EV-FlowNet: Self-Supervised Optical Flow Estimation for Event-based Cameras

Alex Zihao Zhu, Liangzhe Yuan, Kenneth Chaney and Kostas Daniilidis
School of Engineering and Applied Science
University of Pennsylvania
{alexzhu, lzyuan, chaneyk, kostas}@seas.upenn.edu

***Abstract*—Event-based cameras have shown great promise in a variety of situations where frame based cameras suffer, such as high speed motions and high dynamic range scenes. However, developing algorithms for event measurements requires a new class of hand crafted algorithms. Deep learning has shown great success in providing model free solutions to many problems in the vision community, but existing networks have been developed with frame based images in mind, and there does not exist the wealth of labeled data for events as there does for images for supervised training. To these points, we present EV-FlowNet, a novel self-supervised deep learning pipeline for optical flow estimation for event based cameras. In particular, we introduce an image based representation of a given event stream, which is fed into a self-supervised neural network as the sole input. The corresponding grayscale images captured from the same camera at the same time as the events are then used as a supervisory signal to provide a loss function at training time, given the estimated flow from the network. We show that the resulting network is able to accurately predict optical flow from events only in a variety of different scenes, with performance competitive to image based networks. This method not only allows for accurate estimation of dense optical flow, but also provides a framework for the transfer of other self-supervised methods to the event-based domain.**

I. Introduction

By registering changes in log intensity in the image with microsecond accuracy, event-based cameras offer promising advantages over frame based cameras in situations with factors such as high speed motions and difficult lighting. One interesting application of these cameras is the estimation of optical flow. By directly measuring the precise time at which each pixel changes, the event stream directly encodes fine grain motion information, which researchers have taken advantage of in order to perform optical flow estimation. For example, Benosman et al. [5] show that optical flow can be estimated from a local window around each event in a linear fashion, by estimating a plane in the spatio-temporal domain. This is significantly simpler than image-based methods, where optical flow is performed using iterative methods. However, analysis in Rueckauer and Delbruck [26] has shown that these algorithms require significant, hand crafted outlier rejection schemes, as they do not properly model the output of the sensor.

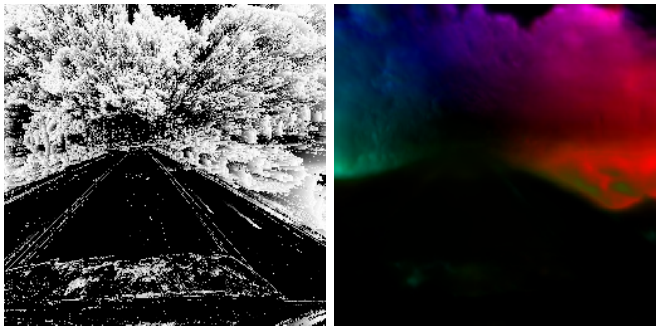

Figure 1: Left: Event input to the network visualizing the last two channels (latest timestamps). Right: Predicted flow, colored by direction. Best viewed in color.

For traditional image-based methods, deep learning has helped the computer vision community achieve new levels of performance while avoiding having to explicitly model the entire problem. However, these techniques have yet to see the same level of adoption and success for event-based cameras. One reason for this is the asynchronous output of the event-based camera, which does not easily fit into the synchronous, frame-based inputs expected by image-based paradigms. Another reason is the lack of labeled training data necessary for supervised training methods. In this work, we propose two main contributions to resolve these issues.

First, we propose a novel image-based representation of an event stream, which fits into any standard image-based neural network architecture. The event stream is summarized by an image with channels representing the number of events and the latest timestamp at each polarity at each pixel. This compact representation preserves the spatial relationships between events, while maintaining the most recent temporal information at each pixel and providing a fixed number of channels for any event stream.

Second, we present a self-supervised learning method for optical flow estimation given only a set of events and the corresponding grayscale images generated from the same camera. The self-supervised loss is modeled after frame based self-supervised flow networks such as Yu et al. [28] and Meister et al. [16], where a photometric loss is used as a supervisory signal in place of direct supervision. As a result,

Associated dataset: https://daniilidis-group.github.io/mvsec/.
Supplementary video: https://youtu.be/eMHZBSoq0sE.

the network can be trained using only data captured directly from an event camera that also generates frame based images, such as the Dynamic and Active-pixel Vision (DAVIS) Sensor developed by Brandli et al. [8], circumventing the need for expensive labeling of data.

These event images combined with the self-supervised loss are sufficient for the network to learn to predict accurate optical flow from events alone. For evaluation, we generate a new event camera optical flow dataset, using the ground truth depths and poses in the Multi Vehicle Event Camera Dataset by Zhu et al. [30]. We show that our method is competitive on this dataset with UnFlow by Meister et al. [16], an image-based self supervised network trained on KITTI, and fine tuned on event camera frames, as well as standard non-learning based optical flow methods.

In summary, our main contributions in this work are:

- We introduce a novel method for learning optical flow using events as inputs only, without any supervision from ground-truth flow.
- Our CNN architecture uses a self-supervised photoconsistency loss from low resolution intensity images used in training only.
- We present a novel event-based optical flow dataset with ground truth optical flow, on which we evaluate our method against a state of the art frame based method.

II. Related Work

A. Event-based Optical Flow

There have been several works that attempt to take advantage of the high temporal resolution of the event camera to estimate accurate optical flow. Benosman et al. [5] model a given patch moving in the spatial temporal domain as a plane, and estimate optical flow as the slope of this plane. This work is extended in Benosman et al. [6] by adding an iterative outlier rejection scheme to remove events significantly far from the plane, and in Barranco et al. [3] by combining the estimated flow with flow from traditional images. Brosch et al. [9] present an analogy of Lucas et al. [15] using the events to approximate the spatial image gradient, while Orchard and Etienne-Cummings [21] use a spiking neural network to estimate flow, and Liu and Delbruck [14] estimate sparse flow using an adaptive block matching algorithm. In other works, Bardow et al. [2] present the optical flow estimation problem jointly with image reconstruction, and solve the joint problem using convex optimization methods, while Zhu et al. [29] present an expectation-maximization based approach to estimate flow in a local patch. A number of these methods have been evaluated in Rueckauer and Delbruck [26] against relatively simple scenes with limited translation and rotation, with limited results, with ground truth optical flow estimated from a gyroscope. Similarly, Barranco et al. [4] provide a dataset with optical flow generated from a known motion combined with depths from a RGB-D sensor.

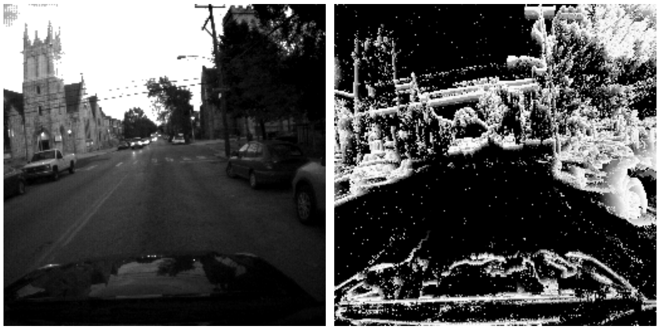

Figure 2: Example of a timestamp image. Left: Grayscale output. Right: Timestamp image, where each pixel represents the timestamp of the most recent event. Brighter is more recent.

B. Event-based Deep Learning

One of the main challenges for supervised learning for events is the lack of labeled data. As a result, many of the early works on learning with event-based data, such as Ghosh et al. [10] and Moeys et al. [17], rely on small, hand collected datasets.

To address this, recent works have attempted to collect new datasets of event camera data. Mueggler et al. [18], provide handheld sequences with ground truth camera pose, which Nguyen et al. [20] use to train a LSTM network to predict camera pose. In addition, Zhu et al. [30] provide flying, driving and handheld sequences with ground truth camera pose and depth maps, and Binas et al. [7] provide long driving sequences with ground truth measurements from the vehicle such as steering angle and GPS position.

Another approach has been to generate event based equivalents of existing image based datasets by recording images from these datasets from an event based camera (Orchard et al. [22], Hu et al. [11]).

Recently, there have also been implementations of neural networks on spiking neuromorphic processors, such as in Amir et al. [1], where a network is adapted to the TrueNorth chip to perform gesture recognition.

C. Self-supervised Optical Flow

Self-supervised, or unsupervised, methods have shown great promise in training networks to solve many challenging 3D perception problems. Yu et al. [28] and Ren et al. [24] train an optical flow prediction network using the traditional brightness constancy and smoothness constraints developed in optimization based methods such as the Lucas Kanade method Lucas et al. [15]. Zhu et al. [31] combine this self-supervised loss with supervision from an optimization based flow estimate as a proxy for ground truth supervision, while Meister et al. [16] extend the loss with occlusion masks and a second order smoothness term, and Lai et al. [13] introduce an adversarial loss on top of the photometric error.

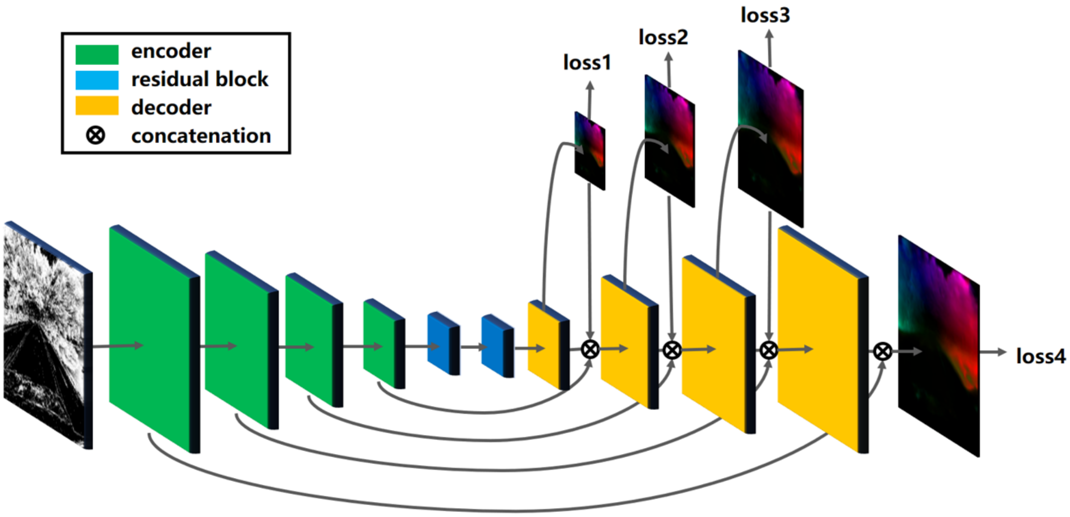

Figure 3: EV-FlowNet architecture. The event input is downsampled through four encoder (strided convolution) layers, before being passed through two residual block layers. The activations are then passed through four decoder (upsample convolution) layers, with skip connections to the corresponding encoder layer. In addition, each set of decoder activations is passed through another depthwise convolution layer to generate a flow prediction at its resolution. A loss is applied to this flow prediction, and the prediction is also concatenated to the decoder activations. Best viewed in color.

III. Method

In this section, we describe our approach in detail. In Sec. III-A, we describe our event representation, which is an analogy to an event image. In Sec. III-B, we describe the self-supervised loss used to provide a supervisory signal using only the gray scale images captured before and after each time window, and in Sec. III-C, we describe the architecture of our network, which takes as input the event image and outputs a pixel-wise optical flow. Note that, throughout this paper, we refer to optical flow as the displacement of each pixel within a given time window.

A. Event Representation

An event-based camera tracks changes in the log intensity of an image, and returns an event whenever the log intensity changes over a set threshold θ:

$$\log(I_{t+1}) - \log(I_t) \geq \theta \tag{1}$$

Each event contains the pixel location of the change, timestamp of the event and polarity:

$$e = \{\mathbf{x}, \quad t, \quad p\} \tag{2}$$

Because of the asynchronous nature of the events, it is not immediately clear what representation of the events should be used in the standard convolutional neural network architecture. Most modern network architectures expect image-like inputs, with a fixed, relatively low, number of channels (recurrent networks excluded) and spatial correlations between neighboring pixels. Therefore, a good representation is key to fully take advantage of existing networks while summarizing the necessary information from the event stream.

Perhaps the most complete representation that preserves all of the information in each event would be to represent the events as a $n \times 4$ matrix, where each column contains the information of a single event. However, this does not directly encode the spatial relationships between events that is typically exploited by convolutions over images.

In this work, we chose to instead use a representation of the events in image form. The input to the network is a 4 channel image with the same resolution as the camera.

The first two channels encode the number of positive and negative events that have occurred at each pixel, respectively. This counting of events is a common method for visualizing the event stream, and has been shown in Nguyen et al. [20] to be informative in a learning based framework to regress 6dof pose.

However, the number of events alone discards valuable information in the timestamps that encode information about the motion in the image. Incorporating timestamps in image form is a challenging task. One possible solution would be to have k channels, where k is the most events in any pixel in the image, and stack all incoming timestamps. However, this would result in a large increase in the dimensionality of the input. Instead, we encode the pixels in the last two channels as the timestamp of the most recent positive and negative event at that pixel, respectively. This is similar to the ”Event-based Time Surfaces” used in Lagorce et al. [12] and the ”timestamp images” used in Park et al. [23]. An example of this kind of image can be found in Fig. 2, where we can see that the flow is evident by following the gradient in the image, particularly for closer (faster moving) objects. While this representation inherently discards all of the timestamps but the most recent at each pixel, we have observed that this representation is sufficient for the network to estimate the correct flow in most regions. One deficiency of this representation is that areas with very dense events and large motion will have all pixels overridden by very recent events with very similar timestamps. However, this problem can be avoided by choosing smaller time windows, thereby reducing the magnitude of the motion.

In addition, we normalize the timestamp images by the size of the time window for the image, so that the maximum value in the last two channels is 1. This has the effect of both scaling the timestamps to be on the same order of magnitude as the event counts, and ensuring that fast motions with a small time window and slow motions with a large time window that generate similar displacements have similar inputs to the network.

B. Self-Supervised Loss

Due to the fact that there is a relatively small amount of labeled data for event based cameras as compared to traditional cameras, it is difficult to generate a sufficient dataset for a supervised learning method. Instead, we utilize the fact that the DAVIS camera generates synchronized events and grayscale images to perform self-supervised learning using the grayscale images in the loss. At training time, the network is provided with the event timestamp images, as well as a pair of grayscale images, occurring immediately before and after the event time window. Only the event timestamp images are passed into the network, which predicts a per pixel flow. The grayscale images are then used to apply a loss over the predicted flow in a self-supervised manner.

The overall loss function used follows traditional variational methods for estimating optical flow, and consists of a photometric and a smoothness loss.

To compute the photometric loss, the flow is used to warp the second image to the first image using bilinear sampling, as described in Yu et al. [28]. The photometric loss, then, aims to minimize the difference in intensity between the warped second image and the first image:

$$\ell_{\text{photometric}}(u, v; I_t, I_{t+1}) = \sum_{x,y} \rho(I_t(x, y) - I_{t+1}(x + u(x, y), y + v(x, y))) \tag{3}$$

where ρ is the Charbonnier loss function, a common loss in the optical flow literature used for outlier rejection (Sun et al. [27]):

$$\rho(x) = (x^2 + \epsilon^2)^\alpha \tag{4}$$

As we are using frame based images for supervision, this method is susceptible to image-based issues such as the aperture problem. Thus, we follow the other works in the frame based domain, and apply a regularizer in the form of a smoothness loss. The smoothness loss aims to regularize the output flow by minimizing the difference in flow between neighboring pixels horizontally, vertically and diagonally.

$$\ell_{\text{smoothness}}(u, v) = \sum_{x,y} \sum_{i,j \in \mathcal{N}(x,y)} \rho(u(x, y) - u(i, j)) + \rho(v(x, y) - v(i, j)) \tag{5}$$

where $\mathcal{N}$ is the set of neighbors around (x, y).

The total loss is the weighted sum of the photometric and smoothness losses:

$$L_{\text{total}} = \ell_{\text{photometric}} + \lambda \ell_{\text{smoothness}} \tag{6}$$

C. Network Architecture

The EV-FlowNet architecture very closely resembles the encoder-decoder networks such as the stacked hourglass (Newell et al. [19]) and the U-Net (Ronneberger et al. [25]), and is illustrated in Fig. 3. The input event image is passed through 4 strided convolution layers, with output channels doubling each time. The resulting activations are passed through 2 residual blocks, and then four upsample convolution layers, where the activations are upsampled using nearest neighbor resampling and then convolved, to obtain a final flow estimate. At each upsample convolution layer, there is also a skip connection from the corresponding strided convolution layer, as well as another convolution layer to produce an intermediate, lower resolution, flow estimate, which is concatenated with the activations from the upsample convolution. The loss in (6) is then applied to each intermediate flow by downsampling the grayscale images. The tanh function is used as the activation function for all of the flow predictions.

IV. Optical Flow Dataset

For ground truth evaluation only, we generated a novel dataset for ground truth optical flow using the data provided in the Multi-Vehicle Stereo Event Camera dataset (MVSEC) by Zhu et al. [30]. The dataset contains stereo event camera data in a number of flying, driving and handheld scenes. In addition, the dataset provides ground truth poses and depths maps for each event camera, which we have used to generate reference ground truth optical flow.

From the pose (consisting of rotation R and translation $\mathbf{p}$) of the camera at time t_0 and t_1, we make a linear velocity assumption, and estimate velocity and angular velocity using numerical differentiation:

$$\mathbf{v} = \frac{(\mathbf{p}(t_1) - \mathbf{p}(t_0))}{dt} \tag{7}$$

$$\omega^\wedge = \frac{\operatorname{logm}\left(R_{t_0}^T R_{t_1}\right)}{dt} \tag{8}$$

where logm is the matrix logarithm, and $\omega^\wedge$ converts the vector ω into the corresponding skew symmetric matrix:

$$\omega^\wedge = \begin{bmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{bmatrix} \tag{9}$$

A central moving average filter is applied to the estimated velocities to reduce noise. We then use these velocities to estimate the motion field, given the ground truth depths, Z, at each undistorted pixel position:

$$\begin{pmatrix}\dot{x}\\ \dot{y}\end{pmatrix} = \begin{bmatrix} -\frac{1}{Z} & 0 & -\frac{x}{Z} & xy & -(1+x^2) & y \\ 0 & -\frac{1}{Z} & \frac{y}{Z} & 1+y^2 & -xy & -x \end{bmatrix} \begin{pmatrix}\mathbf{v}\\ \omega\end{pmatrix} \tag{10}$$

Finally, we scale the motion field by the time window between each pair of images dt, and use the resulting displacement as an approximation to the true optical flow for each pixel. To apply the ground truth to the distorted images, we shift the undistorted pixels by the flow, and apply distortion to the shifted pixels. The distorted flow is, then, the displacement from the original distorted position to the shifted distorted position.

In total, we have generated ground truth optical flow for the indoor_flying, outdoor_day and outdoor_night sequences. In addition to using the indoor_flying and outdoor_day ground truth sets for evaluation, we will also release all sequences as a dataset.

V. Empirical Evaluation

A. Training Details

Two networks were trained on the two outdoor_day sequences from MVSEC. outdoor_day1 contains roughly 12000 images, and outdoor_day2 contains roughly 26000 images. The images are captured from driving in an industrial complex and public roads, respectively, where the two scenes are visually very different. The motions include mostly straights and turns, with occasional independently moving objects such as other cars and pedestrians. The input images are cropped to 256x256, the number of output channels at the first encoder layer is 64 and the number of output channels in each residual block is 512.

To increase the variation in the magnitude of the optical flow seen at training, we randomly select images up to k images apart in time, and all of the events that occurred between those images. In our experiments, $k \in [2, 4, 6, 8, 10, 12]$. In addition, we randomly flip the inputs horizontally, and randomly crop them to achieve the desired resolution.

The weight on the smoothness loss (6), λ, is set to 0.5. Each of the intermediate losses is weighted equally in the final loss. For the Charbonnier loss (4), α was set to be 0.45 and ϵ was set to be 1e-3. The Adam optimizer is used, with learning rate initialized at 1e-5, and exponentially decayed every 4 epochs by 0.8. The model is trained for 300,000 iterations, and takes around 12 hours to train on a 16GB NVIDIA Tesla V100.

B. Ablation Studies

In addition to the described architecture (denoted EV-FlowNet$_{\mathrm{2R}}$), we also train three other networks to test the effects of varying the input to the network, as well as increasing the capacity of the network.

To test the contribution of each of the channels in the input, we train two additional networks, one with only the event counts (first two channels) as input (denoted EV-FlowNet$_{\mathrm{C}}$), and one with only the event timestamps (last two channels) as input (denoted EV-FlowNet$_{\mathrm{R}}$).

In addition, we tested different network capacities by training a larger model with 4 residual blocks (denoted EV-FlowNet$_{\mathrm{4R}}$). A single forward pass takes, on average, 40ms for the smaller network, and 48ms for the larger network, when run on a NVIDIA GeForce GTX 1050, a laptop grade GPU.

C. Comparisons

To compare our results with other existing methods, we tested implementations of Event-based Visual Flow by Benosman et al. [6], an optimization based method that works on events, and UnFlow by Meister et al. [16], a self supervised method that works on traditional frames.

As there is no open source code by the authors of Event-based Visual Flow, we designed an implementation around the method described in Rueckauer and Delbruck [26]. In particular, we implemented the robust Local Plane Fit algorithm, with a spatial window of 5x5 pixels, vanishing gradient threshold th3 of 1e-3, and outlier distance threshold of 1e-2. However, we were unable to achieve any reasonable results on the datasets, with only very few points returning valid flow values ($< 5\%$), and none of the valid flow values being visually correct. For validation, we also tested the open source MATLAB code provided by the authors of Mueggler et al. [18], where we received similar results. As a result, we believe that the method was unable to generalize to the natural scenes in the test set, and so did not include the results in this paper.

For UnFlow, we used the unsupervised model trained on KITTI raw, and fine tuned on outdoor_day2. This model was able to produce reasonable results on the testing sets, and we include the results in the quantitative evaluation in Tab. I.

D. Test Sequences

For comparison against UnFlow, we evaluated 800 frames from the outdoor_day1 sequence as well as sequences 1 to 3 from indoor_flying. For the event input, we used all of the events that occurred in between the two input frames.

The outdoor_day1 sequence spans between 222.4s and 240.4s. This section was chosen as the grayscale images were consistently bright, and there is minimal shaking of the camera (the provided poses are smoothed and do not capture shaking of the camera if the vehicle hits a bump in the road). In order to avoid conflicts between training and testing data, a model trained only using data from outdoor_day2 was used, which is visually significantly different from outdoor_day1.

The three indoor_flying sequences total roughly 240s, and feature a significantly different indoor scene, containing vertical and backward motions, which were previously unseen in the driving scenes. A model trained on both outdoor_day1 and outdoor_day2 data was used for evaluation on these sequences. We avoided fine tuning on the flying sequences, as the sequences are in one room, and all relatively similar in visual appearance. As a result, it would be very easy for a network to overfit the environment. Sequence 4 was omitted as the majority of the view was just the floor, and so had a relatively small amount of useful data for evaluation.

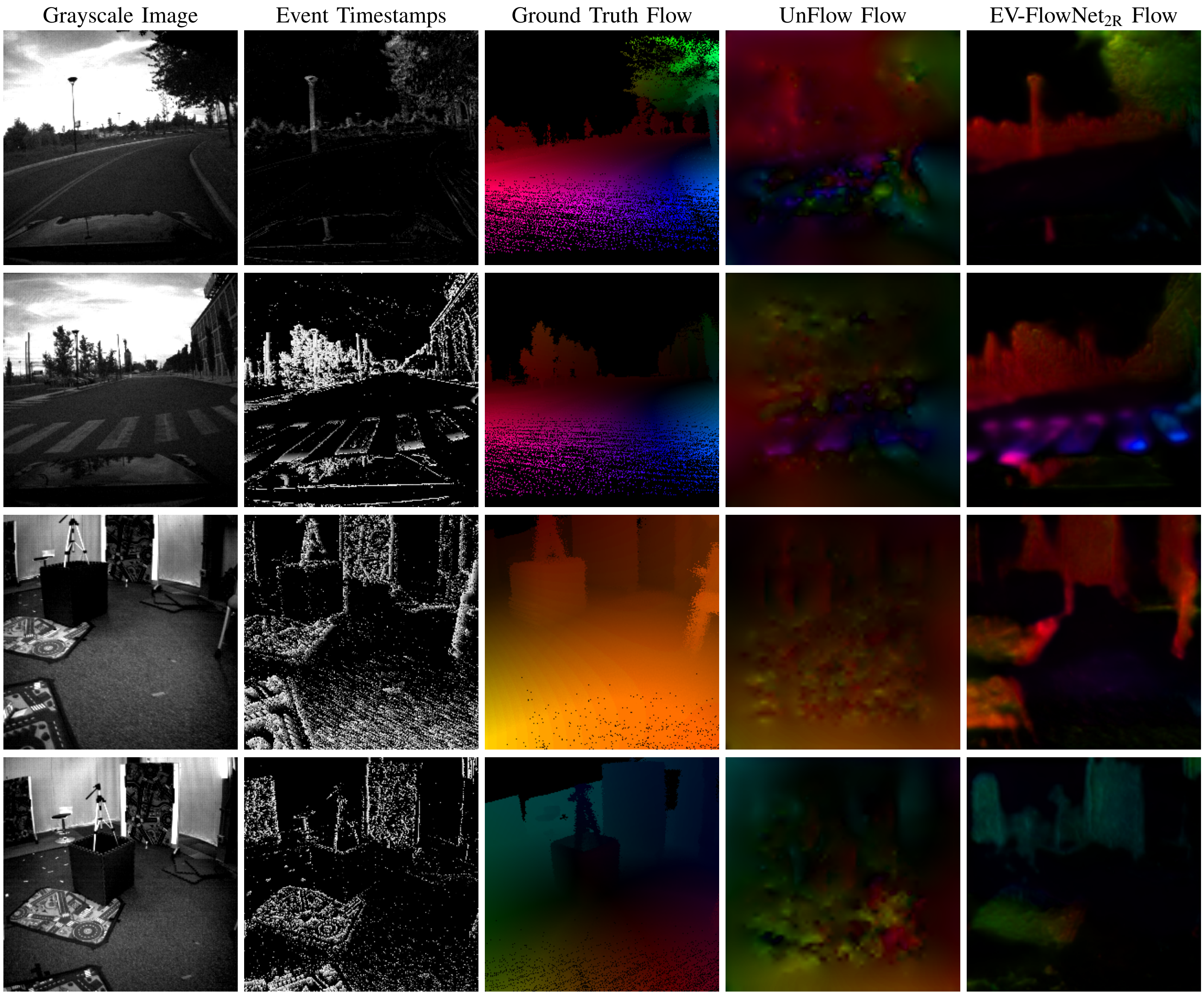

Figure 4: Qualitative results from evaluation. Examples were collected from outdoor_day1, outdoor_day1, indoor_flying1 and indoor_flying2, in that order. Best viewed in color.

E. Metrics

For each method and sequence, we compute the average endpoint error (AEE), defined as as the distance between the endpoints of the predicted and ground truth flow vectors:

$$AEE = \sum_{x,y} \left\| \begin{pmatrix} u(x,y)_{\text{pred}} \\ v(x,y)_{\text{pred}} \end{pmatrix} - \begin{pmatrix} u(x,y)_{\text{gt}} \\ v(x,y)_{\text{gt}} \end{pmatrix} \right\|_2 \tag{11}$$

In addition, we follow the KITTI flow 2015 benchmark and report the percentage of points with EE greater than 3 pixels and 5% of the magnitude of the flow vector. Similarly to KITTI, 3 pixels is roughly the maximum error observed when warping the grayscale images according to the ground truth flow, and comparing against the next image.

However, as the input event image is relatively sparse, the network only returns accurate flow on points with events. As a result, we limit the computation of AEE to pixels in which at least one event was observed. For consistency, this is done with a mask applied to the EE for both event-based and frame-based methods. We also mask out any points for which we have no ground truth flow (i.e. regions with no ground truth depth). In practice, this results in the error being computed over 20-30% of the pixels in each image.

In order to vary the magnitude of flow observed for each test, we run two evaluations per sequence: one with input frames and corresponding events that are one frame apart, and one with frames and events four frames apart. We outline the results in Tab. I.

dt=1 frame	outdoor driving		indoor flying1		indoor flying2		indoor flying3	
	AEE	% Outlier	AEE	% Outlier	AEE	% Outlier	AEE	% Outlier
UnFlow	0.97	1.6	**0.50**	**0.1**	**0.70**	**1.0**	**0.55**	**0.0**
EV-FlowNet$_C$	**0.49**	**0.2**	1.30	6.8	2.34	25.9	2.06	22.2
EV-FlowNet$_T$	0.52	**0.2**	1.20	4.5	2.15	22.6	1.91	19.8
EV-FlowNet$_{2R}$	**0.49**	**0.2**	1.03	2.2	1.72	15.1	1.53	11.9
EV-FlowNet$_{4R}$	**0.49**	**0.2**	1.14	3.5	2.10	21.0	1.85	18.8
dt=4 frames	outdoor driving		indoor flying1		indoor flying2		indoor flying3	
	AEE	% Outlier	AEE	% Outlier	AEE	% Outlier	AEE	% Outlier
UnFlow	2.95	40.0	3.81	56.1	6.22	79.5	**1.96**	**18.2**
EV-FlowNet$_C$	1.41	10.8	3.22	41.4	5.30	60.1	4.68	57.0
EV-FlowNet$_T$	1.34	8.4	2.53	33.7	4.40	51.9	3.91	47.1
EV-FlowNet$_{2R}$	**1.23**	**7.3**	**2.25**	**24.7**	**4.05**	**45.3**	3.45	39.7
EV-FlowNet$_{4R}$	1.33	9.4	2.75	33.5	4.82	53.3	4.30	47.8

Table I: Quantitative evaluation of each model on the MVSEC optical flow ground truth. Average end-point error (AEE) and percentage of pixels with EE above 3 and 5% of the magnitude of the flow vector(% Outlier) are presented for each method (lower is better for both), with evaluation run with image pairs 1 frame apart (top) and 4 frames apart (bottom). The EV-FlowNet methods are: Counts only (EV-FlowNet$_c$), Timestamps only (EV-FlowNet$_T$), 2 Residual blocks (EV-FlowNet$_{2R}$) and 4 Residual blocks (EV-FlowNet$_{4R}$).

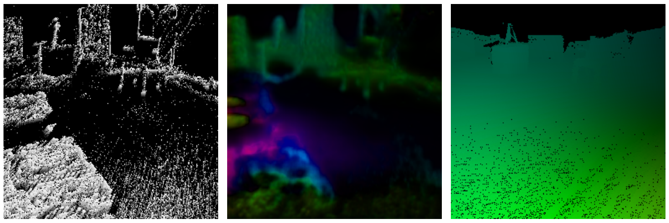

Figure 5: Common failure case, where fast motion causes recent timestamps to overwrite older pixels nearby, resulting in incorrect predictions. Best viewed in color.

F. Results

1) Qualitative Results: In addition to the quantitative analysis provided, we provide qualitative results in Fig. 4. In these results, and throughout the test set, the predicted flow always closely follows the ground truth. As the event input is quite sparse, our network tends to predict zero flow in areas without events. This is consistent with the photometric loss, as areas without events are typically low texture areas, where there is little change in intensity within each pixel neighborhood. In practice, the useful flow can be extracted by only using flow predictions at points with events. On the other hand, while UnFlow typically performs reasonably on the high texture regions, the results on low texture regions are very noisy.

2) Ablation Study Results: From the results of the ablation studies in Tab. I, EV-FlowNet$_C$ (counts only) performed the worst. This aligns with our intuition, as the only information attainable from the counts is from motion blur effects, which is a weak signal on its own. EV-FlowNet$_T$ (timestamps only) performs better for most tests, as the timestamps carry information about the ordering between neighboring events, as well as the magnitude of the velocity. However, the timestamp only network fails when there is significant noise in the image, or when fast motion results in more recent timestamps covering all of the older ones. This is illustrated in Fig 5, where even the full network struggles to predict the flow in a region dominated by recent timestamps. Overall, the combined models clearly perform better, likely as the event counts carry information about the importance of each pixel. Pixels with few events are likely to be just noise, while pixels with many events are more likely to carry useful information. Somewhat surprisingly, the larger network, EV-FlowNet$_{4R}$ actually performs worse than the smaller one, EV-FlowNet$_{2R}$. A possible explanation is that the larger capacity network learned to overfit the training sets, and so did not generalize as well to the test sets, which were significantly different. For extra validation, both EV-FlowNet$_{2R}$ and EV-FlowNet$_{4R}$ were trained for an additional 200,000 iterations, with no appreciable improvements. It is likely, however, that, given more data, the larger model would perform better.

3) Comparison Results: From our experiments, we found that the UnFlow network tends to predict roughly correct flows for most inputs, but tends to be very noisy in low texture areas of the image. The sparse nature of the events is a benefit in these regions, as the lack of events there would cause the network to predict no flow, instead of an incorrect output.

In general, EV-FlowNet performed better on the dt=4 tests, while worse on the dt=1 tests (with the exception of outdoor_driving1 and indoor_flying3). We observed that UnFlow typically performed better in situations with very small or very large motion. In these situations, there are either few events as input, or so many events that the image is overriden by recent timestamps. However, this is a problem intrinsic to the testing process, as the time window is defined by the image frame rate. In practice, these problems can be avoided by choosing time windows large enough so that sufficient information is available while avoiding saturating the event image. One possible solution to this would be to have a fixed number of events in the window each time.

VI. Conclusion

In this work, we have presented a novel design for a neural network architecture that is able to accurately predict optical flow from events alone. Due to the method's self-supervised nature, the network can be trained without any manual labeling, simply by recording data from the camera. We show that the predictions generalize beyond hand designed laboratory scenes to natural ones, and that the method is competitive with state of the art frame-based self supervised methods. We hope that this work will provide not only a novel method for flow estimation, but also a paradigm for applying other self-supervised learning methods to event cameras in the future. For future work, we hope to incorporate additional losses that provide supervisory signals from event data alone, to expose the network to scenes that are challenging for traditional frame-based cameras, such as those with high speed motions or challenging lighting.

Acknowledgments

Thanks to Tobi Delbruck and the team at iniLabs for providing and supporting the DAVIS-346b cameras. We also gratefully appreciate support through the following grants: NSF-IIS-1703319, NSF-IIP-1439681 (I/UCRC), ARL RCTA W911NF-10-2-0016, and the DARPA FLA program.

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.

[2] Patrick Bardow, Andrew J Davison, and Stefan Leutenegger. Simultaneous optical flow and intensity estimation from an event camera. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 884–892, 2016.

[3] Francisco Barranco, Cornelia Fermüller, and Yiannis Aloimonos. Contour motion estimation for asynchronous event-driven cameras. *Proceedings of the IEEE*, 102(10): 1537–1556, 2014.

[4] Francisco Barranco, Cornelia Fermuller, Yiannis Aloimonos, and Tobi Delbruck. A dataset for visual navigation with neuromorphic methods. *Frontiers in Neuroscience*, 10:49, 2016.

[5] Ryad Benosman, Sio-Hoi Ieng, Charles Clercq, Chiara Bartolozzi, and Mandyam Srinivasan. Asynchronous frameless event-based optical flow. *Neural Networks*, 27: 32–37, 2012.

[6] Ryad Benosman, Charles Clercq, Xavier Lagorce, Sio-Hoi Ieng, and Chiara Bartolozzi. Event-based visual flow. *IEEE transactions on neural networks and learning systems*, 25(2):407–417, 2014.

[7] Jonathan Binas, Daniel Neil, Shih-Chii Liu, and Tobi Delbrück. DDD17: End-To-End DAVIS Driving Dataset. *CoRR*, abs/1711.01458, 2017.

[8] Christian Brandli, Raphael Berner, Minhao Yang, Shih-Chii Liu, and Tobi Delbruck. A 240× 180 130 db 3 μs latency global shutter spatiotemporal vision sensor. *IEEE Journal of Solid-State Circuits*, 49(10):2333–2341, 2014.

[9] Tobias Brosch, Stephan Tschechne, and Heiko Neumann. On event-based optical flow detection. *Frontiers in neuroscience*, 9:137, 2015.

[10] Rohan Ghosh, Abhishek Mishra, Garrick Orchard, and Nitish V Thakor. Real-time object recognition and orientation estimation using an event-based camera and CNN. In *Biomedical Circuits and Systems Conference (BioCAS), 2014 IEEE*, pages 544–547. IEEE, 2014.

[11] Yuhuang Hu, Hongjie Liu, Michael Pfeiffer, and Tobi Delbruck. DVS benchmark datasets for object tracking, action recognition, and object recognition. *Frontiers in Neuroscience*, 10, 2016.

[12] Xavier Lagorce, Garrick Orchard, Francesco Galluppi, Bertram E Shi, and Ryad B Benosman. HOTS: a hierarchy of event-based time-surfaces for pattern recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(7):1346–1359, 2017.

[13] Wei-Sheng Lai, Jia-Bin Huang, and Ming-Hsuan Yang. Semi-supervised learning for optical flow with generative adversarial networks. In *Advances in Neural Information Processing Systems*, pages 353–363, 2017.

[14] Min Liu and Tobi Delbruck. Abmof: A novel optical flow algorithm for dynamic vision sensors. *arXiv preprint arXiv:1805.03988*, 2018.

[15] Bruce D Lucas, Takeo Kanade, et al. An iterative image registration technique with an application to stereo vision. 1981.

[16] Simon Meister, Junhwa Hur, and Stefan Roth. UnFlow: Unsupervised Learning of Optical Flow with a Bidirectional Census Loss. *arXiv preprint arXiv:1711.07837*, 2017.

[17] Diederik Paul Moeys, Federico Corradi, Emmett Kerr, Philip Vance, Gautham Das, Daniel Neil, Dermot Kerr, and Tobi Delbrück. Steering a predator robot using a mixed frame/event-driven convolutional neural network. In *Event-based Control, Communication, and Signal Processing (EBCCSP), 2016 Second International Conference on*, pages 1–8. IEEE, 2016.

[18] 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.

[19] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *European Conference on Computer Vision*, pages 483–499. Springer, 2016.

[20] Anh Nguyen, Thanh-Toan Do, Darwin G Caldwell, and Nikos G Tsagarakis. Real-Time Pose Estimation for Event Cameras with Stacked Spatial LSTM Networks. *arXiv preprint arXiv:1708.09011*, 2017.

[21] Garrick Orchard and Ralph Etienne-Cummings. Bioinspired visual motion estimation. *Proceedings of the IEEE*, 102(10):1520–1536, 2014.

[22] Garrick Orchard, Ajinkya Jayawant, Gregory K Cohen, and Nitish Thakor. Converting static image datasets to

spiking neuromorphic datasets using saccades. *Frontiers in neuroscience*, 9, 2015.

[23] Paul KJ Park, Baek Hwan Cho, Jin Man Park, Kyoobin Lee, Ha Young Kim, Hyo Ah Kang, Hyun Goo Lee, Jooyeon Woo, Yohan Roh, Won Jo Lee, et al. Performance improvement of deep learning based gesture recognition using spatiotemporal demosaicing technique. In *Image Processing (ICIP), 2016 IEEE International Conference on*, pages 1624–1628. IEEE, 2016.

[24] Zhe Ren, Junchi Yan, Bingbing Ni, Bin Liu, Xiaokang Yang, and Hongyuan Zha. Unsupervised Deep Learning for Optical Flow Estimation. In *AAAI*, pages 1495–1501, 2017.

[25] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 234–241. Springer, 2015.

[26] Bodo Rueckauer and Tobi Delbruck. Evaluation of event-based algorithms for optical flow with ground-truth from inertial measurement sensor. *Frontiers in neuroscience*, 10, 2016.

[27] Deqing Sun, Stefan Roth, and Michael J Black. A quantitative analysis of current practices in optical flow estimation and the principles behind them. *International Journal of Computer Vision*, 106(2):115–137, 2014.

[28] Jason J Yu, Adam W Harley, and Konstantinos G Derpanis. Back to basics: Unsupervised learning of optical flow via brightness constancy and motion smoothness. In *Computer Vision–ECCV 2016 Workshops*, pages 3–10. Springer, 2016.

[29] Alex Zihao Zhu, Nikolay Atanasov, and Kostas Daniilidis. Event-based feature tracking with probabilistic data association. In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 4465–4470. IEEE, 2017.

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

[31] Yi Zhu, Zhenzhong Lan, Shawn Newsam, and Alexander G Hauptmann. Guided optical flow learning. *arXiv preprint arXiv:1702.02295*, 2017.