUG) during training to create a variety of possible movements, and apply these augmentations for both the voxel and the grayscale frames sequences. The augmentations applied to the dataset included: 1. Reversing sequence, which reversed the order of the voxels sequence; 2. Opposite polarities sequence, where each event had the opposite polarity; 3. Random order sequence, which randomly chooses the order of the search voxels in the sequence; and 4. Fastback sequence, where the last search voxel or frame was replaced by the template voxel or frame.

2.7. Dataset acquisition

Several real event-tracking datasets have been introduced in the literature [5, 10, 24]. However, a portion of these datasets remain unavailable to the public. Moreover, these datasets often consist of merely a few frames per video and contain scenes that are not relevant to typical camera usage scenarios. Therefore, we captured and built a new event dataset using the DAVIS346 [17], an event camera with a spatial resolution of $260X346$. Our dataset contains 19 different sequences captured in two main scenarios: a moving object with a static camera and a moving camera with a static object, in various conditions, such as daylight, low light, fast motion, motion blur, over and under exposure, and reflective scenery and transparent objects. We manually annotated the dataset. The dataset sequence lengths vary when the shortest contained 8 frames and the longest contained 240 frames. Figure 3 presents examples from the dataset and it is available in project's repository.

2.8. Real time object tracking

A trained network can be deployed as a real time (online) tracker. In the frame-based implementation of the online learning tracker, a DCF (Equation 9) between two consecutive frames can be updated as follows:

$$Net_t = (1 - \alpha_t)Net_{t-1} + \alpha_t Net \tag{15}$$

where $\alpha_t \in [0, 1]$ is the linear interpolation coefficient, and:

$$\begin{aligned} Net &= DCFNet_{Events} + Enc - Dec_{Events} \\ Net &= DCFNet_{Frames} + Enc - Dec_{Frames} \end{aligned} \tag{16}$$

The online DCF tracker works as follows: after receiving the initialized bounding box, the tracker updates the DCF according to this patch. Afterward, for all incoming voxels, we cut around the current patch a series of windows with growing sizes, where each window transverses back to the network, retrieving a response map. The maximum correlation score of each map is calculated, while each is penalized by its scale factor. As soon as the tracker has determined the best window for continuing tracking, we update the tracker's DCF, window size, and window position in preparation for the next voxel search.

As shown in Figure 1, we can partition the new architecture into distinct domains and leverage each one as an

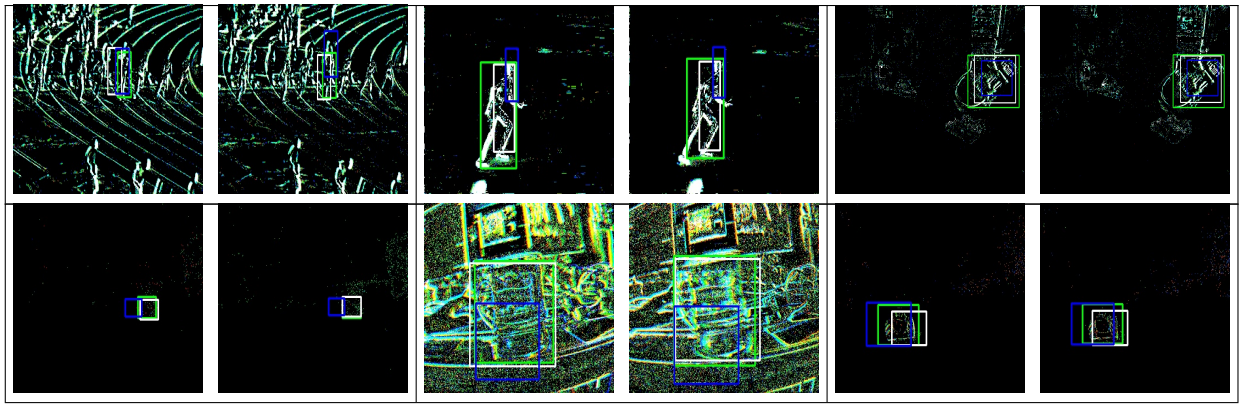

Figure 5. Selected results from the events datasets, where the first row showcases scenes from OTB2015 and the second row showcases scenes from our real-event-camera-indoor dataset.

independent tracker. The combined-domain network works as follows:

$$Net = DCFNet_{Events} + Enc - Dec_{Events} + Enc - Dec_{Frames} + DCFNet_{Frames} \quad (17)$$

3. Results

3.1. Evaluation metrics

To evaluate the performance of our ED-DCFNet tracker, we correlated the annotated ground truth (GT) and the predicted bounding boxes using the following metrics: 1. Success score – the percentage of overlap between the predicted and GT bounding boxes; 2. Precision score – the distance between the centers of the predicted and the GT bounding boxes; and 3. Success rate – the proportion of the predicted bounding boxes that have at least a 50% overlap with the GT.

	Success score	Success Rate	Precision Score
DCFNet Events	0.2731	0.303	0.3665
DCFNet Frames	0.4575	0.5449	0.5728
ED-DCFNet Events	0.2987	0.3437	0.3894
ED-DCFNet Frames	0.4326	0.5027	0.5407
ED-DCFNet combined	0.4634	0.5511	0.5756
VisEvent	0.33		

Table 1. Evaluation of the OTB2015 dataset using VisEvent [21], DCFNet grayscale frames [26] and DCFNet events. Higher scores indicate superior tracking performance.

	Success score	Success Rate	Precision Score
DCFNet Events	0.5787	0.6363	0.7071
DCFNet Frames	0.7843	0.9227	0.8739
ED-DCFNet Events	0.5995	0.6693	0.7168
ED-DCFNet Frames	0.785	0.9231	0.874
ED-DCFNet combined	0.785	0.9231	0.874
ED-DCFNet* combined	0.7873	0.9226	0.8729

Table 2. Evaluation of the real-event-camera-indoor dataset using the DCFNet grayscale frames [26] and DCFNet events. Higher scores indicate superior tracking performance.

3.2. Testing datasets

We tested our architecture on the OTB2015 dataset [22]. The OTB2015 dataset contains 100 videos, recorded at 30 fps, featuring a variety of attributes, including illumination variation and motion blur. The videos in the dataset are fully annotated and depict a wide range of scenarios, from small moving objects to static objects with camera movement. We used ESIM [11] to convert the dataset into events, employing contrast thresholds of 0.15 for both negative and positive contrasts.

We compared our results with the original DCFNet [20] and with DCFNet-events [21]. We unsupervisedly trained DCFNet with our grayscale training data (the original DCFNet was trained with color [19]) and the DCFNet-events with the OTB2015 [22] dataset. Table 1 shows the evaluation results, comparing the different modalities of our framework with DCFNet [20], and DCFNet-event. Results

Sequence Name	Changing scales	Long term moving object	Transparent glass	Under-exposure moving object	Under-exposure moving camera
Events	0.6232	0.2984	0.6156	0.4159	0.2504
Frames	0.6896	0.8926	0.4546	0.6491	0.8674
Combined	0.6896	0.8926	0.4546	0.6491	0.8674
Combined*	0.7032	0.888	0.4557	0.6547	0.8396

Table 3. The success scores of our tracker on the real-event-camera-indoor dataset, showing some of the challenging scenes results.

show that our proposed architecture, which combines both events and grayscale frames, outperformed [19], demonstrating that adding events-driven features enhances performance. We further compared our design to VisEvent [21], orders of magnitude larger supervised transformer. While our design surpasses the performance of DCFNet-events, it falls short compared to the VisEvent [21].

We further assessed the performance of ED-DCFNet on our new real-event-indoor dataset. The scenes detailed in Table 3 fall into two distinct categories: those featuring a moving object with a stationary camera and those depicting a moving camera with a stationary object. These scenes encompass diverse lighting and camera conditions. For instance, the "changing scales" scene portrays rapid movement toward and away from a stationary object, while the "long-term moving object" spans 240 frames, capturing a moving object with a stationary camera, where the object maneuvers in various directions and scales. Scenes such as those involving under-exposure, characterized by fewer events, and "Transparent glass," which features a moving camera with a transparent object, exhibit challenges including overexposure, reflective light, and rapid movement with motion blur. The results, shown in Figure 3, indicate that various scenarios may necessitate different tracking methods. For instance, in the "transparent glass" scenario, where the video contains transparent objects, the events tracker outperforms both grayscale and hybrid trackers. However, in other scenarios, the hybrid tracker may perform better, while in some cases, the frame-based tracker may exhibit superior performance. Table 2 showcases the evaluation outcomes of the entire dataset. Our ED-DCFNet architecture exhibited strong performance, achieving a success score of 0.785.

As depicted in Tables 2, 3, both the ED-DCFNet Frames and the ED-DCFNet combined achieved close results, indicating that the grayscale frame features dominate over the event voxel features. To mitigate this imbalance, we further propose a method to attenuate the signal of the grayscale frame features by modifying Equation 17 as follows:

$$\begin{aligned} Net* = {} & DCFNet_{Events} + \\ & Enc - Dec_{Events} + Enc - Dec_{Frames} + \\ & DCFNet_{Frames} * p \end{aligned} \tag{18}$$

where p represents a scale factor determining the degree of incorporation of the features into the overall feature representation. Here we chose $p = 0.3$. Results, marked with '*', were incorporated into Tables 2, 3, revealing only marginal enhancements in the success scores.

Figure 4 illustrates tracking outcomes on both the OTB2015 and our real-event-camera-indoor datasets. In the figure, the green bounding box represents the ground truth (GT), the white bounding box indicates the tracker results achieved with ED-DCFNet* when applied to both frames and events, the blue bounding box depicts the results obtained using frames tracking with DCFNet, and the red bounding box shows frames tracking with ED-DCFNet. Similarly, Figure 5 demonstrates tracking results using examples from both the OTB2015 and our real-event-camera-indoor datasets, focusing on event-driven models: ED-DCFNet and DCFNet, highlighting ED-DCFNet's enhanced tracking performance.

3.3. Ablation studies and analyses

		Success score	Success Rate	Precision Score
3 bins	Events	0.1264	0.1355	0.1828
	Frames	0.4445	0.5259	0.5523
	Combined	0.4354	0.5163	0.538
5 bins	Events	0.2987	0.3437	0.3894
	Frames	0.4326	0.5027	0.5407
	Combined	0.4634	0.5511	0.5756
9 bins	Events	0.1936	0.2247	0.2498
	Frames	0.4123	0.4784	0.5262
	Combined	0.4613	0.5544	0.5725

Table 4. Result of using different numbers of temporal bins with ED-DCFNet on the OTB2015 dataset.

To further evaluate ED-DCFNet, we compared the tracker performance with different numbers of temporal bins. Tables 4 and 5 present ED-DCFNet's tracking performance on both OTB2015 and our real-events-indoor, respectively, employing 3, 5, and 9 bins. We further evaluated ED-DCFNet*, on each configuration, tuning the p parameter to $p = 0.4$ for 3 bins, $p = 0.3$ for 5 bins, and $p = 0.1$ for 9 bins. We assessed each configuration with

		Success score	Success Rate	Precision Score
3 bins	Events	0.5343	0.5974	0.6736
	Frames	0.7823	0.9216	0.872
	Combined	0.7823	0.9216	0.872
	Combined*	0.7851	0.9226	0.8707
5 bins	Events	0.5995	0.6693	0.7168
	Frames	0.785	0.9231	0.874
	Combined	0.785	0.9231	0.874
	Combined*	0.7873	0.9226	0.8729
9 bins	Events	0.5072	0.5315	0.6388
	Frames	0.7849	0.9253	0.8726
	Combined	0.7849	0.9253	0.8726
	Combined*	0.7851	0.9229	0.8725

Table 5. Result of using different numbers of temporal bins with ED-DCFNet on our real-event-camera-indoor dataset.

event-based, frame-based, and combined inputs, using success score, success rate, and precision score. Results show that the ED-DCFNet* (Equation 18) enhances the success score, while showing a slight decrease in the other metrics. Note that the results shown in Tables 1, 2 and 3 were gained using $B = 5$ temporal bins.

Additionally, we analyzed all ED-DCFNet variations (events, frames, and their fusion) and compared their performance with DCFNet Events and DCFNet Frames (Table 6). The comparison was based on several metrics, including the number of learnable parameters (network size), network latency, working memory usage per forward or backward pass, and weight storage size for each network. Our proposed ED-DCFNet, which integrates two DCFNets and an Encoder-Decoder architecture with a single Decoder is larger than the individual DCFNets (Figure 1). However, despite its increased memory footprint, the proposed architecture maintains low latency, similar to that of the standalone DCFNets.

	Network size	Latency (mS)	Memory (MB)	Storage (KB)
ED-DCFNet Events	43,040	6.5	28.12	572
ED-DCFNet Frames	40,736	5.0	27.58	
ED-DCFNet Combined	71,424	10.3	53.17	
DCFNet Events	10,144	6.3	8.63	83
DCFNet Frames	9,568	3.78	8.49	78

Table 6. Performance analysis of the ED-DCFNet architecture, the DCFNet Events, and the DCFNet Frames.

4. Conclusions

In this work, we propose an unsupervised multi-domain learning algorithm tailored for object tracking across both events and frames. Our methodology capitalizes on unsupervised training techniques, enabling the utilization of non-annotated and synthetic datasets. Leveraging a lightweight encoder-decoder architecture, our network adeptly learns shared features from both domains, enhancing its versatility and adaptability. With merely 70K parameters, our model is lightweight and resource-efficient, making it adequate for deployment in energy-constrained embedded systems.

References

[1] Arnon Amir, Brian Taba, David Berg, Timothy Melano, Jeffrey McKinstry, Carmelo Di Nolfo, Tapan Nayak, Alexander Andreopoulos, Guillaume Garreau, Marcela Mendoza, et al. A low power, fully event-based gesture recognition system. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7243–7252, 2017. 1

[2] Anastasios N Angelopoulos, Julien NP Martel, Amit PS Kohli, Jorg Conradt, and Gordon Wetzstein. Event based, near eye gaze tracking beyond 10,000 hz. *arXiv preprint arXiv:2004.03577*, 2020. 1

[3] Avinoam Bitton, Hadar Cohen Duwek, and Elishai Ezra Tsur. Adaptive attention with a neuromorphic hybrid frame and event-based camera. In *2022 IEEE 21st International Conference on Cognitive Informatics & Cognitive Computing (ICCI* CC)*, pages 242–247. IEEE, 2022. 1

[4] David S Bolme, J Ross Beveridge, Bruce A Draper, and Yui Man Lui. Visual object tracking using adaptive correlation filters. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 2544–2550. IEEE, 2010. 4

[5] Yujeong Chae, Lin Wang, and Kuk-Jin Yoon. Siamevent: Event-based object tracking via edge-aware similarity learning with siamese networks. *arXiv preprint arXiv:2109.13456*, 2021. 1, 5

[6] Haosheng Chen, David Suter, Qiangqiang Wu, and Hanzi Wang. End-to-end learning of object motion estimation from retinal events for event-based object tracking. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 10534–10541, 2020. 1

[7] Hadar Cohen Duwek, Avinoam Bitton, and Elishai Ezra Tsur. 3d object tracking with neuromorphic event cameras via image reconstruction. In *2021 IEEE Biomedical Circuits and Systems Conference (BioCAS)*, pages 1–4. IEEE, 2021. 1

[8] Guillermo Gallego, Tobi Delbrück, Garrick Orchard, Chiara Bartolozzi, Brian Taba, Andrea Censi, Stefan Leutenegger, Andrew J Davison, Jörg Conradt, Kostas Daniilidis, et al. Event-based vision: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 44(1):154–180, 2020. 1

[9] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence

Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014. 4

[10] Anton Mitrokhin, Cornelia Fermüller, Chethan Parameshwara, and Yiannis Aloimonos. Event-based moving object detection and tracking. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–9. IEEE, 2018. 1, 5

[11] Elias Mueggler, Henri Rebecq, Guillermo Gallego, Tobi Delbruck, and Davide Scaramuzza. The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and slam. *The International Journal of Robotics Research*, 36(2):142–149, 2017. 4, 6

[12] Yonhon Ng, Yasir Latif, Tat-Jun Chin, and Robert Mahony. Asynchronous kalman filter for event-based star tracking. In *European Conference on Computer Vision*, pages 66–79. Springer, 2022. 1

[13] Bharath Ramesh, Shihao Zhang, Zhi Wei Lee, Zhi Gao, Garrick Orchard, and Cheng Xiang. Long-term object tracking with a moving event camera. In *Bmvc*, page 241, 2018. 1

[14] Bharath Ramesh, Shihao Zhang, Hong Yang, Andres Ussa, Matthew Ong, Garrick Orchard, and Cheng Xiang. e-tld: Event-based framework for dynamic object tracking. *IEEE Transactions on Circuits and Systems for Video Technology*, 31(10):3996–4006, 2020. 1

[15] Cedric Scheerlinck, Henri Rebecq, Daniel Gehrig, Nick Barnes, Robert Mahony, and Davide Scaramuzza. Fast image reconstruction with an event camera. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 156–163, 2020. 2

[16] Timo Stoffregen, Cedric Scheerlinck, Davide Scaramuzza, Tom Drummond, Nick Barnes, Lindsay Kleeman, and Robert Mahony. Reducing the sim-to-real gap for event cameras. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXVII 16*, pages 534–549. Springer, 2020. 2, 4

[17] Gemma Taverni, Diederik Paul Moeys, Chenghan Li, Celso Cavaco, Vasyl Motsnyi, David San Segundo Bello, and Tobi Delbruck. Front and back illuminated dynamic and active pixel vision sensors comparison. *IEEE Transactions on Circuits and Systems II: Express Briefs*, 65(5):677–681, 2018. 5

[18] Elishai Ezra Tsur. *Neuromorphic Engineering: The Scientist's, Algorithms Designer's and Computer Architect's Perspectives on Brain-Inspired Computing*. CRC Press, 2021. 1

[19] Ning Wang, Wengang Zhou, Yibing Song, Chao Ma, Wei Liu, and Houqiang Li. Unsupervised deep representation learning for real-time tracking. *International Journal of Computer Vision*, 129:400–418, 2021. 1, 2, 3, 6, 7

[20] Qiang Wang, Jin Gao, Junliang Xing, Mengdan Zhang, and Weiming Hu. Dcfnet: Discriminant correlation filters network for visual tracking. *arXiv preprint arXiv:1704.04057*, 2017. 1, 2, 3, 6

[21] Xiao Wang, Jianing Li, Lin Zhu, Zhipeng Zhang, Zhe Chen, Xin Li, Yaowei Wang, Yonghong Tian, and Feng Wu. Visevent: Reliable object tracking via collaboration of frame and event flows. *IEEE Transactions on Cybernetics*, 2023. 1, 6, 7

[22] Yi Wu, Jongwoo Lim, and Ming-Hsuan Yang. Object tracking benchmark. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1834–1848, 2015. 6

[23] Jiqing Zhang, Bo Dong, Haiwei Zhang, Jianchuan Ding, Felix Heide, Baocai Yin, and Xin Yang. Spiking transformers for event-based single object tracking. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pages 8801–8810, 2022. 1

[24] Jiqing Zhang, Xin Yang, Yingkai Fu, Xiaopeng Wei, Baocai Yin, and Bo Dong. Object tracking by jointly exploiting frame and event domain. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13043–13052, 2021. 1, 5

[25] Jiqing Zhang, Kai Zhao, Bo Dong, Yingkai Fu, Yuxin Wang, Xin Yang, and Baocai Yin. Multi-domain collaborative feature representation for robust visual object tracking. *The Visual Computer*, 37(9-11):2671–2683, 2021. 1

[26] Alex Zihao Zhu, Liangzhe Yuan, Kenneth Chaney, and Kostas Daniilidis. Unsupervised event-based learning of optical flow, depth, and egomotion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 989–997, 2019. 2, 6

[27] Alex Zihao Zhu, Nikolay Atanasov, and Kostas Daniilidis. Event-based visual inertial odometry. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5391–5399, 2017. 1