Self-Supervised Learning of Inlier Events for Event-based Optical Flow

Jun Nagata[1,2]
jnagata@aoki-medialab.jp
Yoshimitsu Aoki[1]
aoki@elec.keio.ac.jp

[1] Keio University
Kanagawa, Japan
[2] Denso IT Laboratory
Tokyo, Japan

Abstract

Event cameras asynchronously report per-pixel intensity changes with high temporal resolution. Their sparse and temporally precise nature is well suited for fast visual flow estimation. The normal optical flow can be estimated by fitting a plane to spatio-temporal events. However, least-squares plane fitting suffers from outliers due to the significant noise of events. Existing methods involve 1) iterative outlier rejection or 2) goodness-of-fit rejection after single-shot planar fitting from greedily selected spatially neighboring events. In contrast to these methods, we propose a method of selecting the events supporting a plane before performing a fitting, using the inlier probability from a lightweight neural network that captures both global and local structures. During inference, single-shot planar fitting is performed from only events with a higher inlier probability. We model each event selection by a Bernoulli distribution with the inlier probability and train the network to maximize the inlier count while sampling in a self-supervised manner. We verify that our event selection improves the accuracy of optical flow estimation with publicly available real data.

1 Introduction

Optical flow is the motion vector of objects, surfaces, and edges projected onto the image plane. It is important for understanding dynamic scenes with self-driving cars, drones, and automated robots. The classical image-based methods use the brightness consistency assumption that luminance does not change between two images with a certain interval [13, 17]. In recent years, many methods have been developed for complex feature extraction and association at multiple resolutions using deep convolutional neural networks on GPUs [10, 24]. While rich computational power and a large model are driving increasing accuracy, fast sensing and low-power processing are still required for resource-limited applications.

Event-based cameras offer many advantages over standard frame-based cameras, such as low latency, high temporal resolution, and a high dynamic range, and are thus suitable for fast visual flow computations. Because of the nature of their asynchronous response only to changes in the luminance of each pixel, events inherently contain motion information.

Using the spatio-temporal structure of events, the normal flow can be obtained by fitting a plane to events under a local constant velocity assumtion [6]. This approach is described in detail in Section 3. The visual flow based on planar fitting is an event-driven algorithm that takes advantage of asynchrony and spatial sparsity to achieve low latency. However, least-squares plane fitting suffers from outliers due to significant event noise [22]. In previous work [2, 6, 18], iterative outlier rejection has been used, but this increases computation, especially in noisy scenes. Recently, a method has been developed that greedily selects events with the largest timestamps in the spatial neighborhood and performs single-shot plane fitting [16]. This method is non-iterative, but events selected by simple rules may contain noise events, so leading to poor estimation results. In fact, we found that many estimation results were rejected in post-processing due to the goodness-of-fit criterion.

The objective of this research is to achieve accurate single-shot estimation by selecting events that support a plane before fitting. We propose a method of selecting the events based on the inlier probability from a lightweight neural network that captures both global and local structures. During inference, single-shot planar fitting is performed from only events with higher inlier probability. In the training phase, we model each event selection by a Bernoulli distribution with the inlier probability and train the network to maximize the inlier count while sampling in a self-supervised manner. We verify that our event selection improves the accuracy of optical flow estimation with publicly available real data.

2 Related Work

In recent years, various methods have been developed for estimating optical flow from event data. In a variant of the Lucas-Kanade method [5], which uses luminance changes approximated by events, there is a loss of luminance information in the conversion from intensity to events. More recently, there are methods based on block matching of time slices of events [15] and variational methods for simultaneous image reconstruction and optical flow estimation [3]. There are methods for regressing optical flow directly with a deep convolutional neural network by inputting timestamps and frequencies of events as images [11, 25, 26, 27]. These methods are computationally intensive because of their image-based dense processing.

The event-based visual flow estimation algorithm with planar fitting takes advantage of the sparsity and the temporal nature of events. However, least-squares plane fitting suffers from outliers due to significant event noise [22]. Outlier rejection done in an iterative way [2, 6, 18] will result in loss of latency. Recently, there is a method that greedily selects events with the largest timestamps in the neighborhood of the incoming event and performs single-shot planar fitting [16]. This method is non-iterative, but the simple rule may select noise events, and many of the estimation results are rejected in post-processing due to poor fitting.

We solved this problem by noise-robust event selection that takes into account the structure of a plane before fitting. We realize this selection by using a lightweight neural network that captures both global and local structures, outputting the inlier probability for each event from a set of local events.

There are several methods for outputting the importance of each data point using a neural network. DeepFit [4] uses a neural network to output a weight for each point in a weighted least-squares method during jet fitting of a point cloud. It requires the ground truth of the normals for training. In contrast, since events are noisy, we design a framework that excludes points that are not associated with a plane as outliers. Our network is trained without any labels by using sampling. NG-RANSAC [7] is a method of sampling the minimum set using a categorical distribution represented by a neural network, rather than a uniform distribution, when creating the hypothesis pool in the RANSAC algorithm. Following the RANSAC framework, multiple hypotheses are generated and the best one is selected during inference. In contrast, we model whether to select each data point with a Bernoulli distribution, and a single-shot estimation is performed with points with higher probability during inference.

3 Event-based Optical Flow by Fitting a Plane

An event-based camera outputs an event when the logarithmic intensity changes by more than a pre-defined threshold value at each pixel. An event is represented by a tuple (x,y,t,p) of four values, where x,y are the spatial localtion of the change in the pixel coordinate, t is the timestamp of the triggerd event, and $p \in \{+1,-1\}$ is the polarity that corresponds to the positive or negative intensity change, respectively.

By continuously reacting to changes in luminance at each pixel, the most recent events form a surface in space-time. The spatio-temporal surface is well known as the surface of active events (SAE) [6], which is defined mathematically as follows:

$$\Sigma_e : \mathbb{R}^2 \rightarrow \mathbb{R} \tag{1}$$

$$\mathbf{x} \mapsto \Sigma_e(\mathbf{x}) = t \tag{2}$$

where $\mathbf{x} = [x,y]^\top$. The spatial gradient vector of the SAE $\nabla\Sigma_e(\mathbf{x}) = \left[\frac{\partial\Sigma_e(\mathbf{x})}{\partial x}, \frac{\partial\Sigma_e(\mathbf{x})}{\partial y}\right]^\top$ represents the rate and the direction of change of time with respect to the space, i.e., its components are the inverse of the components of the motion vector at $\mathbf{x}$. Under the assumption of constant local velocity, the SAE can be represented by a plane $t = ax + by + c$. Planar parameters are estimated by fitting a plane using the least-squares method. Let $B = [x_j, y_j, 1]_{j=1,\ldots,n} \in \mathbb{R}^{n\times 3}$ be the planar basis and $\mathbf{t} = [t_1,\ldots,t_n]^\top$ be the target, and the plane parameter $\beta = [a,b,c]^\top$ is estimated as follows:

$$\hat{\beta} = (B^\top B)^{-1} B^\top \mathbf{t} \tag{3}$$

where n is the number of the events. The optical flow is calculated as the normal flow with local planar parameters as follows:

$$\mathbf{v} = \frac{1}{\|\nabla\Sigma_e(\mathbf{x})\|}\nabla\hat{\Sigma}_e(\mathbf{x}) = \frac{1}{a^2+b^2}\begin{bmatrix} a \\ b \end{bmatrix} \tag{4}$$

where $\nabla\hat{\Sigma}_e(\mathbf{x}) = \frac{\nabla\Sigma_e(\mathbf{x})}{\|\nabla\Sigma_e(\mathbf{x})\|}$ is the unit vector of $\nabla\Sigma_e(\mathbf{x})$. A series of the normal flow estimation is performed in an event-by-event manner for each new event.

In summary, when a new event occurs, a plane fitting is performed on the neighborhood events, and the normal flow is computed from the estimated planar parameters. Outlier rejection is necessary because events contain a lot of noise that makes least-squares fitting poor. In the next section, we describe the proposed event selection method that takes into account planar structure without repetition using a neural network.

4 Proposed Method

In this section, we explain the proposed method of selecting events based on the inlier probability by the neural network and self-supervised learning of the network. First, we describe

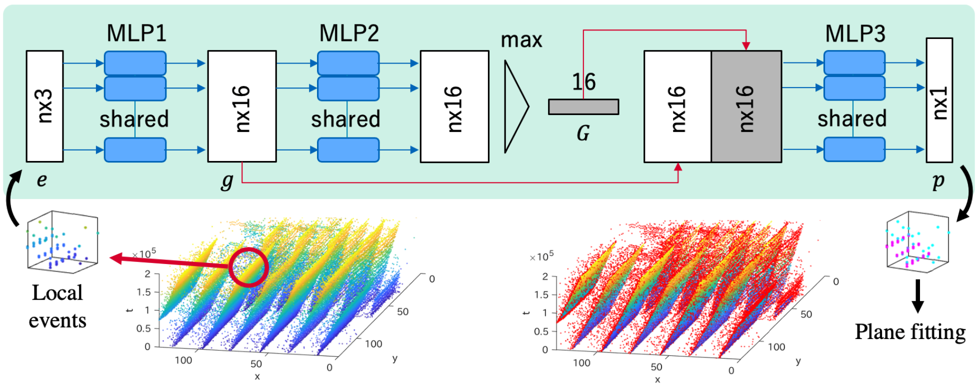

Figure 1: Our network structure using a shallow PointNet. It takes as input the spatio-temporal events of the coordinates centered at the EOI in the local window and outputs the inlier probabilty p of each event e. To capture the relationship between a plane and individual events, we extracted local features g by MLP and global features G by max aggregation. The red dots indicate the EOIs that were determined to be outliers.

the design of the input-output and the network for event inlier probability, as well as the operations during inference (Section 4.1). Second, we describe sampling-based self-supervised learning of the network, including modeling of sampling distributions and the derivation of the gradients (Section 4.2).

4.1 Event inlier probabilty

For outlier-robust single-shot plane fitting, inlier selection must take into account the geometric structure of the plane formed by spatio-temporal events. Therefore, we propose a scheme to output the inlier probability for each event from a local event cloud using a neural network that captures the three-dimensional structure.

To achieve low-latency event processing, the algorithm is performed for each new event, denoted as the event of interest (EOI). When an EOI $(\mathbf{x}_i, t_i, p_i)$ arrives, a group of events in the spatio-temporal neighborhood of the EOI is formed in the coordinate centered at the EOI as follows: $S_i = \{e_j = (\Delta\mathbf{x}_j, \Delta t_j) : \mathbf{x}_j \in \mathcal{N}(\mathbf{x}_i), t_j > t_i - \delta t, p_j = p_i\}$ where $\Delta\mathbf{x}_j = \mathbf{x}_j - \mathbf{x}_i$, $\Delta t_j = t_j - t_i$, and $\mathcal{N}(\mathbf{x}_i)$ is a square patch with length r centered at $\mathbf{x}_i$. It is then passed through a neural network to output the inlier probability p_j of each event.

We used a PointNet [21] architecture as shown in Fig. 1 to capture global and local geometric structures while taking advantage of event sparsity. We emphasize that it is a shallow network because it only needs to capture the relationship between a simple plane and each event in a small event cloud. The network outputs a local representation $g(e_j)$ by a multi-layer perceptron (MLP) for each event and a global representation $G(S_i)$ by subsequent MLP transformation and max aggregation. These representations are concatenated and fed into a MLP $h(\cdot)$, followed by the sigmoid function to output the event inlier probabilty p_j, as follows:

$$z_j = h(G(S_i), g(e_j)), \tag{5}$$
$$p_j = \sigma(z_j). \tag{6}$$

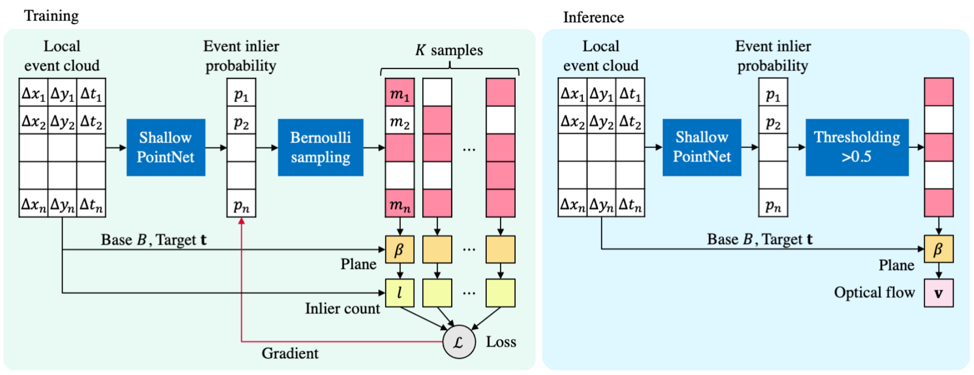

Figure 2: Our self-supervised learning and inference procedures. During training (left), the probabilities outputs of the network are used as parameters of a Bernoulli distribution to sample each event. At each event, we sample whether to select (m=1, red) or not (m=0, white), and estimate the plane on the selected events and calculate the negative inlier count loss. The final loss $\mathcal{L}$ is the average of K samples, and the gradients are calculated for network outputs. During inference (right), the inlier events (red) are selected by thresholding the probabilities, and the plane is estimated using only these events.

During inference, an event with a probability greater than a threshold of 0.5 is considered as an inlier. When the EOI is inlier ($p_i > 0.5$), plane fitting is performed on the inlier event cloud $\hat{S}_i = \{e_j : e_j \in S_i, p_j > 0.5\}$ by Eq. 3, followed by normal flow calculation by Eq. 4, as shown on the right side of Fig. 2. At this time, the plane is defined by coordinates centered at the EOI as $\Delta t_j = a\Delta x_j + b\Delta y_j + c$. Thus, the base and target in Eq. 3 are $B = [\Delta x_j, \Delta y_j, 1]_{j=1,\dots,n}$ and $\mathbf{t} = [\Delta t_1, \dots \Delta t_n]^\top$, respectively.

4.2 Self-supervised learning

We describe how to train the network that outputs event-inlier probabilities in a self-supervised manner without labels. The goodness of the output probability is measured by how many events support the estimated plane β. The negative inlier count loss is defined as the negative number of events for which the magnitude of the time residuals is less than a threshold θ as follows:

$$l = -\#\{e_j : e_j \in S_i, \left|\Delta\hat{t}_j - \Delta t_j\right| < \theta\} \tag{7}$$

where $\Delta\hat{t}_j = \hat{a}\Delta x_j + \hat{b}\Delta y_j + \hat{c}$. The symbol # denotes the number of elements in the set. Since the derivative of the loss function with respect to the network's output p cannot be computed, we define the training objective as the minimization of the expected loss inspired by [7, 8]. We define the binary variable m_j whether to use the event or not with a Bernoulli distribution f with a parameter of probability p_j for each event e_j independently, as follows:

$$m_j \sim f(\cdot; p_j) = p_j^{m_j}(1-p_j)^{1-m_j}. \tag{8}$$

The expected loss is defined as follows:

$$\mathcal{L} = \mathbb{E}_{\mathbf{m}\sim f(\cdot;\mathbf{p})}\left[l(\mathbf{m})\right] \tag{9}$$

where $\mathbf{m} = [m_1, ..., m_n]^\top$ and $\mathbf{p} = [p_1, ..., p_n]^\top$. $l(\mathbf{m})$ is the negative inlier count loss for the plane computed from the selected events $\hat{S}_i = \{e_j : e_j \in S_i, m_j = 1\}$. The derivative of the expected loss with regard to the distribution parameter can be computed [23]. To stabilize the gradient, we compute the gradient with regard to the logit instead of the probability as follows:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{z}} = \mathbb{E}_{\mathbf{m}} \left[l(\mathbf{m}) \frac{\partial}{\partial \mathbf{z}} \log f(\cdot; \mathbf{p}) \right] \tag{10}$$

where $\mathbf{z} = [z_1, ..., z_n]^\top$. Here, $\frac{\partial}{\partial z_j} \log f$ is derived for each state m_j as follows:

$$\frac{\partial}{\partial z_j} \log f = \begin{cases} 1 - \sigma(z_j) & (m_j = 1) \\ -\sigma(z_j) & (m_j = 0) \end{cases}. \tag{11}$$

Proof: Substitute Eq. 6 for the logarithm of the distribution (Eq. 8) and differentiate with regard to z_j to obtain $\frac{\partial}{\partial z} \log f = m \frac{\sigma'(z)}{\sigma(z)} - (1-m) \frac{\sigma'(z)}{1-\sigma(z)}$. Then, substitute the well-known derivative of the sigmoid function $\sigma'(z_j) = \sigma(z_j)(1 - \sigma(z_j))$ into this equation and organize it for each value of m_j.

The number of all possible states of $\mathbf{m}$ is 2^n, and it is infeasible to integrate them to calculate the expectation. Therefore, we approximate the gradient by the average of the K drawn samples:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{z}} \approx \frac{1}{K} \sum_{k=1}^{K} \left[l(\mathbf{m}_k) \frac{\partial}{\partial \mathbf{z}} \log f(\cdot; \mathbf{p}) \right] \tag{12}$$

where $\mathbf{m}_k \sim f(\cdot; \mathbf{p})$ is the k-th drawn sample. Because the approximation is done by sampling, the gradient variance can be high. We apply the variance reduction technique of subtracting the average $\bar{l}$ as in [7]:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{z}} \approx \frac{1}{K} \sum_{k=1}^{K} \left[(l(\mathbf{m}_k) - \bar{l}) \frac{\partial}{\partial \mathbf{z}} \log f(\cdot; \mathbf{p}) \right]. \tag{13}$$

The gradients are back-propagated to the network parameters, and the parameters are updated by descending the gradients.

5 Experiment

To verify the effectiveness of the proposed method, we compared the accuracy of the estimated optical flow with the full model of SOFEA [16], the latest non-iterative event-based optical flow estimation method. In the proposed method, planar fitting was performed in two different parameterizations. One is the general plane in the EOI center coordinates described earlier, $\Delta t = a\Delta x + b\Delta y + c$ (denoted as w/ bias). The other is a plane through the EOI $(\Delta x_j, \Delta y_j, \Delta t_j) = (0, 0, 0)$ as $\Delta t = a\Delta x + b\Delta y$, similar to SOFEA (denoted as w/o bias). Two metrics are used to evaluate optical flow accuracy. One is the relative endpoint error (REE) which is the magnitude of the error vector relative to the ground truth in percentage, and the other is the angular error (AE) in space-time [22]. We report sample means and standard deviations for both metrics. If the slope of the fitted plane is incorrectly small, the length of the estimated flow will be unusually large, significantly worsening the evaluation. Large estimates with more than corner-to-corner movement in the image at 30 fps were excluded.

5.1 Dataset

For evaluation, we used three types of data in SOFEA [16].
Stripes: This is a planar scene with stripes sliding at a constant speed from [18]. The camera is a dynamic vision sensor (DVS) [14] with a resolution of 128×128.
Rotating bar: This is the data capturing a scene with a rotating bar used for evaluation in [2, 12]. The velocity is assumed to be constant within a local patch for planar fitting, although it varies in the radial direction. The camera is an asynchronous time-based image sensor (ATIS) [20] with a resolution of 304×240.
HDR: This is the data capturing a high dynamic range (HDR) scene by sliding the camera, which is a part of the Event Camera Dataset [19]. The camera is a dynamic and active pixel vision sensor (DAVIS) [9] with a resolution of 240×180.

5.2 Training and implementation details

Here, MLP1 and MLP2 are single-layer perceptrons of size 16, and MLP3 is a 2-layer MLP of size 16 and 1, as shown in Fig. 1. All layers except the final layer contain batch normalization and ReLU activation. The patch size r of the input was set to 7 and the temporal window size dt to 50 ms. For efficient implementation, the maximum event buffer size for a single pixel was set to 2. The number of sampling K was set to 64. The input timestamp Δt was normalized by dividing by 10 ms. Δx and Δy, which are integers in $[-3, 3]$ when $r = 7$, were not normalized. As data augmentation, for each sample, the timestamps were scaled by a factor drawn from a uniform distribution with $[5/6, 6/5]$. For the rotating-bar sequence, the flip augmentation in the x and y directions with a probability of 0.5 was added.

The data used in SOFEA were taken with different cameras and in different environments. The experiments aimed to confirm the effectiveness of self-supervised learned inlier points within each scene, rather than generalization between sequences. In each sequence, 0.8 of the total temporal length was used for training and 0.2 for testing. The threshold was set to 5 ms for the HDR scene and 1 ms for the other scenes. We introduced the goodness-of-fit noise rejection with the number of events supporting the estimated plane as a metric, similar to SOFEA. The threshold of temporal residuals is the same as in training, and the number of events N_{sp} is set to 7.

We used a batch size of 64, the Adam optimizer, and a learning rate of 10^{-4}. The implementation for training was done in PyTorch and trained on a single Nvidia RTX A6000 GPU. The convergence of the loss was related to the amount of noise, requiring 30,000 iterations for the less noisy rotating-bar squence and 500,000 iterations for the noisy HDR squence. The time taken per iteration was 0.025 seconds for training. The tests were performed in CPU-based MATLAB.

5.3 Results

The numerical results of the experiments are shown in Table 1 and the visualizations are shown in Fig. 3. In all scenes, either or both of the proposed methods achieve better accuracy than the existing method in both metrics. The values in Table 1 differ from those in the SOFEA paper because we evaluated on a test set. Looking at the estimated optical flow of the HDR scene, our method has more estimated points than SOFEA and is closer to ground truth. In practice, the estimated points were 288k for our method (w/ bias) and 114k for SOFEA in the HDR scene. Figure 4 shows that variation in the number of estimated points and the

Table 1: Experimental results. The evaluation metrics are the relative end-point error (REE) and the angular error (AE). Lower is better for both.

Scene	Stripes		Rotating bar		HDR	
	REE [%]	AE [°]	REE [%]	AE [°]	REE [%]	AE [°]
SOFEA[16]	17.3±14.4	2.06±7.72	22.0±31.1	6.15±6.71	31.6±45.5	8.82±10.4
Ours (w/o bias)	12.3±6.56	0.67±0.57	**16.9 ± 12.3**	**5.72 ± 4.30**	**30.2 ± 33.7**	**5.79 ± 4.76**
Ours (w/ bias)	**10.8 ± 6.09**	**0.65 ± 0.55**	17.5±15.4	5.79±4.76	37.3±48.8	7.16±7.68

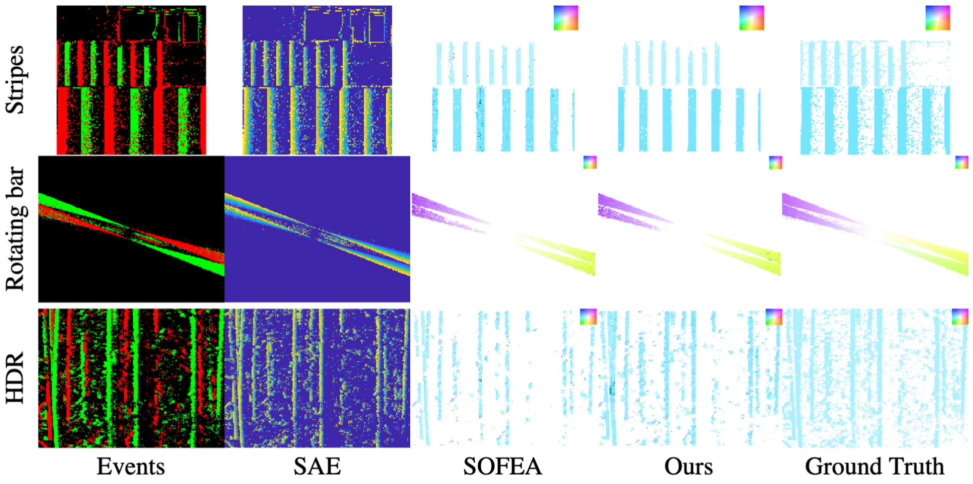

Figure 3: Visualization of experimental results from test data. The rows show each sequence. The columns, from left to right, show the events, the SAE, the optical flow by SOFEA [16], the optical flow by our proposed method (w/ bias), and the ground truth optical flow. All are displayed as accumulated for 50 ms. The events are indicated by red for positive and green for negative polarity. In the optical flow display, hue represents orientation and color intensity represents length, as in the color map shown in the upper-right corner of each image.

average of REE when varying N_{sp} by 4 from 3 to 19, which is the tightness of the estimated-points rejection with goodness of fit in post-processing. SOFEA can achieve low errors when evaluating fewer points by the strict rejection, but the errors increase as the number of evaluation points is increased by loosening the goodness-of-fit criterion. In contrast, our method keeps low errors regardless of the tightness of the rejection. This shows that greedy selection in SOFEA contains many outliers, and many of the estimates are rejected by the goodness-of-fit check in post-processing. In contrast, our event selection is able to exclude outliers in advance of fitting by capturing the planar structure with the neural network.

5.4 Runtime

Runtime was measured for the event selection for plane fitting after local patches were extracted. We made a comparison between two methods: our event selection based on the inlier probability from the neural network output and SOFEA's greedy selection of associated neighbouring events in their original implementation. Both implementations are event-by-event processing. We measured the time in MATLAB using a 3.30GHz Core i9-9820X

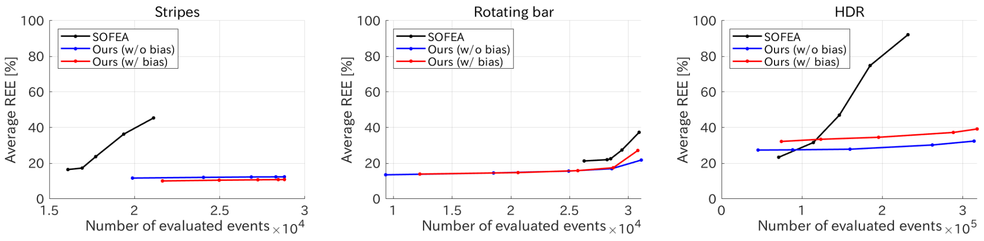

Figure 4: Variation in the number of estimated points and the average of REE when varying the tightness of the post-processing checking with the goodness of fit for the three scenes.

CPU. SOFEA's runtime was about 2×10^{-4} seconds per event, while our method's was about 3×10^{-5} seconds per event. Our method uses a shallow network, which allows for efficient event selection. Note that there may be room to optimize the implementation of both approaches in terms of both software and hardware.

6 Conclusion

We proposed a method for event selection using a neural network for event-based optical flow with plane fitting. The network that captures global and local structures allows events supporting a plane to be determined before fitting is performed. We model each event selection with a Bernoulli distribution with the inlier probability of the network output and train the model to maximize the inlier count while sampling in a self-supervised manner. During inference, the inlier event set is determined by thresholding the inlier probability and plane fitting is performed in a single-shot way. Experiments with real data showed that our event selection improved the accuracy of optical flow estimation. Furthermore, when we evaluated at many points by relaxing the criteria for rejection of estimates, the accuracy of the existing method degraded, while the accuracy of our method did not. This indicates that our event selection is of higher quality than the existing greedy event selection.

Local plane fitting methods, such as the proposed method and SOFEA, have limitations due to the linear assumption and the aperture problem. If the linear assumption is violated, model errors may bias the estimation and reduce its accuracy. The spatial-temporal window needs to be set appropriately for the scene to satisfy the assumption to mitigate the error. These methods, in principle, can only obtain the normal flow, and require a wide range of consistency measurements to estimate the full flow. Akolkar *et al.* [1] have reported that the aperture-robust visual flow could be estimated in realistic scenes by linearizing a complex-shaped object with multiple small edges and multiscale pooling of normal flows. Therefore, verifying the proposed method under the linear assumption is informative. Evaluation in realistic scenarios and generalization testing for scenes remain as future work.

References

[1] Himanshu Akolkar, Sio Hoi Ieng, and Ryad Benosman. Real-time high speed motion prediction using fast aperture-robust event-driven visual flow. *IEEE Transactions on*

Pattern Analysis and Machine Intelligence, 44(1):1–12, 2020. ISSN 0162-8828. doi: 10.1109/tpami.2020.3010468.

[2] Myo Tun Aung, Rodney Teo, and Garrick Orchard. Event-based Plane-fitting Optical Flow for Dynamic Vision Sensors in FPGA. *IEEE International Symposium on Circuits and Systems*, 2018. ISSN 02714310. doi: 10.1109/ISCAS.2018.8351588.

[3] Patrick Bardow, Andrew J. Davison, and Stefan Leutenegger. Simultaneous Optical Flow and Intensity Estimation from an Event Camera. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 884–892, 2016. ISSN 10636919. doi: 10.1109/CVPR.2016.102. URL http://ieeexplore.ieee.org/document/7780471/.

[4] Yizhak Ben-Shabat and Stephen Gould. DeepFit: 3D Surface Fitting via Neural Network Weighted Least Squares. *European Conference on Computer Vision (ECCV)*, 12346:20–34, 2020. ISSN 16113349. doi: 10.1007/978-3-030-58452-8_2.

[5] Ryad Benosman, Sio-Hoi Ieng, Charles Clercq, Chiara Bartolozzi, and Mandyam Srinivasan. Asynchronous frameless event-based optical flow. *Neural Networks*, 27:32–37, mar 2012. ISSN 08936080. doi: 10.1016/j.neunet.2011.11.001. URL http://dx.doi.org/10.1016/j.neunet.2011.11.001.

[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, feb 2014. ISSN 2162-237X. doi: 10.1109/TNNLS.2013.2273537. URL http://ieeexplore.ieee.org/document/6589170/.

[7] Eric Brachmann and Carsten Rother. Neural-Guided RANSAC: Learning Where to Sample Model Hypotheses. *IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4322–4331, 2019. ISSN 15505499. doi: 10.1109/ICCV.2019.00442.

[8] Eric Brachmann, Alexander Krull, Sebastian Nowozin, Jamie Shotton, Frank Michel, Stefan Gumhold, and Carsten Rother. DSAC - Differentiable RANSAC for Camera Localization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6684–6692, 2017. ISBN 9781538604571. doi: 10.1109/CVPR.2017.267.

[9] Christian Brandli, Raphael Berner, Minhao Yang, Shih Chii Liu, and Tobi Delbruck. A 240 $\times$ 180 130 dB 3 μs latency global shutter spatiotemporal vision sensor. *IEEE Journal of Solid-State Circuits*, 49(10):2333–2341, 2014. ISSN 00189200. doi: 10.1109/JSSC.2014.2342715.

[10] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Häusser, Caner Hazırbas, Vladimir Golkov, Patrick van der Smagt, Daniel Cremers, and Thomas Brox. FlowNet: Learning Optical Flow with Convolutional Networks Alexey. *IEEE International Conference on Computer Vision (ICCV)*, pages 2758–2766, 2015. ISSN 03794369.

[11] Mathias Gehrig, Mario Millhäusler, Daniel Gehrig, and Davide Scaramuzza. E-RAFT: Dense Optical Flow from Event Cameras. *International Conference on 3D Vision (3DV)*, 2021. URL http://arxiv.org/abs/2108.10552.

[12] Germain Haessig, Andrew Cassidy, Rodrigo Alvarez, Ryad Benosman, and Garrick Orchard. Spiking Optical Flow for Event-based Sensors Using IBM's TrueNorth Neurosynaptic System. pages 1–11, 2017. ISSN 19324545. doi: 10.1109/TBCAS.2018.2834558. URL `http://arxiv.org/abs/1710.09820`.

[13] B. K. B. Horn and B. G. Schunck. Determining Optical Flow. *Artificial Intelligence*, 17(1-3):185–203, 1981. ISSN 0004-3702. doi: 10.1016/0004-3702(81)90024-2.

[14] Patrick Lichtsteiner, Christoph Posch, Tobi Delbruck, and Senior Member. A 128 $\times$ 128 120 dB 15 μs Latency Asynchronous Temporal Contrast Vision Sensor. *IEEE Journal of Solid-State Circuits*, 43(2):566–576, 2008.

[15] Min Liu and Tobi Delbruck. Adaptive time-slice block-matching optical flow algorithm for dynamic vision sensors. *British Machine Vision Conference 2018, BMVC 2018*, 2019.

[16] Weng Fei Low, Zhi Gao, Cheng Xiang, and Bharath Ramesh. SOFEA: A non-iterative and robust optical flow estimation algorithm for dynamic vision sensors. *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2020. ISSN 21607516. doi: 10.1109/CVPRW50498.2020.00049.

[17] Bruce D. Lucas and Takeo Kanade. An Iterative Image Registration Technique with an Application to Stereo Vision, 1981.

[18] Elias Mueggler, Christian Forster, Nathan Baumli, Guillermo Gallego, and Davide Scaramuzza. Lifetime Estimation of Events from Dynamic Vision Sensors. *IEEE International Conference on Robotics and Automation (ICRA)*, 2015.

[19] 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. *International Journal of Robotics Research*, 36(2): 142–149, 2017. ISSN 17413176. doi: 10.1177/0278364917691115.

[20] Christoph Posch, Daniel Matolin, and Rainer Wohlgenannt. A QVGA 143 dB dynamic range frame-free PWM image sensor with lossless pixel-level video compression and time-domain CDS. *IEEE Journal of Solid-State Circuits*, 46(1):259–275, 2011. ISSN 00189200. doi: 10.1109/JSSC.2010.2085952.

[21] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. PointNet: Deep Learning on Point Sets for 3D Classification and Segmentation. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 652–660, 2017. ISSN 1063-6919. doi: 10.1109/3DV.2016.68.

[22] Bodo Rueckauer and Tobi Delbruck. Evaluation of Event-Based Algorithms for Optical Flow with Ground-Truth from Inertial Measurement Sensor. *Frontiers in Neuroscience*, 10(176), apr 2016. ISSN 1662-453X. doi: 10.3389/fnins.2016.00176. URL `http://journal.frontiersin.org/Article/10.3389/fnins.2016.001`

[23] John Schulman, Nicolas Heess, Theophane Weber, and Pieter Abbeel. Gradient estimation using stochastic computation graphs. *International Conference on Neural Information Processing Systems (NIPS)*, 2:3528–3536, 2015. ISSN 10495258.

[24] Zachary Teed and Jia Deng. RAFT: Recurrent All-Pairs Field Transforms for Optical Flow. *European Conference on Computer Vision (ECCV)*, 2020. URL http://arxiv.org/abs/2003.12039.

[25] Chengxi Ye, Anton Mitrokhin, Cornelia Fermüller, James A Yorke, and Yiannis Aloimonos. Unsupervised Learning of Dense Optical Flow, Depth and Egomotion from Sparse Event Data. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020. URL http://arxiv.org/abs/1809.08625.

[26] Alex Zhu, Liangzhe Yuan, Kenneth Chaney, and Kostas Daniilidis. EV-FlowNet: Self-Supervised Optical Flow Estimation for Event-based Cameras. *Proceedings of Robotics: Science and Systems*, jun 2018. doi: 10.15607/RSS.2018.XIV.062. URL http://www.roboticsproceedings.org/rss14/p62.pdf.

[27] Alex Zihao Zhu, Liangzhe Yuan, Kenneth Chaney, and Kostas Daniilidis. Unsupervised event-based optical flow using motion compensation. *European Conference on Computer Vision (ECCV) Workshop*, pages 711–714, 2018. ISSN 16113349. doi: 10.1007/978-3-030-11024-6_54.