This CVPR Workshop paper is the Open Access version, provided by the Computer Vision Foundation.
Except for this watermark, it is identical to the accepted version;
the final published version of the proceedings is available on IEEE Xplore.

EV-LayerSegNet: Self-supervised Motion Segmentation using Event Cameras

Youssef Farah[1] Federico Paredes-Vallés[2] Guido C.H.E. De Croon[2]
Muhammad Ahmed Humais[1] Hussain Sajwani[1] Yahya Zweiri[1]
[1]Advanced Research and Innovation Center, Khalifa University [2]MAVLab, TU Delft
[1]{youssef.farah, 100061899, hussain.sajwani, yahya.zweiri}@ku.ac.ae
[2]fedeparedesv@gmail.com, [2]G.C.H.E.deCroon@tudelft.nl

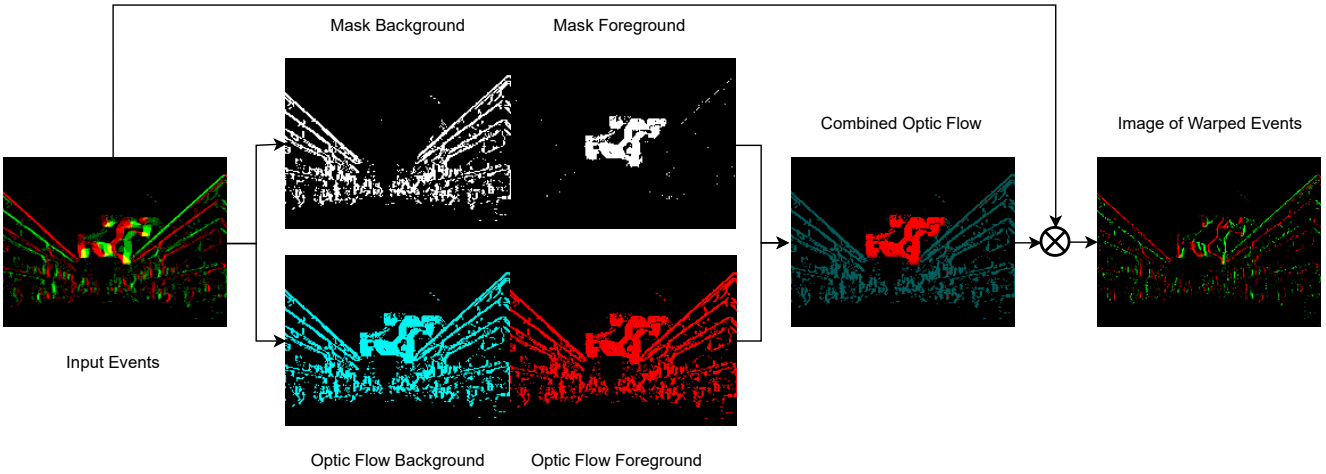


Figure 1. Our method takes as input an event volume that results in a blurry scene. It then attempts to generate two different masks to differentiate the background from the foreground. Next, it estimates the affine optical flow for both models, and combines the flow together using the masks. Finally, it warps the events according to the combined flow. A successful motion deblur leads to accurate segmentation.

Abstract

Event cameras are novel bio-inspired sensors that capture motion dynamics with much higher temporal resolution than traditional cameras, since pixels react asynchronously to brightness changes. They are therefore better suited for tasks involving motion such as motion segmentation. However, training event-based networks still represents a difficult challenge, as obtaining ground truth is very expensive, error-prone and limited in frequency. In this article, we introduce EV-LayerSegNet, a self-supervised CNN for event-based motion segmentation. Inspired by a layered representation of the scene dynamics, we show that it is possible to learn affine optical flow and segmentation masks separately, and use them to deblur the input events. The deblurring quality is then measured and used as self-supervised learning loss. We train and test the network on a simulated dataset with only affine motion, achieving IoU and detection rate up to 71% and 87% respectively.

1. Introduction

Event cameras, such as the Dynamic Vision Sensor (DVS), are novel bio-inspired sensors that perceive motion by detecting brightness changes instead of capturing intensity changes in images within a time interval [4]. In other words, if at time t a brightness change larger than the threshold C is detected by a pixel with x and y coordinates, an event is generated by recording the brightness increase or decrease, the pixel location and the time the event was generated. In contrast, traditional cameras record color intensities in all pixels at a specified frame rate.

This fundamental shift in the acquisition of visual information brings several advantages. High temporal resolution and low latency (in the order of μs) make event cameras well suited for recording very fast moving scenes without suffering from motion blur as frame-based cameras. The High Dynamic Range (HDR) allows the cameras to capture details in very dark or very bright scenes, making them appropriate to be used in light challenging environments

such as the underwater ocean, night driving or fireworks display [38].

The combination of these advantages unlocks the possibility of performing tasks inaccessible until now for frame-based cameras such as low-latency optical flow estimation, high-speed control and tracking, and Simultaneous Localization and Mapping (SLAM) [4]. Nevertheless, a new generation of event-based vision algorithms needs to be developed to enable the event cameras to express their full potential, as the data structure of events is not compatible with the conventional algorithms based on images.

Furthermore, an additional challenge is posed by the lack of ground truth from real-word datasets, at microsecond resolution and with HDR. In image-based datasets (COCO [16], BD100K [36]), the ground truth is often obtained by humans manually annotating each frame. This process is already expensive and laborious for image-based datasets, and becomes unfeasible for event data, which are sparse and have low latency. To avoid manual annotation, additional sensors and cameras were used to generate the ground truth in traditional event-based datasets such as DSEC [7] and MVSEC [41]. Yet, sensors and cameras are limited by their natural Field Of View (FOV), spatial and temporal resolution. Therefore, it is of paramount importance to develop algorithms that are able to perform their tasks by relying on the nature of events, without the use of ground truth that are expensive to collect and that carry error measurements from additional sensors.

In the context of motion estimation and its use in other tasks such as surveillance, tracking or obstacle avoidance, segmenting the scene into independently moving objects is fundamental and often referred as *motion segmentation* [27]. Even though great progress has been made using frame-based cameras, the latter are not ideally suited for tasks involving motion as they suffer from motion blur and keep track of all pixel values even when no motion occurs [27]. Under constant illumination, event cameras are well suited for this task, since events are sampled at exactly the same rate of the scene dynamics and the information acquired is only related to the motion of the camera and the objects in the scene.

Event-based motion segmentation can be broadly categorized in model-based methods [21, 27, 40] and learning-based methods [19, 20, 25]. Model-based methods exploit the nature of event data to cluster events generated by Independently Moving Objects (IMO) in an iterative fashion, thus they are unsupervised and do not use ground truth. Yet, the iterative approach requires long computational time and limits their real-word applicability. On the other hand, learning-based methods apply deep learning techniques while also relying on the event characteristics. In some instances, they perform motion segmentation in combination with other tasks such as tracking and optical flow, but they all require ground truth or pretrained weights during training. To the best of our knowledge, there is no existence of a learning-based method that is self-supervised, thus learning directly from the input events instead of relying on ground truth or pretrained weights from supervised methods.

Looking outside of the event domain, significant progress has been made in self-supervised image segmentation. These methods segment the scene into the dominant moving object as foreground and the rest as background, and learning is driven by image or optic flow reconstruction. In most instances, they frequently make use of layered representation [29]. The pixels are separated into layers based on motion similarity, and they are moved according to the associated motion and synthesized together to reconstruct the next image in the sequence. Recently, Shrestha *et al*. [26] integrated this process into a end-to-end differentiable CNN pipeline which segments the dominant moving object in a video using two consecutive frames.

Inspired by their work, we propose a self-supervised convolutional neural network for motion segmentation using events. Specifically, we transfer the CNN segmentation approach from [26] to the event domain, and combine it with the encoder-decoder structure from [43]. We show that, under the assumption of affine motion and constant brightness, it is possible to segment independently moving objects using contrast maximization loss.

We identify our main contributions as follows:

1. We propose a novel self-supervised network for learning event-based motion segmentation by introducing a novel optical flow module that enables self-supervised learning of affine optical flow and a segmentation module that learns separately the masks corresponding to the independently moving objects.
2. We contribute a new event-based dataset of several simulated backgrounds and objects moving according to affine motion.
3. We demonstrate through experiments that our method shows superior performance in comparison with the start-of-the-art in unsupervised motion segmentation.

2. Related Work

In computer vision, motion segmentation is defined as the task of retrieving the shape of moving objects [39]. Image difference such as in [37] attempts to detect moving objects by finding intensity differences between pixels across video frames. Layer-based techniques [6, 11] divide frames into layers, based on the number of uniform motions. Statistical methods such as Expectation Maximization (EM) are among the most common methods used. Recent developments in deep learning and optical flow estimation allowed a sharp improvement in this motion task, often combined with statistics and layer-based techniques. However, the

main limitation of the state-of-the-art methods in motion segmentation is the reliance on ground truth labels, which limit the applicability of such methods scenes outside the annotated datasets. In this regard, attempts to develop self-supervised methods have resulted in frameworks developed in the frame domain, while methods for event cameras were yet to be found.

2.1. Frame-based Motion Segmentation

In literature, many methods are considered unsupervised or self-supervised, but only during inference and test. In reality, parts of their architecture (i.e. networks, masks) are pretrained on ground truth. For example, COSNet [17] uses co-attention to capture rich correlations between frames of a video, but masks are pretrained on ground truth. Similarly, Ye *et al*. [35] perform motion segmentation using global sprites, but it also requires precomputed masks. Li *et al*. [14] propose the use of instance embeddings to find the moving object based on motion saliency and objectness. However, the dense embeddings are obtained from an instance segmentation that is performed by a network pretrained on static images. In contrast, we consider as self-supervised methods only the ones which do not need any type of ground truth during training and inference, and do not rely on pretrained weights from supervised methods.

Early attempts of self-supervised or unsupervised learning of motion segmentation start from ConvNet [23]. The network learns high-level features in single frames as follows. A set of single frames with optical flow maps is passed to the network. It then generates the segmentation masks by associating optical flow vectors with similar direction and rate. Using these masks as pseudo-ground truth, the network tries to reproduce the masks by looking only at the static frames .

Yang *et al*. [31] propose a network that performs foreground/background motion segmentation by using a precomputed optic flow as input. Slot attention is then used to group together pixels with visual homogeneity, but the result are very sensitive to the accuracy of the optical flow estimation.

While in [26, 31] the models try to segment the moving object by using the optical flow of the object itself, Yang *et al*. [33] propose to find the moving object by using a Generator (G) and an Inpainter (I) network. The generator applies a mask to the input optic flow, with the aim of hiding the optic flow of the foreground. Next, the generator passes the masked optic flow to the inpainter, which has the task of reconstructing the optic flow that has been hidden by the generator. Accurate motion segmentation is achieved when the optic flow reconstruction works poorly. It obtains good result in public datasets, however the model fails to capture complete objects or differentiate regions in the background. This is because it operates with one single scale temporal information and it also introduces the bias from the camera motion. To solve this issue, Yang *et al*. [32] extend this work with MASNet, where they expand the network pipeline with more generator and inpainter modules to deal with multiple input optic flow maps.

Finally, Shrestha *et al*. [26] propose LayerSegNet, a network that combines a layered representation of the scene and affine optical flow estimation. The network takes two images as input and aims to separately estimate the motion models (foreground and background) and the masks. Subsequently, they apply the masks to the first image and warp the masks according to the associated motion models. They then combine the two masks together to generate the second image, and they use the difference between the generated second image and actual second image as learning parameter.

2.2. Event-based Motion Segmentation

Event-based motion segmentation is very recent and attempts to make use of the outstanding properties of event cameras.

Stoffregen *et al*. [27] use Expectation Maximization and propose the idea of segmenting moving objects by simultaneously grouping the events into different clusters and estimating the motion model associated to each cluster. The events are then warped according to the associated motion model to produce an Image of Warped Events (IWE). An iteration process is then used to find the right motion models and clusters such that the objective function is maximized. The model is proven not to be sensitive to the number of clusters specified in the iteration process and works well in real-world dataset,but it is not able to estimate the number of moving objects in the scene.

Zhou *et al*. [40] propose to solve this issue by introducing two spatial regularizers, which minimize the number of clusters. The underlying concept remains the same as in [27], however the input events are first initialized in a space-time event graph cut and the clusters are smoothly sharped via the addition of energy terms in the objective function.

Always based on event clustering, Chethan *et al*. [21] split the scene into multiple motions and merge them, allowing also for feature tracking.

Despite these methods produce promising results, they are not appropriate to be deployed in real-world applications as the iteration processes takes considerable amount of time and computational effort, which inhibit the potential applications on mobile platforms such as drones. Therefore, a learning-based approach needs to be developed.

Sanket *et al*. [25] propose EVDodgeNet, a learning-based pipeline that solves simultaneously motion segmentation, optical flow and 3-D motion, but relies on ground truth masks. Instead, Mitrokhin *et al*. [19] use Graph Con-

volutional Networks to learn motion segmentation from a 3-D representation of events. Chethan *et al*. [20] also propose SpikeMS, the first Spiking Neural Network (SNN) for event-based motion segmentation. Very recently, Georgoulis *et al*. [8] performs the task by first compensating the events with egomotion, and then use attention modules for temporal consistency. Alkendi *et al*. [1] uses instead Graph Transformers neural networks to denoise the scene and perform segmentation.

Despite these efforts, the aforementioned methods continue to depend on ground truth, leaving a gap in the availability of self-supervised approaches. In this direction, Wang *et al*. [30] generate pseudo-labels from geometrical constraints, and Arja *et al*. [2] propose the use of self-supervised vision transformers. Both works rely on pretrained weights from supervised networks.

Inspired by LayerSegNet [26], we propose an end-to-end CNN architecture that learns motion segmentation by jointly estimating affine optical flow and segmentation masks, using contrast maximization as learning loss and without using any sort of pretrained weights.

3. Method

The approach to the segmentation task is inspired by [26]. However, given the unique nature of event data, a distinct approach to the input is needed. To this end, we use the input event representation from [43]. We then jointly estimate the segmentation masks and the affine optical flow maps. Next, we apply the segmentation masks to the associated optical flow, and we combine these maps to obtain a single optical flow map. We then use this map to warp the events forward and backward in time, and we apply the self-supervised loss as in [9]. The more the image of warped events is deblurred with respect to the input events, the more accurate are the segmentation masks.

3.1. Input Event Representation

Selecting the right representation of events for a given task is still a challenging problem. Event-based methods can process single events $e_k \doteq (x_k, t_k, p_k)$, however events alone carry very little information and are subject to noise. It is often preferred to process a group of events $\mathcal{E} \doteq \{e_k\}_{k=1}^{N_e}$ that yields a sufficient signal-to-noise ratio, which also carries more information for the given task. The most common methods either discretize the group of events in different frames of event counts [12, 13, 22, 28, 43] or the per-pixel average/most recent event timestamps [15, 34, 42].

Since we develop a non-recurrent CNN pipeline, temporal information need to be encoded in the input, thus we adopt the event representation from [43]. Given N input events and B bins, we first scale the event timestamps in the range $[0, B-1]$ and generate the event volume:

$$t_i^* = (B-1)\,(t_i - t_0) \,/\, (t_N - t_1) \tag{1}$$

$$V(x,y,t) = \sum_i p_i k_b\,(x - x_i)\,k_b\,(y - y_i)\,k_b\,(t - t_i^*) \tag{2}$$

$$k_b(a) = \max(0, 1 - |a|) \tag{3}$$

In order to perform 2D convolutions, the time domain is treated as a channel in a traditional 2D image.

3.2. Motion Model

The authors in [26] propose the segmentation of the dominant moving object in the scene using a two-layer representation. The foreground layer captures the dominant object, while the background layer represents the rest of the scene.

In our case, assuming constant illumination, events are generated by the relative motion between the camera and the scene. Since events are triggered by brightness changes, these are produced by moving edges. Hence, if the camera is moving and the scene is static, the events are generated by the motion of the camera. When objects are also moving within the scene, the generation of events occurs due to both the edges of moving objects and the edges of static objects caused by the egomotion of the camera.

Let us assume a scene where an object and background are moving distinctively. We also assume that the motion is affine, hence the scene may undergo translation, rotation, shearing and scaling. The two motions can then be described by two motion models A_1 and A_2:

$$W_i(x,y) = \mathbf{A}_i \begin{bmatrix} 1 \\ x \\ y \end{bmatrix} = \begin{bmatrix} a_i^1 & a_i^2 & a_i^3 \\ a_i^4 & a_i^5 & a_i^6 \end{bmatrix} \begin{bmatrix} 1 \\ x \\ y \end{bmatrix}, \quad i \in \{1,2\} \tag{4}$$

where W_i represent the dense flow maps corresponding to affine motion matrix A_i.

Corresponding to the two affine motions, we separately generate two alpha masks, which represent the moving objects. Ideally, the segmentation masks should be a one-hot vector for each pixel, which assigns the pixel to the corresponding object. However, such operation would be non differentiable, preventing the backpropagation of the gradients during training. We overcome this issue by assigning two arbitrary numbers for each pixel, where each number corresponds to one layer. We then apply softmax to ensure that the values are bounded in [0,1] and the values sum up to 1. For each pixel, we then retain the maximum value in the two layers, and set the other one to zero. This operation, called maxout operation, makes the classification differentiable. It also makes sure that one pixel can not belong to two objects at the same time.

Next, we compute the combined optical flow by element-wise multiplication of the affine flow maps with the corresponding alpha maps:

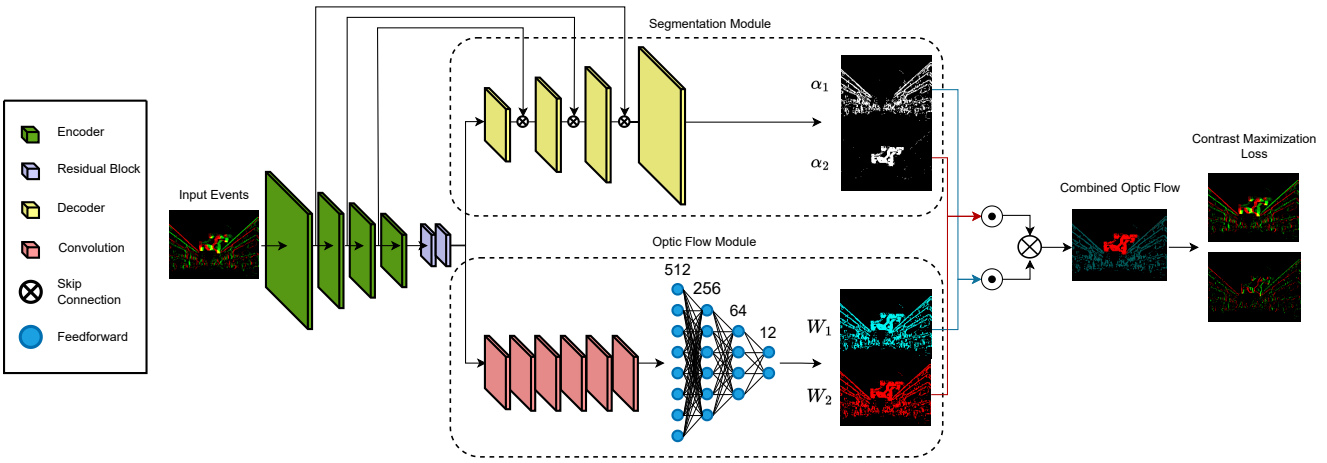


Figure 2. EV-LayerSegNet architecture. Events are downsampled by 4 encoding layers before being passed to 2 residual blocks. The output of the residual blocks is then passed to the segmentation and optical flow module. The segmentation module upsamples with 4 decoder layers connected with the encoders by skip connections. The optical flow module is composed of 6 convolutions and a feedforward network of 4 layers.

$$W_{\text{comb}} = \alpha_1 \odot W_1 + \alpha_2 \odot W_2 \tag{5}$$

The maxout operation ensures that each pixel will have flow components from only one motion model, scaled with the respective value of the alpha map to which the pixel is associated. During training, we expect that the network will adjust itself to the scaling effect of the flow computation.

3.3. Self-supervision by Contrast Maximization

Self-supervised learning is achieved by applying contrast maximization [5]. As described in [9], events which are generated by the same moving edge encode exact optical flow. Since these events are misaglined in the input partition (motion blur), the events can be propagated to a reference time t_{ref} using per-pixel optical flow $\boldsymbol{u}(\boldsymbol{x}) = (u(\boldsymbol{x}), v(\boldsymbol{x}))^T$ to realign them and show the initial edge that has generated the events:

$$\boldsymbol{x}_i' = \boldsymbol{x}_i + (t_{\text{ref}} - t_i)\, \boldsymbol{u}\,(\boldsymbol{x}_i) \tag{6}$$

The metric used to measure the deblurring quality is the the per-pixel and per-polarity average timestamp of the image of warped events [18, 43]. The lower the loss, the better the deblurring, which consequently means that the estimated optical flow and alpha maps are more accurate.

We initially generate an image of the average timestamp at each pixel for each polarity using bilinear interpolation, as in [9]:

$$\begin{aligned} T_{p'}\,(\boldsymbol{x}; \boldsymbol{u} \mid t_{\text{ref}}) &= \frac{\sum_j \kappa(x - x_j')\kappa(y - y_j')t_j}{\sum_j \kappa(x - x_j')\kappa(y - y_j') + \epsilon} \\ j = \{i \mid p_i = p'\}, &\quad p' \in \{+, -\}, \quad \epsilon \approx 0 \end{aligned} \tag{7}$$

The loss is then the sum of the squared temporal images. This is also scaled by the sum of pixels with at least one event, in order to prevent the network from keeping events with large timestamps out of the image space, such that they would not contribute to the loss function.

$$\mathcal{L}_{\text{contrast}}\ (t_{\text{ref}}) = \frac{\sum_{\boldsymbol{x}} T_{\pm}\,(\boldsymbol{x}; \boldsymbol{u} \mid t_{\text{ref}})^2}{\sum_{\boldsymbol{x}} \left[n\,(\boldsymbol{x}') > 0\right] + \epsilon} \tag{8}$$

The warping process is performed both forward (t_{ref}^{fw}) and backward (t_{ref}^{bw}) to prevent temporal scaling issues during backpropagation. The total loss is then the sum of the backward and forward warping loss, and λ is a scalar balancing the Charbonnier smoothness prior $\mathcal{L}_{\text{smooth}}$ [3].

$$\mathcal{L}_{\text{contrast}} = \mathcal{L}_{\text{contrast}}\ \left(t_{\text{ref}}^{\text{fw}}\right) + \mathcal{L}_{\text{contrast}}\ \left(t_{\text{ref}}^{\text{bw}}\right) \tag{9}$$

$$\mathcal{L}_{\text{flow}} = \mathcal{L}_{\text{contrast}} + \lambda \mathcal{L}_{\text{smooth}} \tag{10}$$

Notice that the self-supervised loss becomes a strong supervisory signal only when there is enough blur in the input event partition. It is of paramount importance to check that the input partition contains enough events for linear blur.

4. Network Implementation

Our network EV-LayerSegNet is similar to encoder-decoder architectures and is inspired by [26, 43].

As in [43], events are downsampled by 4 encoder layers and passed to 2 residual blocks. We then stack the output of the residual blocks and pass it to the segmentation module and optical flow module.

4.1. Optical Flow Module

The optical flow module contains 6 convolutional layers, each followed by leaky ReLU activation. We then flatten the output of the last convolution layer and pass it to a feedforward network consisting of 4 layers (512, 256, 64 and 12 output units), followed by tanh activation except the last layer. We then use the output of the feedforward network and split it to two sets of 6 affine motion parameters, and we compute the two flow maps W_1 and W_2.

4.2. Segmentation Module

In the segmentation part, the output of the residual blocks is bilinearly upsampled by 4 decoding layers. Each decoding layer is connected to the respective encoding layer by skip connection. We apply a leaky double-rectified ReLU activation (*leaky DoReLU*) [26] which promises an increase in segmentation performance:

$$y(x) = \begin{cases} 1 + \frac{x-1}{\gamma} & , x > 1 \\ x & , \text{ if } 0 \leq x \leq 1 \\ \frac{x}{\gamma} & , x < 0 \end{cases} \tag{11}$$

In this module, each decoding layer is followed by leaky DoReLU activation with γ = 100. At the last layer, softmax is applied instead to ensure that the channel values are bounded [0,1] and sum up to 1.

5. Experiments

5.1. Training Details

We implement our pipeline in Pytorch. We use Adam optimizer [10] and a learning rate of $1 \cdot 10^{-5}$. At training, we used a batch size of 8 and train for 400 epochs. The input consists of N = 200,000 events, which is a sufficient number of events to obtain blur in the scene. The Charbonnier loss is balanced with weight λ = 0.001 .

5.2. Dataset

Due to the absence of a sufficiently large dataset with objects moving with affine motion, we train EV-LayerSegNet on our dataset, generated using ESIM [24]. Specifically, we make use of the Multiple 2D Objects rendering engine to generate the simulated events. We use similar settings as in a DVS camera, with resolution 640x480 pixels and threshold C = 0.5. For the training set, we generate 1000 sequences of 1 second each, using 10 background images and 10 foreground objects. Similarly, the test set is composed of 25 sequences of 1 second each, with five background images and five foreground images. The ground truth is generated by using a threshold on optical flow, since the foreground objects move significantly faster than the background images.

5.3. Evaluation Metrics

We use *Intersection over Union* (IoU) and *Detection Rate* (DR) as standard metrics for our quantitative evaluation [40]. IoU is formulated as in Eq. (12), where S_D stands for the predicted mask and S_G for the ground truth mask:

$$\text{IoU} = \frac{S_D \cap S_G}{S_D \cup S_G} \tag{12}$$

Detection rate returns 1 if the following conditions in Eq. (13) are met, where B_D and B_G are the bounding boxes for predicted mask and ground truth:

$$B_D \cap B_G > 0.5 \quad \text{and} \quad (B_D \cap B_G) > (S_D \cap \overline{B_G}) \tag{13}$$

6. Results

6.1. Qualitative Evaluation

We provide a qualitative example in Fig. 3. The network effectively deblurs the scene, composed of a bird flying in the same direction of the camera but at a higher speed, as encoded by the colors of the optical flow maps. This shows that our method is also capable of segmenting challenging scenarios in which the background and foreground might appear to have similar motions, which could lead the network to group them together. Fig. 4 shows the same scene compared to ground truth and the respective motion segmentation estimated by EMSMC [27] and EMSGC [40]. Our method correctly clusters the bird in the right class, but is subject to noise, as noticed from the sparse events of the background. In this case, EMSMC figuratively shows a better segmentation, while EMSGC segments the bird in different clusters. In the quantitative evaluation, although EMSGC mislabels the foreground in different clusters, we consider all clusters minus the background as one main cluster.

6.2. Quantitative Evaluation

We benchmark our method against EMSMC [27] and EMSGC [40] using mean intersection over union (Mean IoU) and mean detection rate (Mean DR) on our test dataset. This approach is relevant since both methods do not require any sort of ground truth or pretrained weights to perform in testing. The benchmark with EMSMC is fair since both methods require an initial number of clusters, which in our case is two (background and foreground). We compare results also with EMSGC, while keeping in mind the increased challenge of solving also for the increased number of clusters. Due to the high motion present in the test dataset, we set N = 50,000 for all methods. This choice was also motivated due to the extensive computational time needed for both EMSMC and EMSGC to perform. All methods have found challenging the test dataset, having relevant performances in 6 sequences shown in Tab. 1. In most

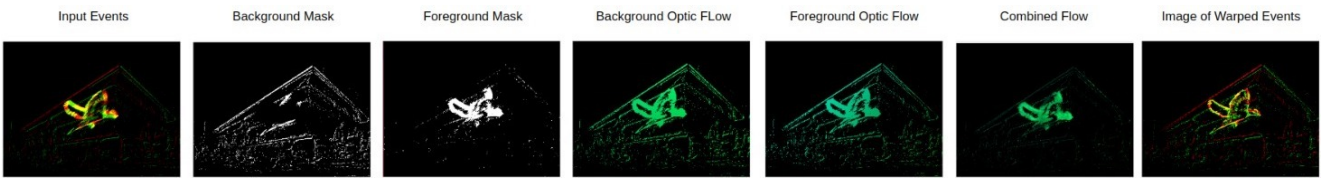


Figure 3. Test result example on synthetic dataset ("Bird in front of building" in Tab. 1). The color scheme can be found in [9].

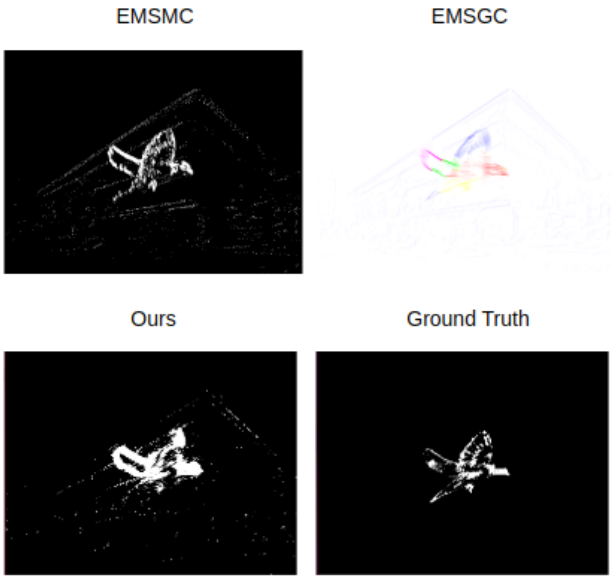


Figure 4. Qualitative comparison of test results from EMSMC [27], EMSGC [40] and our method against ground truth ("Bird in front of building" in Tab. 1).

cases, our method demonstrates higher performance in segmenting the scene. In addition to this, our method performs on-the-fly motion segmentation, requiring few milliseconds to get the masks, while EMSMC and EMSGC, due to their iterations, require several minutes.

7. Ablation Study

7.1. Leaky ReLU vs Leaky DoReLU

In [26] the authors discovered that using leaky DoReLU activation function instead of leaky ReLU improves segmentation. The key distinction between these two activation functions lies in the restriction of the slope for input values above 1 in leaky DoReLU. The authors argue that the reason for the segmentation improvement with leaky DoReLU is the enhanced stability of gradients, considering their alpha maps are limited to the range [0,1].

Inspired by their segmentation approach, we chose to implement leaky DoReLU instead of leaky ReLU. Nevertheless, we investigated the significance of this modified activation function in our specific case.

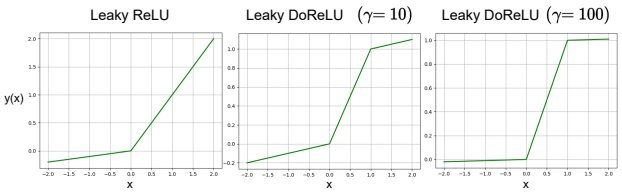


Figure 5. Visualization of leaky ReLU (left) and leaky DoReLU with slope coefficient γ = 10 (center) and γ = 100 (right). The x-axis is the input x and the y-axis is the output $y(x)$.

For this reason, we created an alternative EV-LayerSegNet architecture where leaky ReLU activations are used instead of leaky DoReLU. We trained it on our dataset and subsequently tested it.

The results in Tab. 2 show that the network encounters more difficulties compared to the baseline. This observation suggests that the leaky DoReLU activation improves our network's ability to achieve successful segmentation.

7.2. Decreasing the slope in Leaky DoReLU

By establishing the importance of the leaky DoReLU activation function in our segmentation module, we turned our attention to exploring the impact of modifying the slope coefficient γ. We investigated how the variation γ would affect the segmentation performance of our network.

The characteristics of the activation functions involved are shown in Fig. 5. It can be observed that the leaky DoReLU exhibits a smaller slope compared to the leaky ReLU beyond the range of [0, 1]. This reduced slope plays a vital role in stabilizing the flow of gradients. We suspect that this is due to the function being more non-linear, improving our segmentation performance. To test this hypothesis, we decide to train the network with a higher slope (γ = 10), therefore expecting poorer results.

The corresponding test results are presented in Tab. 3, where the baseline method performs significantly better. This suggests that a nearly flat slope outside of range [0, 1] in the leaky DoReLU function plays a vital role for segmentation.

	Mean IoU			Mean Detection Rate		
Sequence name	**EMSMC**	**EMSGC**	**Ours**	**EMSMC**	**EMSGC**	**Ours**
Drone above playground	0.20	0.05	0.71	0.00	0.00	0.87
Plane over city	0.06	0.03	0.67	0.00	0.00	0.77
Bird above playground	0.26	0.00	0.48	0.00	0.00	0.71
Second drone above playground	0.36	0.02	0.52	0.00	0.00	0.78
Bird in front of building	0.20	0.41	0.10	0.00	0.21	0.00
Helicopter over city	0.07	0.00	0.55	0.00	0.00	0.83

Table 1. Comparison between our method, EMSMC [27] and EMSGC [40] on 6 simulated test sequences using an identity camera matrix.

	Mean IoU		Mean Detection Rate	
Sequence name	**LeakyReLU**	**Ours**	**LeakyReLU**	**Ours**
Drone above playground	0.38	0.71	0.00	0.87
Plane over city	0.48	0.67	0.65	0.77
Bird above playground	0.31	0.48	0.00	0.71
Second drone above playground	0.48	0.52	0.78	0.78
Helicopter over city	0.45	0.55	0.61	0.83

Table 2. Test results of our method with Leaky ReLU against baseline.

	Mean IoU		Mean Detection Rate	
Sequence name	**γ = 10**	**Ours**	**γ = 10**	**Ours**
Drone above playground	0.06	0.71	0.00	0.87
Plane over city	0.03	0.67	0.00	0.77
Bird above playground	0.07	0.48	0.00	0.71
Second drone above playground	0.02	0.52	0.00	0.78
Helicopter over city	0.21	0.55	0.00	0.83

Table 3. Test results of our method with γ = 10 against baseline. Segmentation performance significantly deteriorates across all scenes.

8. Conclusion

We present a novel end-to-end CNN network that performs self-supervised motion segmentation using event cameras. We take inspiration from the state-of-the-art methods in unsupervised motion segmentation, and use the latest development in event-based optical flow estimation using artificial neural networks. We show that it is possible to learn motion models and segmentation masks separately from a blurry scene, and combine them together to deblur the input events. To train and test our method, we generate a new dataset with background and objects moving affinely. We show that the network is able to correctly segment the background and the independently moving objects in the event stream. We also show that our method outperforms the state-of-the-art in unsupervised motion segmentation. On the other hand, our method is limited to affine motion and is sensitive to very high blurriness and noise. For future work, we seek to extend the training and testing dataset to have more diversity, and benchmark the method against other self-supervised methods that require pretrained weights, and finally supervised methods. In the longer term, we seek to test on a 3D real-world dataset. Additional recommendations would be to improved the optical flow module such that also non-affine motions are estimated, and develop the method using Spiking Neural Networks, therefore allowing for fast and low-power computation on board of weight-constrained vehicles such as drones. We hope that our work draws attention to the potential that self-supervised learning can unlock for event-based cameras, allowing learning-based methods to be trained and used without expensive ground-truth annotations.

Acknowledgments

This work is supported by Sandooq Al Watan under Grant SWARD-S22-015, STRATA Manufacturing PJSC, and Advanced Research and Innovation Center (ARIC), which is jointly funded by Aerospace Holding Company LLC, a wholly-owned subsidiary of Mubadala Investment Company PJSC and Khalifa University for Science and Technology.

References

[1] Yusra Alkendi, Rana Azzam, Sajid Javed, Lakmal Seneviratne, and Yahya Zweiri. Neuromorphic vision-based motion segmentation with graph transformer neural network. *Trans. Multi.*, 27:385–400, 2025. 4

[2] Sami Arja, Alexandre Marcireau, Saeed Afshar, Bharath Ramesh, and Gregory Cohen. Motion segmentation for neuromorphic aerial surveillance, 2024. 4

[3] P. Charbonnier, L. Blanc-Feraud, G. Aubert, and M. Barlaud. Two deterministic half-quadratic regularization algorithms for computed imaging. In *Proceedings of 1st International Conference on Image Processing*, pages 168–172 vol.2, 1994. 5

[4] Guillermo Gallego, Tobi Delbruck, Garrick Orchard, Chiara Bartolozzi, Brian Taba, Andrea Censi, Stefan Leutenegger, Andrew Davison, Jorg Conradt, Kostas Daniilidis, and Davide Scaramuzza. Event-based vision: A survey. 2019. 1, 2

[5] Guillermo Gallego, Mathias Gehrig, and Davide Scaramuzza. Focus is all you need: Loss functions for event-based vision. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 12272–12281, 2019. 5

[6] Liyue Ge, Congxuan Zhang, Zhen Chen, and Ming Li. Optical flow estimation from layered nearest neighbor flow fields. pages 1–6, 2018. 2

[7] Mathias Gehrig, Willem Aarents, Daniel Gehrig, and Davide Scaramuzza. Dsec: A stereo event camera dataset for driving scenarios. *IEEE Robotics and Automation Letters*, 6(3): 4947–4954, 2021. 2

[8] Stamatios Georgoulis, Weining Ren, Alfredo Bochicchio, Daniel Eckert, Yuanyou Li, and Abel Gawel. Out of the Room: Generalizing Event-Based Dynamic Motion Segmentation for Complex Scenes . In *2024 International Conference on 3D Vision (3DV)*, pages 442–452, Los Alamitos, CA, USA, 2024. IEEE Computer Society. 4

[9] Jesse Hagenaars, Federico Paredes-Valles, and Guido de Croon. Self-supervised learning of event-based optical flow with spiking neural networks. In *Advances in Neural Information Processing Systems*, pages 7167–7179. Curran Associates, Inc., 2021. 4, 5, 7

[10] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014. 6

[11] M. Kumar, Philip Torr, and A. Zisserman. Learning layered motion segmentations of video. *International Journal of Computer Vision*, 76:301–319, 2008. 2

[12] Chankyu Lee, Adarsh Kosta, Alex Zihao Zhu, Kenneth Chaney, Kostas Daniilidis, and Kaushik Roy. Spike-flownet: Event-based optical flow estimation with energy-efficient hybrid neural networks. In *European Conference on Computer Vision*, 2020. 4

[13] Chankyu Lee, Adarsh Kumar Kosta, and Kaushik Roy. Fusion-flownet: Energy-efficient optical flow estimation using sensor fusion and deep fused spiking-analog network architectures. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 6504–6510, 2022. 4

[14] Siyang Li, Bryan Seybold, Alexey Vorobyov, Alireza Fathi, Qin Huang, and C.-C. Jay Kuo. Instance embedding transfer to unsupervised video object segmentation. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6526–6535, 2018. 3

[15] Zhuoyan Li, Jiawei Shen, and Ruitao Liu. A lightweight network to learn optical flow from event data. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 1–7, 2021. 4

[16] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Doll'a r, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014. 2

[17] Xiankai Lu, Wenguan Wang, Chao Ma, Jianbing Shen, Ling Shao, and Fatih Porikli. See more, know more: Unsupervised video object segmentation with co-attention siamese networks. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3618–3627, 2019. 3

[18] 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, 2018. 5

[19] Anton Mitrokhin, Zhiyuan Hua, Cornelia Fermüller, and Yiannis Aloimonos. Learning visual motion segmentation using event surfaces. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 14402–14411, 2020. 2, 3

[20] Chethan M. Parameshwara, Simin Li, Cornelia Fermüller, Nitin J. Sanket, Matthew S. Evanusa, and Yiannis Aloimonos. Spikems: Deep spiking neural network for motion segmentation. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3414–3420, 2021. 2, 4

[21] Chethan M. Parameshwara, Nitin J. Sanket, Chahat Deep Singh, Cornelia Fermüller, and Yiannis Aloimonos. 0-mms: Zero-shot multi-motion segmentation with a monocular event camera. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9594–9600, 2021. 2, 3

[22] Federico Paredes-Vallés and Guido C. H. E. de Croon. Back to event basics: Self-supervised learning of image reconstruction for event cameras via photometric constancy. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3445–3454, 2021. 4

[23] Deepak Pathak, Ross Girshick, Piotr Dollár, Trevor Darrell, and Bharath Hariharan. Learning features by watching objects move. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6024–6033, 2017. 3

[24] Henri Rebecq, Daniel Gehrig, and Davide Scaramuzza. ESIM: an open event camera simulator. *Conf. on Robotics Learning (CoRL)*, 2018. 6

[25] Nitin J. Sanket, Chethan M. Parameshwara, Chahat Deep Singh, Ashwin V. Kuruttukulam, Cornelia Fermüller, Davide Scaramuzza, and Yiannis Aloimonos. Evdodgenet: Deep dynamic obstacle dodging with event cameras. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10651–10657, 2020. 2, 3

[26] Sahir Shrestha, Mohammad Ali Armin, Hongdong Li, and Nick Barnes. Learning to segment dominant object motion from watching videos. In *2021 Digital Image Computing: Techniques and Applications (DICTA)*, pages 01–08, 2021. 2, 3, 4, 5, 6, 7

[27] Timo Stoffregen, Guillermo Gallego, Tom Drummond, Lindsay Kleeman, and Davide Scaramuzza. Event-based motion segmentation by motion compensation. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7243–7252, 2019. 2, 3, 6, 7, 8

[28] Timo Stoffregen, Cedric Scheerlinck, Davide Scaramuzza, Tom Drummond, Nick Barnes, Lindsay Kleeman, and Robert E. Mahony. Reducing the sim-to-real gap for event cameras. In *European Conference on Computer Vision*, 2020. 4

[29] J.Y.A. Wang and E.H. Adelson. Representing moving images with layers. *IEEE Transactions on Image Processing*, 3 (5):625–638, 1994. 2

[30] Ziyun Wang, Jinyuan Guo, and Kostas Daniilidis. Unevmoseg: Unsupervised event-based independent motion segmentation, 2023. 4

[31] Charig Yang, Hala Lamdouar, Erika Lu, Andrew Zisserman, and Weidi Xie. Self-supervised video object segmentation by

motion grouping. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 7157–7168, 2021. 3

[32] Fan Yang, Srikrishna Karanam, Meng Zheng, Terrence Chen, Haibin Ling, and Ziyan Wu. Multi-motion and appearance self-supervised moving object detection. In *2022 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 2101–2110, 2022. 3

[33] Yanchao Yang, Antonio Loquercio, Davide Scaramuzza, and Stefano Soatto. Unsupervised moving object detection via contextual information separation. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 879–888, 2019. 3

[34] Chengxi Ye, Anton Mitrokhin, Cornelia Fermüller, James A. Yorke, and Yiannis Aloimonos. Unsupervised learning of dense optical flow, depth and egomotion with event-based sensors. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 5831–5838, 2020. 4

[35] Vickie Ye, Zhengqi Li, Richard Tucker, Angjoo Kanazawa, and Noah Snavely. Deformable sprites for unsupervised video decomposition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2657–2666, 2022. 3

[36] Fisher Yu, Wenqi Xian, Yingying Chen, Fangchen Liu, Mike Liao, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving video database with scalable annotation tooling. *ArXiv*, abs/1805.04687, 2018. 2

[37] Luca Zappella, Xavier Llado, Edoardo Provenzi, and Joaquim Salvi. Enhanced local subspace affinity for feature-based motion segmentation. *Pattern Recognition*, 44:454–470, 2011. 2

[38] Bo Zhang, Jinli Suo, and Qionghai Dai. Event-enhanced snapshot compressive videography at 10k fps. *IEEE Trans. Pattern Anal. Mach. Intell.*, 47(2):1266–1278, 2025. 2

[39] Yun Zhang, Bin Luo, and Liangpei Zhang. Permutation preference based alternate sampling and clustering for motion segmentation. *IEEE Signal Processing Letters*, 25(3):432–436, 2018. 2

[40] Yi Zhou, Guillermo Gallego, Xiuyuan Lu, Siqi Liu, and Shaojie Shen. Event-based motion segmentation with spatio-temporal graph cuts. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–13, 2021. 2, 3, 6, 7, 8

[41] Alex Zihao Zhu, Dinesh Thakur, Tolga Özaslan, Bernd Pfrommer, Vijay Kumar, and Kostas Daniilidis. The multi-vehicle stereo event camera dataset: An event camera dataset for 3d perception. *IEEE Robotics and Automation Letters*, 3 (3):2032–2039, 2018. 2

[42] Alex Zihao Zhu, Liangzhe Yuan, Kenneth Chaney, and Kostas Daniilidis. Ev-flownet: Self-supervised optical flow estimation for event-based cameras. *ArXiv*, abs/1802.06898, 2018. 4

[43] Alex Zihao Zhu, Liangzhe Yuan, Kenneth Chaney, and Kostas Daniilidis. Unsupervised event-based learning of optical flow, depth, and egomotion. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 989–997, 2019. 2, 4, 5