2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
October 25-29, 2020, Las Vegas, NV, USA (Virtual)

Unsupervised Learning of Dense Optical Flow, Depth and Egomotion with Event-Based Sensors

Chengxi Ye[1*], Anton Mitrokhin[1*], Cornelia Fermüller[1], James A. Yorke[2], Yiannis Aloimonos[1]

***Abstract*— We present an unsupervised learning pipeline for *dense* depth, optical flow and egomotion estimation for autonomous driving applications, using the event-based output of the Dynamic Vision Sensor (DVS) as input. The backbone of our pipeline is a bioinspired encoder-decoder neural network architecture - ECN. To train the pipeline, we introduce a covariance normalization technique which resembles the lateral inhibition mechanism found in animal neural systems.**

Our work is the first monocular pipeline that generates dense depth and optical flow from sparse event data only, and is able to transfer from day to night scenes without any additional training. The network works in self-supervised mode and has just 150k parameters. We evaluate our pipeline on the MVSEC self driving dataset and present results for depth, optical flow and and egomotion estimation. Thanks to the efficient design, we are able to achieve inference rates of 300 FPS on a single Nvidia 1080Ti GPU. Our experiments demonstrate significant improvements upon works that used deep learning on event data, as well as the ability to perform well during both day and night.

Keywords – event-based learning, neuromorphic sensors, DVS, autonomous driving, low-parameter neural networks, structure form motion, unsupervised learning, lateral inhibition

SUPPLEMENTARY MATERIAL

The supplementary video, appendix and other materials will be made available at http://prg.cs.umd.edu/ECN.html.

I. INTRODUCTION

With recent advances in the field of autonomous driving, autonomous platforms are no longer restricted to research laboratories and testing facilities - they are designed to operate in an open world, where reliability and safety are key factors. As a consequence, modern self-driving cars are often fitted with a sophisticated sensor rig, featuring a number of LiDARs, cameras and radars, but even those undoubtedly expensive setups are prone to misperform in difficult conditions - snow, fog, rain or at night.

A. Event-Based Vision

Recently, there has been much progress in imaging sensor technology, offering alternative solutions to scene perception. A neuromorphic imaging device, called Dynamic Vision Sensor (DVS) [14], inspired by the transient pathway of mammalian vision, can offer exciting alternatives for visual motion perception. The DVS does not record image frames, but instead - the changes of lighting occurring independently at every camera pixel. Each of these changes is transmitted asynchronously and is called an *event*.

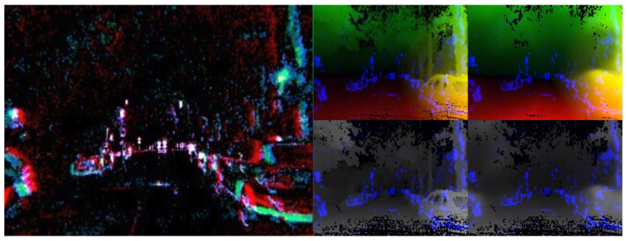


Fig. 1: *Optical flow and depth inference on sparse event data in night scene: event camera output (left), ground truth (middle column), network output (right) (top row - flow, bottom row - depth). The event data is overlaid on the ground truth and inference images in blue. Note, how network is able to 'fill in' the sparse regions and reconstruct the car on the right.*

By its design, this sensor accommodates a large dynamic range, provides high temporal resolution and low latency – ideal properties for applications where high quality motion estimation and tolerance towards challenging lighting conditions are desirable. The DVS offers new opportunities for robust visual perception so much needed in autonomous driving, but challenges associated with the sensor output, such as high noise rates, low spatial resolution and data sparsity ask for different visual processing approaches.

B. Bioinspired Learning

Modern advancements in deep learning make it even more tempting to move away from the traditional feature-based scene reconstruction frameworks. Neural network (NN) based learning approaches [23] have shown promising results on frame-based data in solving video reconstruction problems. However, the design of neural network architectures for event-based sensors is complicated by the challenges associated with representing an asynchronous event data in a form of an image-like input required by most modern networks. Another challenge for learning in networks originates from the spatial sparsity of the event stream, and the resulting difficulty of dense depth or optical flow computation.

In this work we introduce a bioinspired, lightweight encoding-decoding neural network architecture - the Evenly-Cascaded convolutional Network (ECN) to recover dense

[*] The authors contribute equally to this work.

[1]University of Maryland Institute for Advanced Computer Studies, College Park, MD 20742, USA. E-mails: cxy@umd.edu, amitrokh@umd.edu, fer@umiacs.umd.edu, yiannis@cs.umd.edu

[2]Institute for Physical Science and Technology, University of Maryland, College Park, MD 20742, USA. E-mail:yorke@umd.edu

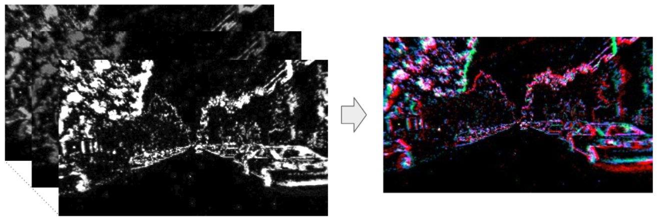


Fig. 2: *A three-channel DVS data representation. The first channel represents the time image [17]. The second and third channels represent the per-pixel positive and negative event counts.*

depth, optical flow and estimate egomotion from sparse event data in a autonomous driving setting. Despite having just 150k parameters, our network is able to generalize well to different types of sequences.

More importantly, the pipeline provides generalization from day to night scenes. Fig. 1 shows an example featuring night driving, where the network was trained on a day scene only. To facilitate reliable night performance, we combine multiple robust time-image representations [17] in a single input, to preserve the 3D structure of the event cloud.

C. Covariance Normalization

As demonstrated later in the text, most existing methods for the training can be suboptimal even for simple linear regression problems. To effectively train neural networks, we introduce a novel, bioinspired covariance normalization similar to the lateral inhibition mechanism found in animal neural cells [2], [10], [11], [22] to remove the correlation in the features and improve the effectiveness of gradient descent training. Our proposed covariance normalization is simple and makes the test-time performance faster compared to traditional normalization approaches. This allows the pipeline to run at more than 300 inferences per second using a single NVIDIA 1080 Ti GPU. We perform ablation studies using the SfMlearner architecture [23] as a baseline and evaluate different normalization techniques (including our novel *covariance normalization*) to show that our model is well suited for event data.

II. Related Work

A. Event-based Depth and Structure from Motion

The majority of event-based depth estimation methods use two or more event cameras [26], [24], [28]. As our proposed approach uses only one event camera, we focus our discussion on monocular depth estimation methods. The first works on event-based monocular depth estimation were presented in [8] and [13]. Rebecq *et al.* [8] used a space-sweep voting mechanism and maximization strategy to estimate semi-dense depth maps where the trajectory is known. Kim *et al.* [13] used probabilistic filters to jointly estimate the motion of the event camera, a 3D map of the scene, and the intensity image. More recently, Gallego *et al.* [6] proposed a unified framework for joint estimation of depth, motion and optical flow. So far there has been no deep learning framework to predict depth from a monocular event camera.

B. Event-based Optical Flow

Previous approaches to image motion estimation used local information in event-space. The different methods often adapt principles known from frame-based vision, namely correlation [4], [16], gradient [3] and local frequency estimation [19]. As discussed in [1], local event information is inherently ambiguous. To resolve the ambiguity the authors proposed to collect events over a longer time intervals and compute the motion from the trace of events at contours.

Recently, neural network approaches have shown promising results in various estimation problems without explicit feature engineering. Most recently, Zhu *et al.* [27] released the MVSEC dataset [25] and proposed self-supervised learning algorithm to estimate optical flow. Unlike [27], which uses grayscale information as a supervision signal, our proposed framework uses only events and thus can work in challenging lighting conditions.

III. Overview of Methods

A. Ego-motion Model

We assume that the camera is moving with rigid motion with translational velocity $v = (v_x, v_y, v_z)^T$ and rotational velocity $\omega = (\omega_x, \omega_y, \omega_z)$, and that the camera intrinsic parameters are provided. Let $\mathbf{X} = (X,Y,Z)^T$ be the world coordinates of a point, and $\mathbf{x} = (x,y)^T$ be the corresponding pixel coordinates in the calibrated camera. Under the assumption of rigid motion, the image velocity $\mathbf{u} = (u,v)^T$ at $(x,y)^T$ is given as:

$$\begin{pmatrix} u \\ v \end{pmatrix} = \frac{1}{Z}\begin{pmatrix} -1 & 0 & x \\ 0 & -1 & y \end{pmatrix}\begin{pmatrix} v_x \\ v_y \\ v_z \end{pmatrix} + \begin{pmatrix} xy & -1-x^2 & y \\ 1+y^2 & -xy & -x \end{pmatrix}\begin{pmatrix} \omega_x \\ \omega_y \\ \omega_z \end{pmatrix} = A\mathbf{p} \quad (1)$$

As such, given the inverse depth and the ego-motion velocities $\mathbf{v}, \omega$, we can calculate the optical flow at a point using a matrix multiplication (Equation 1) Here $\mathbf{p}$ is used to denote the pose vector $(\mathbf{v}, \omega)^T$, and A is a 2×6 matrix. Due to scaling ambiguity in this formulation, depth Z and translation (v_x, v_y, v_z) are computed up to a scaling factor.

B. Event-Based Input

The raw data from the DVS sensor is a stream of events, which we treat as data of 3 dimensions. Each event encodes the pixel coordinate (x,y) and the *timestamp* t. In addition, it also carries information about its *polarity* - a binary value that disambiguates events generated on rising light intensity (positive polarity) and events generated on falling light intensity (negative polarity).

The 3D (x,y,t) event cloud (within a small time slice), called event slice, is projected onto a plane and converted to a 3-channel image. An example of such image can be seen in Fig. 2. Two of the channels are the per-pixel counts of positive and negative events. The third channel is the *time image* as described in [17] - each pixel consists of the average timestamp of the events generated on this pixel, because the averaging of timestamps provides better noise tolerance. More complicated event binning schemes have been used in prior work [28], but we find our slice images faster to

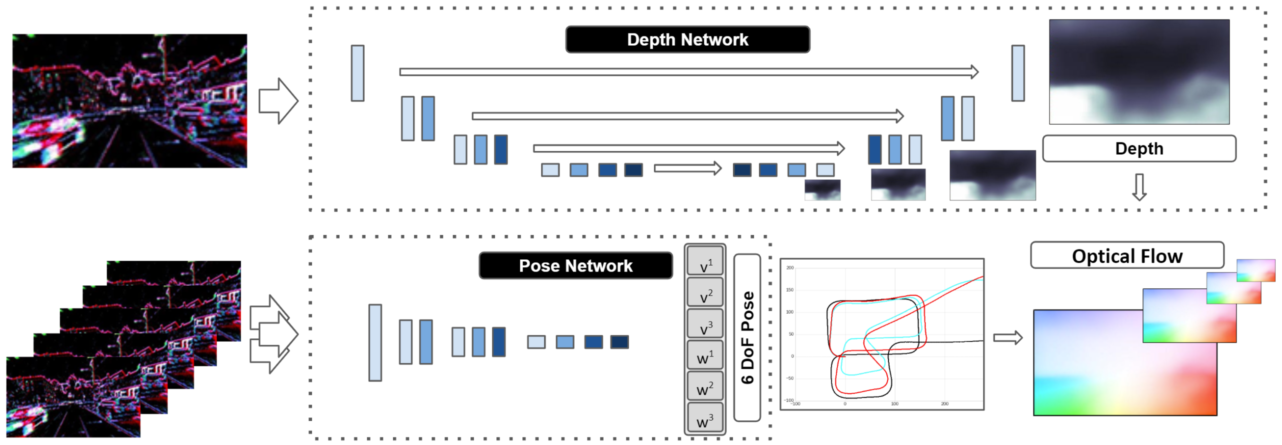


Fig. 3: *The depth network (top) with an encoder-decoder architecture is used to estimate scene depth. The pose network (bottom) takes consecutive frames to estimate the translational velocity and rotational velocity with respect to the middle frame. Given the poses of neighboring frames and the depth of the middle frame, we calculate the optical flow. The neighboring frames are inversely warped to the middle frame and the warping difference provides the supervision loss. In the networks lighter/darker colors represents lower/higher level features.*

compute with the same performance. The neural network input consists of up to 5 such consecutive slice images to better preserve the timestamp information of the event cloud.

C. The Pipeline

Our pipeline is similar to the one proposed in [23]. It consists of a depth prediction component and a parallel pose prediction component, which both feed into a optical flow component to warp successive event slices. The loss from the warping error is backpropagated to train flow, inverse depth, and pose.

The network components are based on the efficient ECN network structure. An (ECN based) encoding-decoding architecture is used to estimate scaled inverse depth $\frac{1}{Z}$ from a single slice of events. To address the data sparsity, we use bilinear interpolation, which propagates local information and fills in the gaps between events. A second network, which takes consecutive slices of signals, is used to derive v and ω. Then, using the rigid motion and inverse depth to predict the optical flow, neighboring slices at neighboring time instances $T+1, T+2$ and $T-1, T-2$ are warped to the slice at T (Fig. 3). The l^1 loss is used to measure the difference between the warped events and the middle slice as

$$Loss_{warp} = \sum_{T-2 \leq n \leq T+2, n \neq T} |I_n^{warpped} - I_T| \qquad (2)$$

It is worth pointing out that the outputs of our networks are multi-scale. The loss functions are weighted by the number of pixels to calculate the total loss.

D. Evenly Cascaded Network Architecture

Our transform of features takes biological inspiration from multi-stage information distillation, and incorporates feedback [21]. In our architecture, the encoding layers split the (layer) input into two streams of features (Fig. 4): one encodes the features from the previous layer at lower resolution (shown with the same color); the other directly generates a set of higher level features, as in CNN architectures. At the end of the encoding stage, the network has a multi-scale feature representation. This representation is used in our pose prediction.

In this work we added to the encoder [21] a decoder, which works as follows: In each decoding layer, we use the higher level features from the previous layer as a feedback signal to improve the lower level features, and combine them with the features from the corresponding encoding layer as in the U-Net [18] architecture.

Our design facilitates training because residual learning is conducted throughout the network for each level of features, while in comparison, the original ResNet [7] does that only in design blocks. In the encoder, we generate higher level features similar to DenseNet [9], but we use residual learning.

To tackle the challenges raised by sparse event data and evenly resize the features, we use bilinear interpolation. In the encoding layers, our network downscales the feature maps by a scaling factor of $(s < 1)$ to get increasingly coarse features. In the decoding layers, the feature maps are upscaled by a factor of $1/s$. Bilinear interpolation propagates the sparse data spatially, facilitating dense prediction of depth and optical flow.

E. Depth Predictions

Initially, both high and low-level coarse features are used to predict a backbone depth map. The depth map is then up-sampled with bilinear interpolation for refinement. Then the enhanced lower level features are used to estimate the prediction residue, which are added to the backbone estimation to refine it.

Our pipeline is monocular and predicts depth up to a scale. In real world driving scenes, the mean depth value stays

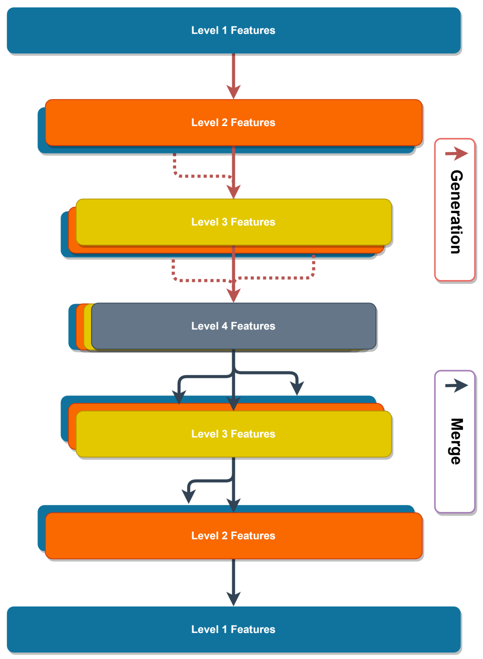


Fig. 4: *The encoder-decoder structure. Only the generation and merging of features are shown.*

relatively stable. Taking advantage of this observation, we use batch normalization before making the depth prediction so the predicted depths have similar range.

To further address the sparsity in data, we utilize a sparsity constraint that promotes edge-preserving information propagation:

$$Loss_{smooth}(I) = \sum_i \sum_{j \in N(i)} |I_j - I_i|^p, 0 < p \leq 1 \quad (3)$$

Here the loss is applied on the first-order derivatives of the depth estimation in a neighborhood $N(i)$. The complete loss of our pipeline is hence defined as:

$$Loss = Loss_{warp} + \lambda Loss_{smooth} \quad (4)$$

F. Covariance Normalization

1) Mathematical Justification: Using gradient descent to solve optimization problems can be challenging for real world data, since the features can be badly scaled and correlated. Even though various normalization strategies have been proposed to train neural networks [12], [20], [15], it is important to point out, scaling the data does not guarantee good conditioning. The covariance matrix of the normalized data matrix X usually remains ill-conditioned, slowing down the gradient descent algorithms.

Proposition 1: Assume we are given a linear regression problem with L_2 loss: $y = Xw$, $Loss = \frac{1}{2}||y - \hat{y}||^2 = \frac{1}{2}||Xw - \hat{y}||^2$. One step of gradient descent leads to the optimal transformation if the features are uncorrelated.

Proof: For an explicit solution we have $\frac{\partial Loss}{\partial w} = X^t(Xw - \hat{y}) = 0$,

$$w = (X^tX)^{-1}X^t\hat{y}. \quad (5)$$

If the input is uncorrelated, we have $w = X^t\hat{y}$. On the other hand, to conduct one iteration of gradient descent: we have $w_{new} = w_{old} - step\_len \times (X^tXw_{old} - X^t\hat{y})$. If $X^tX = I$ then $step\_len = 1$ is optimal and with one iteration we have $w_{new} = X^t\hat{y}$. ■

This above result suggests that correlated data slows down the gradient descent training even in the simplest L_2 optimization problem. Standard normalization methods cannot account for the correlation in the data, and therefore are suboptimal for the gradient descent training. Here we propose the covariance normalization technique to make the gradient descent process more effective, not only reduces the training time but leads to better solutions.

Given the covariance matrix $Cov = (X - \mu)^t(X - \mu)$ between the features, we use Denman-Beavers iteration [5], a coupled Newton-Schulz iteration to calculate the inverse square root. Denman-Beavers iterations start with initial values $Y_0 = Cov$, $Z_0 = I$. The iteration is defined as:

$$\begin{aligned} Y_{k+1} &= \frac{1}{2}Y_k(3I - Z_kY_k) \\ Z_{k+1} &= \frac{1}{2}(3I - Z_kY_k)Z_k \\ Y_k \rightarrow Cov^{\frac{1}{2}}, &\quad Z_k \rightarrow C^{-\frac{1}{2}} \end{aligned} \quad (6)$$

A few iterations(5) lead to significantly faster convergence and better results.

Following this spirit, we insert the covariance normalization operation before the linear/convolution layers to remove the correlation effects between feature channels. In each layer, the transform is:

$$y_i = ReLU \circ W_i \circ D_i \circ x_i, \quad (7)$$

here x_i is the input to the $i-th$ layer, $D_i = Cov^{-0.5}$ is the covariance normalization. It is important to note that this procedure does not change the learning problem, as the standard network training is solving for $W_i \circ D_i$. Another important property is that according to the associate rule of matrix multiplication, $W_i \circ D_i \circ x_i = (W_i \circ D_i) \circ x_i$. Therefore, the normalization matrix D_i can be calculated and merged with the weights in the current layer. In the testing time, we use a running average estimate of D_i and transform W_i to $(W_i \circ D_i)$. As a result, no normalization is needed in testing time. This makes the system faster compared to other normalization techniques.

2) Neurophysiology Analogy: Removing the correlative factors with lateral cells is analogous to the lateral inhibition mechanism [2] found in animal neural systems. The technique can be further generalized to remove the correlation between neighboring pixels [22]. The resulting 'deconvolution' filter has been shown to resemble the center-surround structure [10], [11] in animal visual systems and has recently provided significant improvements to multiple learning tasks such as image classification and semantic segmentation.

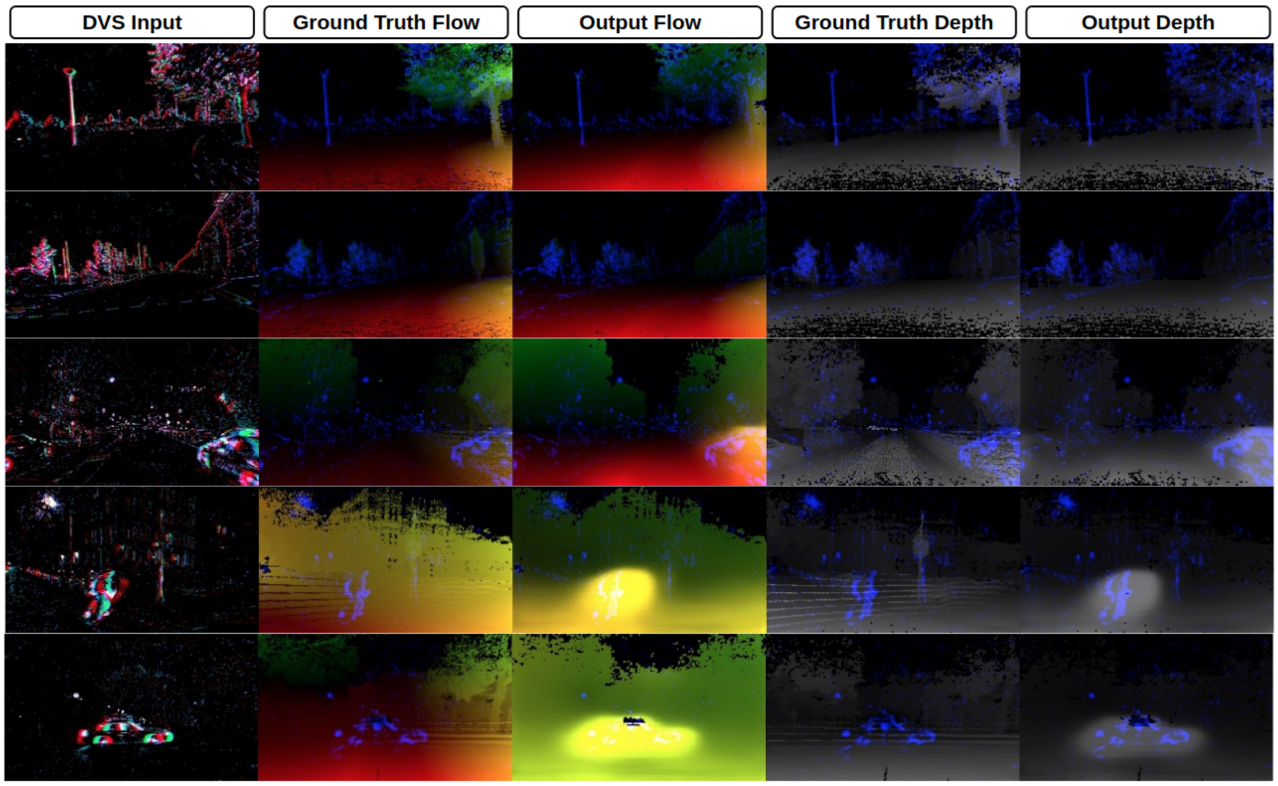


Fig. 5: *Qualitative results from our evaluation. The table entries from left to right: DVS input, ground truth optical flow, network output for flow, ground truth for depth, network output for depth. The event counts are overlaid in blue for better visualization. Examples were collected from sequences of the MVSEC [25] dataset: (top to bottom) outdoor day 1, outdoor day 1, outdoor night 1, outdoor night 2, outdoor night 3. Note that on the 'night' sequences the ground truth is occasionally missing due to Lidar limitations (like in the lowest image) but the pipeline outputs the correct result.*

TABLE I: Evaluation of the optical flow pipeline on MVSEC autonomous driving dataset

	outdoor day 1		outdoor night 1		outdoor night 2		outdoor night 3	
	AEE	% Outlier	AEE	% Outlier	AEE	% Outlier	AEE	% Outlier
ECN	0.35	0.04	**0.49**	**0.82**	**0.43**	**0.79**	**0.48**	**0.80**
ECN_{masked}	**0.30**	0.02	0.53	1.1	0.49	0.98	0.49	1.1
*Zhu*18 **[28]**	0.32	**0.0**	-	-	-	-	-	-
$EV\text{-}FlowNet_{best}$ **[27]**	0.49	0.20	-	-	-	-	-	-
$SfMlearner$	0.58	0.89	0.59	1.01	0.78	1.32	0.59	1.38
ECN_{erate}	0.28	0.02	0.46	0.67	0.40	0.53	0.43	0.67

IV. Experiments

Our our self-supervised learning framework can infer dense optical flow, depth and egomotion from event data. We evaluate our work on the MVSEC [25] event camera dataset which, given a ground truth frequency of 20 Hz, contains over 40000 ground truth images.

The MVSEC dataset, inspired by the KITTI dataset, features 5 sequences of a car on the street (2 during the day and 3 during the night). MVSEC was shot in a variety of lighting conditions and features low-light and high dynamic range frames which are often challenging for an analysis with classical cameras.

A. Implementation Details

Our network architecture has scaling rates of 0.5 and 2.0 respectively for the encoding and decoding layers, which results in 5 encoding/decoding layers. The depth network has 8 initial hidden channels and expands with a growth rate of 8. We halve these settings to 4 for our pose network. The pose network takes 5 consecutive event slices and predicts $6d$ pose (velocity) vectors.

With 3×3 convolutions, the combined network has $150k$ parameters. We train the network with a batch size of 32 and use the Adam optimizer with a learning rate of 0.01. Compared to the standard architecture of the SfMlearner, our architecture allows for higher learning rates, allowing us to train the pipeline faster. The learning rate is annealed using cosine scheduling, and we let the training run for 30 epochs. Our training takes 7-minutes for each epoch using a single Nvidia GTX 1080Ti GPU. We set the smoothness loss weight $\lambda = 0.1$. The version of the network wich uses batch normalization can run at 250 FPS, and the one using covariance normalization runs at 300 FPS.

B. Dataset Preparation

We trained the network using only the *outdoor day 2* sequence with the hood of the car cropped. Our experiments demonstrate that our training generalizes well to the notably different *outdoor day 1* sequence, as well as to the *night* sequences. We would like to note that the *outdoor night* sequences have occasional errors in the ground truth (see for example Fig. 5 last three rows, or Fig. 9). All incorrect frames have been manually removed for the evaluation.

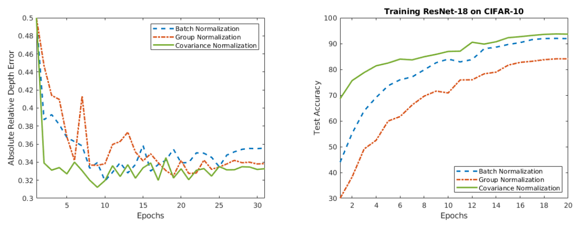
C. Ablation Studies

1) Testing on the SfMlearner: As baseline we use the state-of-the-art *SfMlearner* [23] on our data (event images). *SfMlearner* has a fixed structure of 7 encoding and 7 decoding layers. It has 32 initial hidden channels and expands to 512 channels. Overall, SfMlearner contains 33M parameters - much larger than 150k within ECN. SfMlearner is trained using Adam optimizer with a learning rate of $2e^{-4}$ and a batch size of 4. We replace the inputs with event slices, and we include the evaluation results for flow and egomotion in tables I and III.

2) Normalization Methods: We compare two classical normalization methods and our *covariance normalization method* on the validation set portion of the outdoor day 2 sequence. We apply 5 Denman-Beavers iterations in the normalization procedure. Compared with other normalization methods, covariance normalization leads to more thorough data whitening, and we have noticed this layer-wise whitening lead to faster convergence (Fig. 6). Subsequent experiments shown that on the popular CIFAR-10/100 datasets and a variety of networks, covariance normalization has reduced the required training epochs by $\sim 5\times$ to achieve the same accuracy(Fig. 6(b)).

3) The Impact of Data Sparsity: Since the event data is inherently sparse, we investigate the performance of the pipeline in relation to the data sparsity.

We measure the data sparsity as a percentage of the pixels on the input images with at least one event. Fig. 7 demonstrates how the data sparsity is inversely proportional to the average endpoint error for the optical flow (we have



(a) (b)

Fig. 6: *(a)Comparison of Abs Rel Errors using different normalization methods on outdoor day 1 sequence (less is better). (b) Comparison of different normalization methods on the CIFAR-10 image classification dataset(higher is better, in 20 epochs, ResNet-18 achieves 94% accuracy when trained with covariance normalization).*

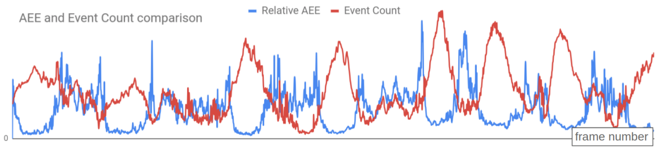


Fig. 7: *The Average Endpoint Error (blue) and the number of pixels with at least one event (red) for the first 1500 frames of 'outdoor˙day1' sequence of the MVSEC [25] dataset. Both plots are normalized so that the mean value is 0.5 for easier comparison.*

observed similar behavior for depth and egomotion). The *outdoor day 1* sequence is used to minimize the influence of the noise.

We find the inverse correlation between sparsity and inference quality to be a useful observation, as this property could be efficiently used in sensor fusion in a robotic system. We provide an additional row to the Table I: ECN_{erate}, and report our error metrics once again only for the frames with *higher than average* number of event pixels across the datasets.

D. Qualitative Results

In addition to the quantitative evaluation, we present a number of samples for qualitative analysis in Fig. 5 - all unseen during the training. The last three rows of the table show the night sequences, when the pipeline has to tackle issues of high noise and low light.

E. Optical Flow Evaluation

The optical flow results are reported in terms of Average Endpoint Error ($AEE = \frac{1}{n}\sum \|\vec{y} - \vec{y}^{*}\|_2$ with y^* and y the estimated and ground truth value, and n the number of pixels for which flow was estimated). We compare our results against two state-of-the-art optical flow methods for event-based cameras: $EV-FlowNet$ [27] and a recent stereo method [28] (in the tables - $Zhu18$).

Because our network produces flow and depth values for every image pixel, our evaluation is not constrained by pixels which did not trigger a DVS event. Still, to be consistent with previous works, we report both numbers for each of our experiments (for example, ECN and ECN_{masked}, where the latter has errors computed only on the pixels with at least one event). Similar to KITTI and EV-FlowNet, we report the percentage of outliers - values with error more than 3 pixels or 5% of the flow vector magnitude.

To compare against [27] and [28], we account for the difference in the frame rates (for example, EV-FlowNet uses the frame rate of the DAVIS classical frames) by scaling our optical flow. Our main results are presented in the Table I.

We show that our optical flow performs well during both day and night, all on unseen sequences. The results are typically better for the experiments with event masks except for the *outdoor night*. A possible explanation is that this sequence is much noisier with events being generated not only on the edges, which leads to suboptimal masking.

F. Depth Evaluation

Since there are currently no monocular event-based methods for the depth estimation based on unsupervised learning,

TABLE II: Evaluation of the depth estimation pipeline. Results on masked, sparse depth are separated by "/", followed by the results on *SfMLearner* in braces.

	outdoor day 1	outdoor night (combined)
Abs Rel	0.29 / 0.33 (0.55)	0.34 / 0.39 (0.53)
RMSE log	0.29 / 0.33 (0.54)	0.38 / 0.42 (0.51)
SILog	0.12 / 0.14 (0.28)	0.15 / 0.18 (0.32)
$\delta < 1.25$	0.80 / 0.97 (0.65)	0.67 / 0.95 (0.56)
$\delta < 1.25^2$	0.91 / 0.98 (0.78)	0.85 / 0.98 (0.75)
$\delta < 1.25^3$	0.96 / 0.99 (0.89)	0.93 / 0.99 (0.87)

we provide the classical scale-invariant depth metrics, used in many works such as [23]:

$$Accuracy: \%of y_i \ s.t. \ max(\frac{y_i}{y_i^*}, \frac{y_i^*}{y_i}) = \delta < th \tag{8}$$

$$SILog: \frac{1}{n}\sum d_i^2 - \frac{1}{n^2}(\sum d_i)^2, d_i = \log y_i - \log y_i^* \tag{9}$$

$$AbsoluteRelativeDifference: \frac{1}{n}\sum \frac{|y-y^*|}{y^*} \tag{10}$$

$$LogarithmicRMSE: \sqrt{\frac{1}{n}\sum \|\log y - \log y^*\|^2} \tag{11}$$

Our results are presented in Table II for both event count-masked depth values and full, dense depth. Since the night driving scenes have similar depth geometries, we aggregate all 3 sequences in a single table entry. We notice that applying an event mask during the evaluation increases accuracy for all scenes - this is expected, as predicting depth information on pixels without any events is inherently more challenging.

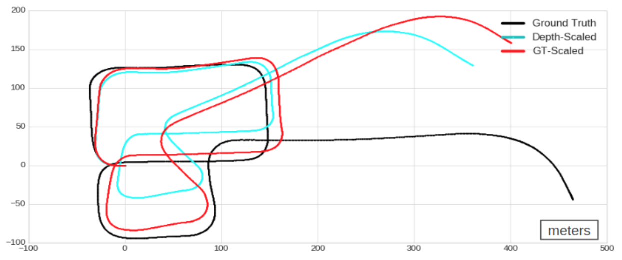
G. Egomotion Estimation

Our pipeline is capable of inferring egomotion on both day and night sequences, and transfers well from *outdoor day 2* onto *outdoor day 1* and *outdoor night 1,2,3*. Since our pipeline is monocular, we predict the translational component of the velocity up to a scaling factor.

For the driving scenarios we *make an important observation* - the mean distance of the road in respect to the camera is often a constant. We crop the lower middle part of the inferred depth image and apply a scaling factor such that the mean depth value (corresponding to the road location) is constant. In our experiments, we report egomotion with translational scales taken both from ground truth (AEE_{tr}^{gt}) and using the depth constancy constraint (AEE_{tr}^{depth}), with a global scale taken from ground truth. The qualitative results are presented in Fig. 8.

Unlike [28], we train *SfMlearner* on the event images, and not on the classical frames to allow for evaluation on the night sequences. We provide comparison to the work in [28], although it uses a stereo setup and reports results only on the *outdoor day 1* sequence.

To be consistent with [28], we report our trajectory estimation relative pose and relative rotation errors as $RPE = arccos(\frac{t_{pred} \cdot t_{gt}}{\|t_{pred}\|_2 \cdot \|t_{gt}\|_2})$ and $RRE = \|logm(R_{pred}^T R_{gt})\|_2$. Here $logm$ is matrix logarithm and R are Euler rotation matrices. To account for translational scale errors, we report classical



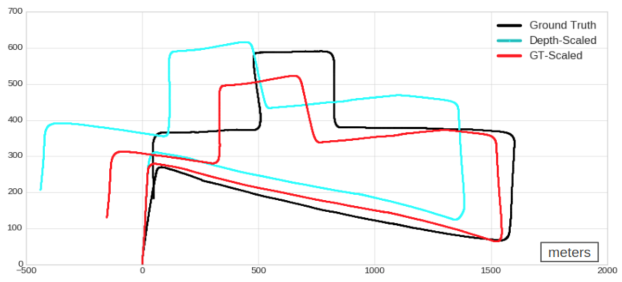


Fig. 8: *Estimated trajectories on 'outdoor day 1' (top) and 'outdoor night 2' (bottom) sequences, acquired by integrating the egomotion predictions. The network was trained only on 'outdoor day 2'. Black: ground truth. Red: network prediction with translational scale applied from ground truth. Cyan: result by assuming the mean depth is fixed throughout the sequence (sec. IV-G) and by applying a single global scaling to the translational pose.*

Endpoint Errors - computed as a magnitude of the differences in translational component of the velocities. Our quantitative results are presented in Table III.

TABLE III: *Egomotion estimation results on driving sequences - 'outdoor day 1' and 'outdoor night 1,2,3'. The ARPE and ARRE are reported in degrees and radians respectively [28], AEE is in m/s. AEE_{tr}^{gt} - translational endpoint error with ground truth normalization. AEE_{tr}^{depth} - normalized using depth prediction and a global scaling factor (see sec. IV-G).*

			ARPE	ARRE	AEE_{tr}^{gt}	AEE_{tr}^{depth}
ECN	outdoor day 1		3.98	0.00267	0.70	1.29
Zhu18 [28]			7.74	0.00867	-	-
$SfMlearner$			16.99	0.00916	1.59	2.39
ECN	outdoor night	1	3.90	0.00139	0.42	1.26
$SfMlearner$			9.95	0.00433	1.04	2.47
ECN		2	3.44	0.00202	0.80	1.34
$SfMlearner$			13.51	0.00499	1.66	3.15
ECN		3	3.28	0.00202	0.49	1.03
$SfMlearner$			1.053	0.00482	1.42	2.74

H. Discussion

A monocular pipeline tends infer more information from the shape of the contours on depth estimation and hence would transfer poorly on completely different scenarios. Nevertheless, we were able to achieve good generalization on night sequences.

We observe a relatively small angular drift on trajectory estimation (Fig. 8). Despite our model predicting a full 6 degree of freedom motion we admit that in the car scenario only 2 motion parameters play a meaningful role and the network may tend to overfit - yet practically, the results are directly applicable for autonomous driving and ground-based robotics in general.

A common limitation of event-based sensors in the lack of data when the relative motion is not present. Fig. 9 shows such an example. This issue can be solved by fusing data from other visual sensors or by introducing jitter to the event-based sensor. Still, for the driving environment the contours of obstacles, people and cars are clearly visible, as can be seen in Fig. 5.

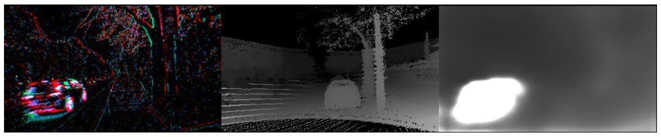


Fig. 9: *A dataset artifact: A non-moving car is not visible on the DAVIS camera (left image) which prevents ECN from inferring optical flow or depth correctly (right image is the inference inverse depth image). The moving car on the left side of the road is clearly visible in the event space and its depth inference is correct, but due to the Lidar limitations the depth ground truth is missing. This frame is taken from the 'outdoor_night 1' MVSEC sequence.*

V. Conclusion

We have presented a lightweight pipeline for generating dense optical flow, depth and egomotion from bioinspired sparse event camera data. We have shown experimentally that our bioinspired neural network architecture using multi-level features improves upon existing work. We introduce a covariance normalization method that resembles the lateral inhibition mechanism in animal neural systems not only to train the network better but make the inference faster.

VI. Acknowledgement

The support of Prophesee SA, Samsung Electronics, the National Science Foundation under grant BCS 1824198, and ONR under grant N00014-17-1-2622 is greatly acknowledged.

References

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

[2] GEORG VON BEKESY. *Sensory Inhibition*. Princeton University Press, 1967.

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

[4] T. Delbruck. Frame-free dynamic digital vision. In *Proceedings of Intl. Symposium on Secure-Life Electronics, Advanced Electronics for Quality Life and Society, Tokyo, Japan,*, pages 21–26, March 2008.

[5] Eugene D. Denman and Alex N. Beavers, Jr. The matrix sign function and computations in systems. *Appl. Math. Comput.*, 2(1):63–94, January 1976.

[6] Guillermo Gallego, Henri Rebecq, and Davide Scaramuzza. A unifying contrast maximization framework for event cameras, with applications to motion, depth, and optical flow estimation. *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2018.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[8] Guillermo Gallego Henri Rebecq and Davide Scaramuzza. Emvs: Event-based multi-view stereo. In E. R. Hancock R. C. Wilson and W. A. P. Smith, editors, *Proceedings of the British Machine Vision Conference (BMVC)*, pages 63.1–63.11, September 2016.

[9] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4700–4708, 2017.

[10] D. H. Hubel and T. N. Wiesel. Integrative action in the cat's lateral geniculate body. *The Journal of Physiology*, 155(2):385–398, 1961.

[11] David H Hubel and Torsten N Wiesel. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology*, 160(1):106–154, 1962.

[12] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167, 2015.

[13] Hanme Kim, Stefan Leutenegger, and Andrew J. Davison. Real-time 3d reconstruction and 6-dof tracking with an event camera. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016*, pages 349–364, Cham, 2016. Springer International Publishing.

[14] P. Lichtsteiner, C. Posch, and T. Delbruck. A 128 x 128 at 120db 15 micros latency asynchronous temporal contrast vision sensor. *Solid-State Circuits, IEEE Journal of*, 43(2):566–576, 2008.

[15] Tsung-Yu Lin and Subhransu Maji. Improved bilinear pooling with cnns. *CoRR*, abs/1707.06772, 2017.

[16] Min Liu and Tobi Delbruck. Block-matching optical flow for dynamic vision sensors: Algorithm and fpga implementation. In *Circuits and Systems (ISCAS), 2017 IEEE International Symposium on*, pages 1–4. IEEE, 2017.

[17] Anton Mitrokhin, Cornelia Fermuller, Chethan Parameshwara, and Yiannis Aloimonos. Event-based moving object detection and tracking. *IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, 2018.

[18] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *MICCAI (3)*, volume 9351 of *Lecture Notes in Computer Science*, pages 234–241. Springer, 2015.

[19] Stephan Tschechne, Tobias Brosch, Roman Sailer, Nora von Egloffstein, Luma Issa Abdul-Kreem, and Heiko Neumann. On event-based motion detection and integration. In *Proceedings of the 8th International Conference on Bioinspired Information and Communications Technologies*, pages 298–305, 2014.

[20] Yuxin Wu and Kaiming He. Group normalization. *CoRR*, abs/1803.08494, 2018.

[21] Chengxi Ye, Chinmaya Devaraj, Michael Maynord, Cornelia Fermüller, and Yiannis Aloimonos. Evenly cascaded convolutional networks. In *IEEE International Conference on Big Data, Big Data 2018, Seattle, WA, USA, December 10-13, 2018*, pages 4640–4647, 2018.

[22] Chengxi Ye, Matthew Evanusa, Hua He, Anton Mitrokhin, Tom Goldstein, James A. Yorke, Cornelia Fermuller, and Yiannis Aloimonos. Network deconvolution. In *International Conference on Learning Representations*, 2020.

[23] Tinghui Zhou, Matthew Brown, Noah Snavely, and David G. Lowe. Unsupervised learning of depth and ego-motion from video. In *CVPR*, 2017.

[24] Yi Zhou, Guillermo Gallego, Henri Rebecq, Laurent Kneip, Hongdong Li, and Davide Scaramuzza. Semi-dense 3d reconstruction with a stereo event camera. *European Conference on Computer Vision(ECCV)*, 2018.

[25] A. Z. Zhu, D. Thakur, T. Özaslan, B. Pfrommer, V. Kumar, and K. Daniilidis. The multivehicle stereo event camera dataset: An event camera dataset for 3d perception. *IEEE Robotics and Automation Letters*, 3(3):2032–2039, July 2018.

[26] Alex Zihao Zhu, Yibo Chen, and Kostas Daniilidis. Realtime time synchronized event-based stereo. *European Conference on Computer Vision(ECCV)*, 2018.

[27] Alex Zihao Zhu, Liangzhe Yuan, Kenneth Chaney, and Kostas Daniilidis. Ev-flownet: Self-supervised optical flow estimation for event-based cameras. *Robotics: Science and Systems*, 2018.

[28] Alex Zihao Zhu, Liangzhe Yuan, Kenneth Chaney, and Kostas Daniilidis. Unsupervised Event-based Learning of Optical Flow, Depth, and Egomotion. *arXiv e-prints*, page arXiv:1812.08156, Dec 2018.