EvLiDAR-Flow: Attention-Guided Fusion between Point Clouds and Events for Scene Flow Estimation

Ankit Sonthalia[*,1,2], Ramy Battrawy[*,1], René Schuster[1] and Didier Stricker[1]
[1]*German Research Center for Artificial Intelligence (DFKI GmbH), Kaiserslautern, Germany*
[2]*Tübingen AI Center, Eberhard Karls University Tübingen, Germany*
[1]*firstname.lastname@dfki.de*, [2]*ankit.sonthalia@tue.ai*

Keywords: LiDAR, Point Cloud, Event Camera, Bi-Directional Fusion, Attention, Scene Flow.

Abstract: In this paper, we propose the fusion of event streams and point clouds for scene flow estimation. Bio-inspired event cameras offer significantly lower latency and higher dynamic ranges than regular RGB cameras, and are therefore appropriate for recording high-speed motions. However, events do not provide depth information, which makes them unsuitable for scene flow (3D) estimation. On the other hand, LiDAR-based approaches are well suited to scene flow estimation due to the high precision of LiDAR measurements for outdoor scenes (e.g. autonomous vehicle applications) but they fail in the presence of unstructured regions (e.g. ground surface, grass, walls, etc.). We propose our EvLiDAR-Flow, a neural network architecture equipped with an attention module for bi-directional feature fusion between an event (2D) branch and a point cloud (3D) branch. This kind of fusion helps to overcome the lack of depth information in events while enabling the LiDAR-based scene flow branch to benefit from the rich motion information encoded by events. We validate the proposed EvLiDAR-Flow by showing that it performs significantly better and is robust to the presence of ground points, in comparison to a state-of-the-art LiDAR-only scene flow estimation method.

1 INTRODUCTION

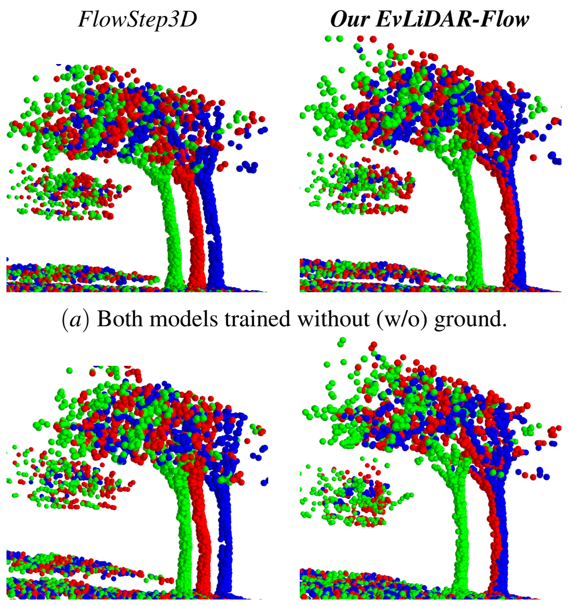
Scene flow estimation is one of the most important steps towards a robust understanding of scene dynamics. It involves constructing a 3D motion field of the scene and can be used in several applications related to navigation and autonomous driving systems.

Until very recently, RGB image-based vision systems (where the input is often a pair of stereo RGB images) have been widely used for scene flow estimation (Ma et al., 2019; Menze and Geiger, 2015; Saxena et al., 2019; Schuster et al., 2018). However, these systems perform poorly in excessively dim or bright environments due to the low dynamic range of RGB sensors. Moreover, the low frame rate typically supported by RGB cameras makes them prone to motion blur when attempting to capture high-speed motions.

On the contrary, event-based cameras are designed to provide high dynamic ranges and low temporal resolutions (Gehrig et al., 2021b; Low et al., 2020; Zhu et al., 2018) suitable for capturing high speed motions even under poor light conditions where RGB cameras suffer. Event cameras detect pixel-level brightness changes and record them as *events*. However, these cameras do not capture depth information, and therefore, do not provide sufficient information for scene flow estimation. Additionally, event data often suffers from noise and conventional methods often have to resort to possibly sub-optimal, hand-crafted noise removal techniques (Low et al., 2020).

Unlike event cameras, LiDAR sensors provide a highly accurate representation of the scene geometry in the form of point clouds. Although LiDAR-based approaches have achieved impressive results on scene flow estimation (Gu et al., 2019; Kittenplon et al., 2021; Liu et al., 2019), most of these methods operate under ideal settings and their accuracy is badly affected in the presence of large planar areas (e.g. the ground surface, grass, etc.). While scene flow estimation for these areas is not useful in practice, the mere presence of these areas can significantly worsen the performance of such models on other parts of the scene. For this reason, current LiDAR-based approaches remove the ground surface, often by applying a naïve threshold (Gu et al., 2019; Kittenplon et al., 2021; Puy et al., 2020; Wei et al., 2021; Wu et al., 2020). This threshold is not trivial to generalize and can mistakenly omit some important parts in the

[*]Equal Contribution



(a) Both models trained without (w/o) ground.

(b) Both models trained with (w/) ground.

Figure 1: Qualitative results on an example from DSEC (Gehrig et al., 2021a), a real-world data set; source points, target points and source points warped by the predicted scene flow have been color-coded as green, blue and red, respectively. Our EvLiDAR-Flow trained without ground points in the inputs, performs better than the state-of-the-art LiDAR only approach FlowStep3D (Kittenplon et al., 2021) (top right and top left, respectively). Even after training both models with ground points, FlowStep3D does not perform well, while our model continues to produce accurate flow estimates (bottom left and bottom right, respectively). The fusion of events and point clouds makes our model robust to the presence of such points in the scene.

scene (e.g. if the ground surface has an incline). Even with the successful removal of ground points, most real world scenes still contain other problematic areas like grass, buildings and walls.

In order to exploit the properties of events for scene flow estimation and to overcome the aforementioned drawbacks of LiDAR-based approaches, we introduce EvLiDAR-Flow, a neural network architecture which, using a learnable attention mechanism, enables the event and point cloud modalities to complement each other towards successful scene flow estimation especially in the presence of large planar areas (e.g. the ground surface). As illustrated in Figure 1, the proposed EvLiDAR-Flow is largely unaffected by the presence of such areas.

Previous works using sensor fusion for scene flow estimation have largely used either RGB-D (e.g. Kinect) or RGB images with LiDAR point clouds in different modes of fusion (Battrawy et al., 2019; Liu et al., 2022; Rishav et al., 2020; Teed and Deng, 2021). To the best of our knowledge, ours is the first work to fuse events and point clouds in the context of scene flow.

In this work, we make the following key contributions:

1. We propose EvLiDAR-Flow – a deep neural network architecture which fuses events and point clouds for jointly estimating scene flow from the point cloud (3D) branch and optical flow from the event (2D) branch.
2. The proposed EvLiDAR-Flow uses a fully learnable attention-based fusion module which forces feature sets from both modalities to pay attention to specific parts of each other. Through an ablation study, we demonstrate the superiority of our bi-directional fusion module over uni-directional fusion.
3. We demonstrate with the help of quantitative and qualitative results, that our proposed EvLiDAR-Flow outperforms the state-of-the-art LiDAR-only scene flow model FlowStep3D (Kittenplon et al., 2021) on the real-world data set DSEC (Gehrig et al., 2021a). Moreover, EvLiDAR-Flow is robust to the presence of ground points which are otherwise detrimental to the performance of FlowStep3D.

2 RELATED WORK

Before 3D scene flow, research in scene understanding was restricted to optical flow, i.e. estimating the motion field only on a two-dimensional plane using image-based vision systems (Dosovitskiy et al., 2015; Hui et al., 2018; Ilg et al., 2017; Sun et al., 2018; Teed and Deng, 2020; Weinzaepfel et al., 2013; Xu et al., 2017).

Event-based Optical Flow: Eliminating the limitations of frame-based images (e.g. motion blur, low dynamic range, etc.), event streams (Gallego et al., 2020) are quickly gaining traction in the field of optical flow estimation (Hu et al., 2022; Lee et al., 2021; Low et al., 2021; Low et al., 2020). Notable examples in deep learning based optical flow estimation are Spike-FlowNet (Lee et al., 2020), EV-FlowNet (Zhu et al., 2018) and E-RAFT (Gehrig et al., 2021b). E-RAFT (Gehrig et al., 2021b) has recently established state-of-the-art on the DSEC (Gehrig et al., 2021a) data set. However, events often contain considerable noise (Gallego et al., 2020) and provide only 2D information. Hence, unlike previous approaches, we use events in fusion with another modality of vision for robustly estimating 3D scene flow instead of 2D optical flow.

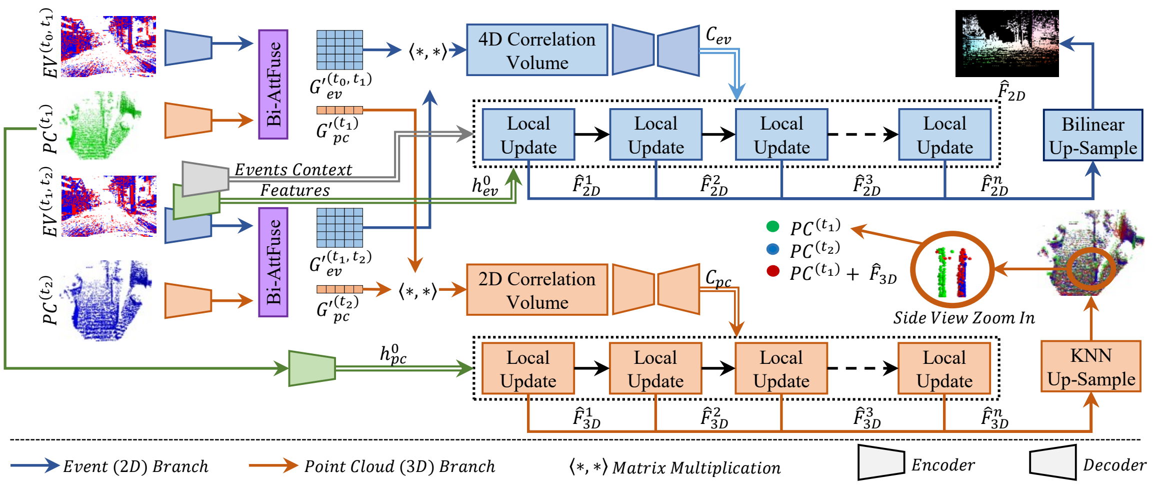


Figure 2: Model Architecture of the Proposed EvLiDAR-Flow.

LiDAR-based Scene Flow: LiDAR-based approaches have recently been shown to be superior in comparison to other modalities (e.g. stereo and RGB-D) for capturing 3D motion fields. FlowNet3D (Liu et al., 2019) is the first LiDAR-based scene flow approach which applies deep learning directly on point clouds (based on PointNet++ (Qi et al., 2017)) to compute scene flow in conjunction with a correlation volume. HPLFlowNet (Gu et al., 2019) further enhances the correlation layer by computing correlations at multiple scales (based on SPLATNet (Su et al., 2018)) and using sparse bilateral convolutional networks on a high-dimensional lattice. PointPWC-Net (Wu et al., 2020) further introduces hierarchical scene flow estimation based on PointConv (Wu et al., 2019). FLOT (Puy et al., 2020) poses scene flow estimation as the task of finding soft correspondences between consecutive point clouds. FlowStep3D (Kittenplon et al., 2021) follows RAFT (Teed and Deng, 2020) and applies an unrolling technique, for iteratively refining the scene flow estimate and achieves state-of-the-art results. PV-RAFT (Wei et al., 2021) follows the same approach but introduces point-voxel correlation fields. Although these LiDAR-based approaches have achieved impressive results, they still suffer in the presence of planar regions (e.g. ground surface for autonomous vehicles) where no rich 3D information is present. The presence of such areas can negatively impact the performance of the model on other, more important parts of the scene. To this end, we take the advantage of event streams for more robustly estimating scene flow even in the presence of unstructured regions.

Fusion for Scene Flow: Prior works have made use of fusion between two modalities for enhancing scene flow estimation (Battrawy et al., 2019; Liu et al., 2022; Rishav et al., 2020). LiDAR-Flow (Battrawy et al., 2019) fuses stereo images with point clouds. DeepLiDARFlow (Rishav et al., 2020) fuses sparse LiDAR points with only a monocular image instead of a pair of stereo images. CamLiFlow (Liu et al., 2022) further introduces multi-level fusion with a learnable interpolation module.

Unlike these approaches which fuse frame-based images and point clouds, we propose to fuse event streams and point clouds in a bi-directional manner. Through the fusion mechanism used in our EvLiDAR-Flow, we aim to use the richness in the information encoded by event streams in conjunction with the precision of LiDAR data for robust scene flow estimation.

3 EvLiDAR-Flow

Our architecture simultaneously operates on two consecutive LiDAR point clouds $PC^{(t_1)}$ and $PC^{(t_2)}$ captured at timestamps t_1 and t_2 respectively, and two consecutive event streams $EV^{(t_0,t_1)}$ and $EV^{(t_1,t_2)}$ for the time intervals $[t_0,t_1]$ and $[t_1,t_2]$ respectively, where $t_2 > t_1 > t_0$. Additionally, in order to approximately synchronize the two modalities, $t_2 - t_1 = t_1 - t_0 \approx \tau$, where τ is also the time interval between two consecutive scans of the LiDAR sensor. EvLiDAR-Flow simultaneously estimates optical flow (2D) and scene flow (3D) between timestamps t_1 and t_2, where t_1 is the reference timestamp. Our architecture consists of an event (2D) branch, a point cloud (3D) branch and our bi-directional attentive Fusion Module (cf. Figure 2).

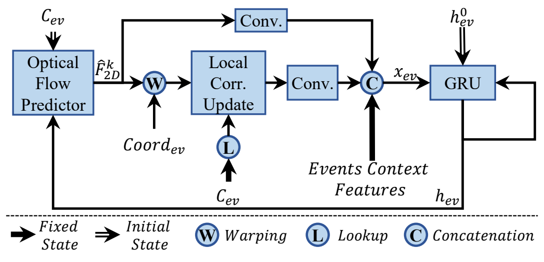


Figure 3: Local Update Unit (2D Branch).

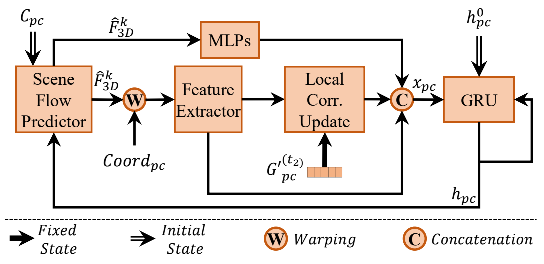


Figure 4: Local Update Unit (3D Branch).

3.1 Event (2D) Branch

The event branch of our model is inspired from E-RAFT (Gehrig et al., 2021b). The two event streams $EV^{(t_0,t_1)}$ and $EV^{(t_1,t_2)}$ are first encoded into voxelized representations $V^{(t_0,t_1)}$ and $V^{(t_1,t_2)}$, respectively, by discretizing the temporal dimension as described in (Zhu et al., 2019). The resulting event volumes are of the shape $B \times H \times W$ where H and W are the height and width of the frame, respectively, and B is the number of bins into which the time interval of the event stream has been divided. Each bin accumulates event information from its corresponding slice of the interval. The event volumes thus obtained are further encoded into local features $G^{(t_0,t_1)}_{ev}$ and $G^{(t_1,t_2)}_{ev}$, respectively, using a shared encoder. Both local feature volumes are shaped $D_2 \times H_l \times W_l$, where $H_l = H/8$, $W_l = W/8$ and D_2 is the number of channels. Our attention module (cf. Section 3.3.2) then fuses these local event features with local point features (cf. Section 3.2), generating $G'^{(t_1,\,t_2)}_{ev}$ and $G'^{(t_1,\,t_2)}_{ev}$, respectively. The **point-aware** local event features are further used to construct a 4D correlation volume C_{ev}. The second event volume $V^{(t_1,t_2)}$ is also encoded into context feature maps using an encoder with an architecture identical to that of the local feature encoder, but with different weights. The rest of the computation is handled by the Local Update Unit (cf. Figure 3) which first produces an initial flow estimate and then iteratively updates the same, for a number of refinement steps. The Local Update Unit includes a Gated Recurrent Unit (GRU), which at each refinement step, accepts: (1) the context features, (2) the local correlation features, and (3) the flow estimate $\hat{F}^{k-1}_{2D}$ from the previous refinement step $k-1$. The GRU then outputs a refined flow estimate $\hat{F}^k_{2D}$ which is used to warp and thereby update the local correlation features of all pixel positions for feeding as inputs in the next refinement step $k+1$. With each refinement, the objective of the GRU is to incrementally reduce the gap between the ground truth optical flow and the estimated flow $\hat{F}^k_{2D}$.

3.2 Point Cloud (3D) Branch

Our scene flow estimation branch exploits concepts used in FlowStep3D (Kittenplon et al., 2021), a recent state-of-the-art neural network architecture which directly operates on point clouds. Both the source point cloud $PC^{(t_1)}$ and the target point cloud $PC^{(t_2)}$ are encoded into local feature sets $G^{(t_1)}_{pc}$ and $G^{(t_2)}_{pc}$, respectively, using the set abstraction layer described in FlowNet3D (Liu et al., 2019). The encoders for the source and target point clouds share their weights. Our attention module (cf. Section 3.3.2), then fuses these local point features $G^{(t_1)}_{pc}$ and $G^{(t_2)}_{pc}$ with local event features from the 2D branch (cf. Section 3.1), generating $G'^{(t_1)}_{pc}$ and $G'^{(t_2)}_{pc}$, respectively.

The **event-aware** local point features are used to compute the global correlation volume C_{pc} which, in turn, is used to produce an initial estimate of scene flow across the consecutive LiDAR scans. The scene flow is further iteratively updated by the Local Update Unit (cf. Figure 4). At each refinement step k, the Local Update Unit accepts: (1) the hidden state h^{k-1}_{pc}, (2) the flow estimate $\hat{F}^{k-1}_{3D}$ from the previous step, (3) the features of the previous flow estimate, (4) the local features of the source point cloud warped by the previous flow estimate, i.e., $PC^{(t_1)} + \hat{F}^{k-1}_{3D}$, and finally, (5) the local flow embedding between the warped source $PC^{(t_1)} + \hat{F}^{k-1}_{3D}$ and the target $PC^{(t_2)}$. These inputs are concatenated together and passed into a Gated Recurrent Unit (GRU). With each step, the objective is to reduce the gap between the warped source $PC^{(t_1)} + \hat{F}^{k-1}_{3D}$ and the target $PC^{(t_2)}$, i.e. by generating more and more accurate estimates.

3.3 Fusion Module

3.3.1 Motivation

We wish to fuse the local features of the source point cloud $PC^{(t_1)}$ with the local features of the first event volume $V^{(t_0,t_1)}$ and a similar operation for the features of the target point cloud $PC^{(t_2)}$ and the second event volume $V^{(t_1,t_2)}$.

An event volume (Zhu et al., 2019) is generated from an event stream and therefore contains information from a time *interval* rather than a single timestamp. For instance, each channel in $V^{(t_0,t_1)}$ roughly resembles what the scene must have looked like for the corresponding slice of the time interval $[t_0,t_1]$. Consider an object U in the field of view of the event and LiDAR sensors, and a point $p \in PC^{(t_1)}$, belonging to object U. The object U (and along with it, the point p) might have undergone considerable motion and overlapped with several pixel positions $(x_1,y_1),(x_2,y_2),\ldots,(x_n,y_n)$ in the 2D space during the interval $[t_0,t_1]$, thereby scattering its features throughout the length and breadth of the event volume. Similarly, given a pixel location (x,y) in the event volume, there could be several objects $U_1,U_2,\ldots,U_K$ which would have overlapped with (x,y) at some point of time during the interval $[t_0,t_1]$. Hence, each pixel location in the 2D space contains features from potentially multiple objects.

Thus, our objective is to match parts of objects in the 3D space with their counterparts in a sequence of snapshots of the 2D space. Further, this matching needs to account for the possibility of many-to-many correspondences between points in the 3D space and pixel locations in the 2D space. These correspondences can possibly vary in strength, and we seek to make weighted decisions when fusing features from the 2D space into any point p, or when fusing features from the 3D space into any pixel location (x,y). We delegate the task of finding and quantifying the correspondences to a learnable attention module, called *Bi-AttFuse* (cf. Figure 5), details of which will be described next.

3.3.2 Bi-Directional Attentive Fusion Module

Query Encoder: The event features of shape $D_2 \times H_l \times W_l$ are encoded into query vectors Q_{ev}. This is accomplished using a 2D convolution module which accepts the original D_2 channels and outputs D' channels. The resulting query encoding is of the shape $D' \times H_l \times W_l$, which is subsequently flattened along the last two dimensions to $D' \times N_2$ where N_2 is the product of H_l and W_l.

Key Encoder: Point features of shape $D_1 \times N_1$ are encoded into key vectors K_{pc}. Here, a 1D convolution is used for representing the information in the original D_1 channels using D' channels. The resulting key encodings have the shape $D' \times N_1$.

Attention: Given queries Q_{ev} of shape $D' \times N_2$ and keys K_{pc} of shape $D' \times N_1$, we calculate the attention map as a matrix multiplication:

$$A = Q_{ev}^T\, K_{pc} \tag{1}$$

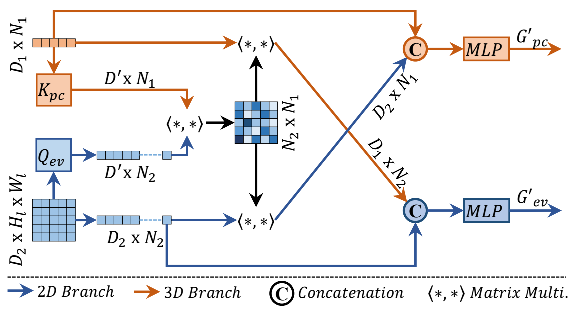


Figure 5: Our Bi-Directional Attentive Fusion Module (referred to as *Bi-AttFuse* in Figure 2).

Both the queries and the keys can be thought of as hidden encodings for the objects. Over time, queries and keys coming from the same object must learn to *find* each other using guidance from the attention map A. Given point features G_{pc} of shape $N_1 \times D_1$ and event features G_{ev} of shape $N_2 \times D_2$ (after the requisite re-shaping and transpose operations), we perform the following computations:

$$G'_{ev} = M_{2D}\left(\left[s(A)\; G_{pc},\; G_{ev}\right]\right) \tag{2}$$

$$G'_{pc} = M_{1D}\left(\left[s(A^T)\; G_{ev},\; G_{pc}\right]\right) \tag{3}$$

where s stands for the Softmax function over the last dimension, M_{2D}, M_{1D} refer respectively to a 2D and a 1D convolution, and $[A,B]$ stands for a concatenation operation between matrices A and B. The resulting features G'_{ev} and G'_{pc} have the same shapes as G_{ev} and G_{pc}, respectively. Hence, using the two branches of our fusion module described above, we compute point features which are aware of their corresponding event features, and vice versa.

3.4 Losses

In order to help the 2D and 3D branches of our model to learn through mutual enhancement, we formulate a loss which simultaneously sends supervisory feedback to both branches of the architecture. The losses have been developed considering n iterations of the Gated Recurrent Unit (GRU) for both branches.

2D Branch: Given optical flow estimates $\hat{F}^1_{2D}$, $\hat{F}^2_{2D}$, ..., $\hat{F}^n_{2D}$, the loss L_{2D} for the 2D branch is a supervised loss, defined as

$$L_{2D} = \sum_{k=1}^{n} w_k ||\hat{F}^k_{2D} - F_{2D}||_1 \tag{4}$$

where $w_1, w_2, \ldots, w_n$ are hyperparameters.

3D Branch: Due to the unavailability of scene flow ground truth in DSEC (Gehrig et al., 2021a), we follow FlowStep3D (Kittenplon et al., 2021) and use the Chamfer Loss, a self-supervised loss which attempts to minimize the distance between the source point cloud warped by the estimated scene flow, and the target point cloud. Given a source point cloud S (also referred to as $PC^{(t_1)}$ in the rest of this section) and flow estimates $\hat{F}^1_{3D}, \hat{F}^2_{3D}, ..., \hat{F}^n_{3D}$, the Chamfer loss can be defined as:

$$L_{3D} = \sum_{k=1}^{n} w_k \left(\sum_{p \in S} \min_{q \in T_k} ||p-q||_2^2 + \sum_{q \in T_k} \min_{p \in S} ||q-p||_2^2 \right) \tag{5}$$

where

$$T_k = S + \hat{F}^k_{3D} \tag{6}$$

Total Loss: The total loss is a weighted sum of the 3D and 2D losses:

$$L = \beta_{2D} L_{2D} + \beta_{3D} L_{3D} \tag{7}$$

where β_{2D} and β_{3D} are hyperparameters.

4 EXPERIMENTS AND RESULTS

4.1 Data Set

DSEC (Gehrig et al., 2021a) is a newly released **real-world** data set (unlike popular scene flow data sets like FlyingThings3D (Mayer et al., 2016), which are synthetic) and contains driving scenes under different illumination conditions and at varying levels of difficulty. To the best of our knowledge, it is the only existing data set which provides event streams along with LiDAR scans. However, DSEC lacks scene flow labels and only provides optical flow ground truth. For evaluation purposes, we use camera intrinsics to project the estimated scene flow vectors into their respective 2D (optical flow) counterparts. Since the latter is an algebraic transformation of the former, we posit that the accuracy of the optical flow estimations is contingent upon that of the scene flow estimations.

Although DSEC has a total of 53 scenes, the training split contains only 18 scenes for which the ground truth optical flow has been made available; out of these, we use 11 scenes (5357 examples) for training, 2 scenes (457 examples) for validation and the remaining 5 scenes (2356 examples) are held out for testing our trained models.

Pre-processing and post-processing: In all our experiments, we sample 8192 points from each point cloud. Following FlowStep3D (Kittenplon et al., 2021), we do not consider points with depth greater than 35 m for evaluation. We train and evaluate our models under two modes: **with ground points (w/ ground)** and **without ground points (w/o ground)**, wherein we include / exclude the ground points from the input point clouds, respectively. In both cases, we eventually exclude the ground points during evaluation. The motivation for this approach is to clearly observe how the ground points affect the overall scene flow estimation accuracy for other, more important parts of the scene, while ignoring the quality of the flow estimates for the less important ground points themselves. In other words, the ground points are either fed into the model (**w/ ground**) or not (**w/o ground**), but after obtaining the predictions of the model, all ground points are **always** removed before computing the evaluation metrics. However, as an important exception, we do not remove the ground while evaluating the results of the 2D branch of our model.

4.2 Evaluation Metrics

Since the data set of DSEC (Gehrig et al., 2021a) used by us contains only optical flow ground truth, scene flow predictions are always projected into optical flow predictions for quantitative evaluation against the optical flow ground truth values for the corresponding pixel locations. We use the following metrics:

- End Point Error (**EPE**): Average across all points (or pixels) of the L2 norm of the difference between the predicted and ground truth optical flow values.
- Outliers (**OUT**): Percentage of points (or pixels) for which the absolute EPE is greater than 3 pixels and the relative EPE is greater than 5%.
- Accuracy (**ACC**): Percentage of points (or pixels) for which the absolute EPE is less than 3 pixels or the relative EPE is less than 10%.

4.3 Implementation Details

The learning rate is set to 10^{-4}, and is multiplied with 0.3 at epochs 50 and 70. We use 4 refinement steps of the Gated Recurrent Unit (GRU) in both branches and set the loss weights (cf. Section 3.4) as $w_1 = 0.1$, $w_2 = 0.2$, $w_3 = 0.3$, and $w_4 = 0.4$. Further, we set both the hyperparameters β_{3D} and β_{2D} to 0.5. All models have been trained on the entire training set for 90 epochs (both with ground and without ground). As an exception, E-RAFT (Gehrig et al., 2021b) has

Table 1: Evaluation Results on DSEC (Gehrig et al., 2021a) (2D Branch); trained and evaluated **with** ground.

Model	**EPE ↓**	**OUT ↓**	**ACC ↑**
	[px]	**[%]**	**[%]**
E-RAFT	**1.149**	**5.75**	**94.69**
Ours (2D)	1.179	6.16	94.33

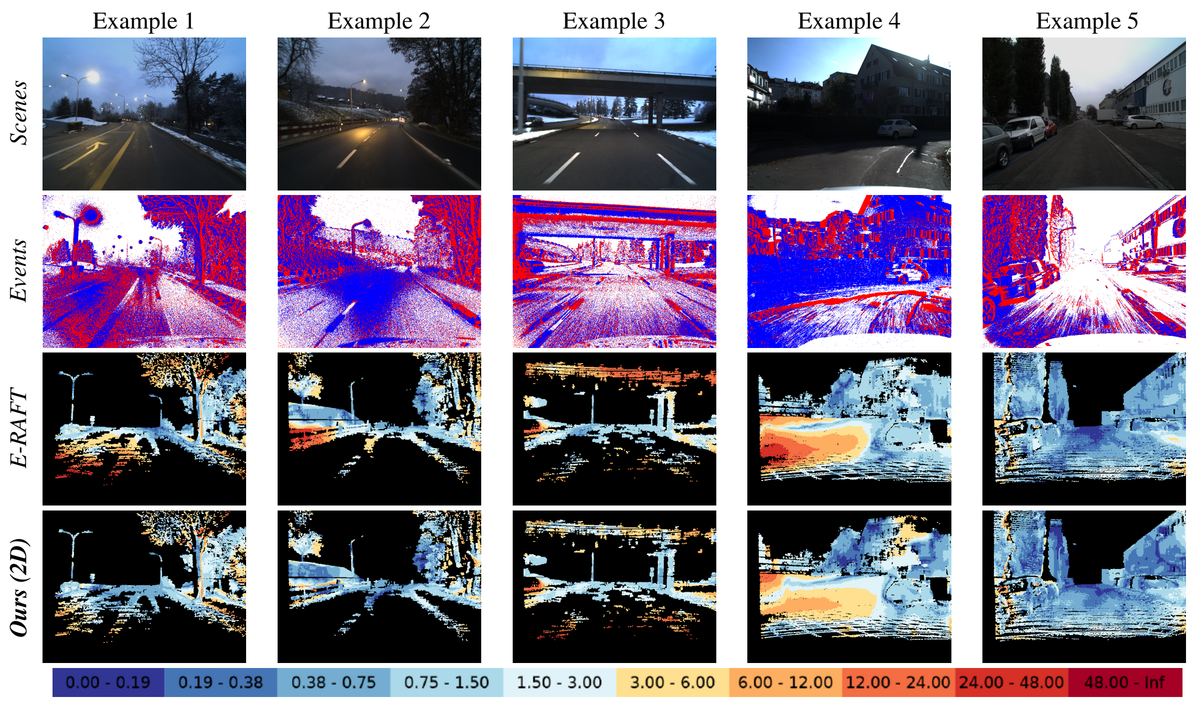


Figure 6: Qualitative Results on five examples selected from DSEC (Gehrig et al., 2021a). In each column, the first row is the RGB frame (**solely for reference purposes; not fed into the model**), the second row is the event stream represented as an image, the third and fourth rows are the optical flow estimation error maps for E-RAFT (Gehrig et al., 2021b) and our EvLiDAR-Flow, respectively. The errors have been color coded; the color scale for this coding has been provided above. Black pixels in the error maps denote locations where valid optical flow ground truth was unavailable. Our model performs largely on par with E-RAFT, while eliminating extremely high error values (reddish regions in the error maps of E-RAFT).

been trained (with ground) for 40 epochs, following the original paper E-RAFT. The threshold for removing the ground points, wherever applicable, is set to 0.4.

4.4 Results

E-RAFT (Gehrig et al., 2021b) and FlowStep3D (Kittenplon et al., 2021) are state-of-the-art works in the optical flow and scene flow domains, respectively. We demonstrate that our fusion module acts as a bridge for sharing information between the optical flow and scene flow branches based on these two works, and thereby enhances the quality of the corresponding flow predictions in comparison to the individual architectures (without fusion).

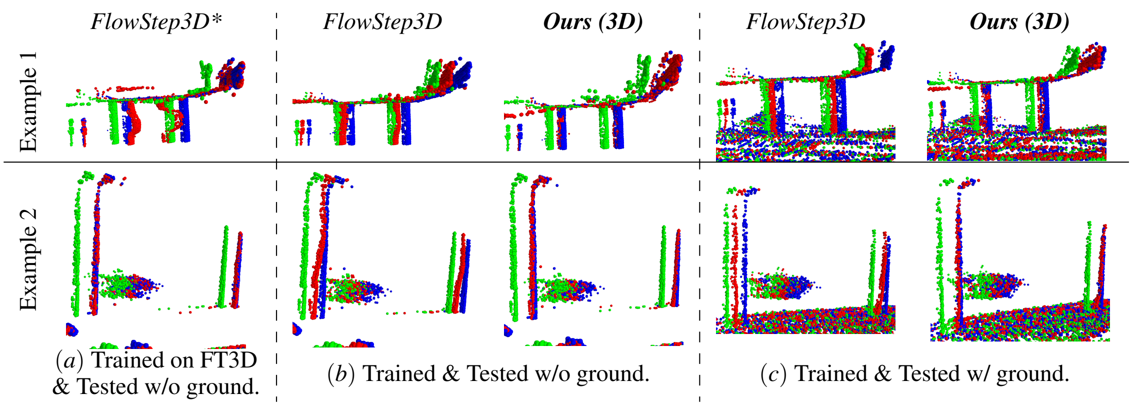
4.4.1 2D Branch

For fair comparison, we train E-RAFT (Gehrig et al., 2021b) on the same training set used for training our EvLiDAR-Flow. As can be seen in Table 1, our model performs at par with E-RAFT on the quantitative evaluation metrics. This indicates that the optical flow branch might need more data in order to take full advantage of the attention module and outperform E-RAFT. In general, attention-based models are trained on relatively huge data sets (Caron et al., 2021; Girdhar et al., 2019). With the limited number of training examples and variability available to us in this work, it is possible that the attention module could not generalize enough for enhancing the optical flow branch, which operates on noisy event data. We present some qualitative results in Figure 6.

4.4.2 3D Branch

First, we establish a baseline by testing the pre-trained model of FlowStep3D* (Kittenplon et al., 2021) (trained on FlyingThings3D (Mayer et al., 2016)). Subsequently, for fair comparison, we also train both FlowStep3D and our EvLiDAR-Flow on DSEC (Gehrig et al., 2021a), under two different settings each. The quantitative results have been presented in Table 2.

Training both models w/o ground: We observe that the performance of both FlowStep3D and our EvLiDAR-Flow deteriorates across all metrics upon



FlowStep3D*
FlowStep3D
Ours (3D)
FlowStep3D
Ours (3D)
Example 1
Example 2
(a) Trained on FT3D & Tested w/o ground.
(b) Trained & Tested w/o ground.
(c) Trained & Tested w/ ground.

Figure 7: A qualitative evaluation of our model on two examples taken from the DSEC (Gehrig et al., 2021a) data set. The points are color-coded: source points are colored green, target points are colored blue and source points warped by the predicted scene flow are colored red. The pre-trained model of FlowStep3D* (first column) produces some impressive estimates, because of being trained on the extensive FlyingThings3D (FT3D) (Mayer et al., 2016) data set. However, this model fails for planar regions (Example 1). Trained and evaluated on the DSEC (Gehrig et al., 2021a) data set, FlowStep3D (Kittenplon et al., 2021) produces poor predictions, which worsen further with the introduction of the ground points in the input (second and fourth columns, respectively). In comparison, our EvLiDAR-Flow produces highly accurate scene flow estimates, both without and with ground points in the inputs (third and fifth columns, respectively).

Table 2: **Evaluation Results on DSEC (Gehrig et al., 2021a) (3D Branch)**. The third column specifies whether ground points were *included* or *excluded* during training. Each model has been evaluated once without feeding ground points (**w/o ground**) into the model (fourth, fifth and sixth columns), and then with ground points (**w/ ground**) fed into the model (seventh, eighth and ninth columns). **However, ground points have never been considered during the evaluation itself.** The first row refers to the pre-trained FlowStep3D* (Kittenplon et al., 2021) architecture (trained on FlyingThings3D (Mayer et al., 2016)).

Model	**Input**	**Ground Surface [Training]**	**Input w/o Ground**			**Input w/ Ground**		
			EPE ↓ [px]	**OUT ↓ [%]**	**ACC ↑ [%]**	**EPE ↓ [px]**	**OUT ↓ [%]**	**ACC ↑ [%]**
FlowStep3D*	Points	-	8.357	45.64	54.45	8.299	53.40	46.67
FlowStep3D	Points	w/o ground	7.977	62.98	37.11	8.012	68.57	31.50
Ours (3D)	Points + Events	w/o ground	**5.869**	**41.76**	**58.38**	**5.930**	**44.69**	**55.41**
FlowStep3D	Points	w/ ground	8.141	64.76	35.32	7.438	63.51	36.58
Ours (3D)	Points + Events	w/ ground	**5.433**	**45.58**	**54.49**	**5.136**	**43.26**	**56.84**

re-introducing the ground points into the input. It is worth noting that while for FlowStep3D, the outliers increase by 5.59 %, for our model, they only increase by 2.93 % (second and third rows, respectively) with the introduction of the ground points. Moreover, under all settings, the absolute values of the metrics are significantly better for our architecture in comparison to FlowStep3D.

Training both models w/ ground: We posit that models trained with ground points will learn to estimate relatively accurate flow in the presence of ground points. Our EvLiDAR-Flow continues to significantly outperform FlowStep3D in this setting.

The best EPE that our model achieves is 5.136 when trained with ground, while the corresponding EPE for FlowStep3D is 7.438. This gap between the two models is clear when we consider that our model makes use of rich event information in parallel with point clouds, and is therefore significantly more robust to the presence of unstructured areas. It is noteworthy that the attention mechanism could bring this improvement despite the small size of the training set. Through the qualitative results in Figure 7 and the corresponding caption, we further validate the performance of our fusion module.

4.5 Ablation Studies

Through ablation experiments (cf. Table 3), we validate the superiority of bi-directional fusion over uni-directional fusion. For the $2D \rightarrow 3D$ setting, we only fuse event features into point features (cf. Equation (3)) and the original event features (without fusion) are used for the event branch. For

Table 3: Ablation Experiments.

Fusion	2D Branch			3D Branch		
	EPE ↓ [px]	OUT ↓ [%]	ACC ↑ [%]	EPE ↓ [px]	OUT ↓ [%]	ACC ↑ [%]
Attention (2D → 3D)	1.270	7.01	93.76	6.860	58.47	41.64
Attention (2D ← 3D)	1.284	7.24	93.59	8.450	70.60	29.49
Attention (2D ↔ 3D)	**1.179**	**6.16**	**94.33**	**5.136**	**43.26**	**56.84**

the $3D \rightarrow 2D$ setting, we only fuse point features into event features (cf. Equation (2)) and the original point features (without fusion) are used for the point branch. All models were trained with ground and tested with ground, **excluding ground points from consideration during evaluation of the 3D branch**, and using identical loss functions (cf. Section 3.4). Bi-directional fusion significantly improves both branches in comparison to uni-directional fusion in either direction, hence suggesting that both modalities significantly enhance each other when they are both "aware" of each other.

5 CONCLUSION

In this paper, we propose, for the first time according to our knowledge, the fusion of events and point clouds for scene flow estimation using our deep neural network architecture EvLiDAR-Flow. In order to facilitate this fusion, we propose a learnable attention module. Our model takes the advantage of rich event information to overcome the difficulty of robustly estimating accurate scene flow in the presence of unstructured areas in the scene, where LiDAR-only methods often suffer. Provided that a larger data set can be developed in the future, there exist some promising possibilities for future work in the area of fusion between point clouds and event streams (e.g. multiple levels of fusion for even more robust estimation).

ACKNOWLEDGEMENTS

This work was partially funded by the Federal Ministry of Education and Research Germany in the funding program Photonics Research Germany under the project FUMOS (13N16302).

REFERENCES

Battrawy, R., Schuster, R., Wasenmüller, O., Rao, Q., and Stricker, D. (2019). LiDAR-Flow: Dense Scene Flow Estimation from Sparse LiDAR and Stereo Images. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2, 3

Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., and Joulin, A. (2021). Emerging Properties in Self-Supervised Vision Transformers. In *IEEE/CVF International Conference on Computer Vision (ICCV)*. 7

Dosovitskiy, A., Fischer, P., Ilg, E., Hausser, P., Hazirbas, C., Golkov, V., Van Der Smagt, P., Cremers, D., and Brox, T. (2015). FlowNet: Learning Optical Flow with Convolutional Networks. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*. 2

Gallego, G., Delbrück, T., Orchard, G., Bartolozzi, C., Taba, B., Censi, A., Leutenegger, S., Davison, A. J., Conradt, J., Daniilidis, K., and Scaramuzza, D. (2020). Event-based Vision: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*. 2

Gehrig, M., Aarents, W., Gehrig, D., and Scaramuzza, D. (2021a). DSEC: A Stereo Event Camera Dataset for Driving Scenarios. *IEEE Robotics and Automation Letters (RA-L)*. 2, 5, 6, 7, 8

Gehrig, M., Millhäusler, M., Gehrig, D., and Scaramuzza, D. (2021b). E-RAFT: Dense Optical Flow from Event Cameras. In *International Conference on 3D Vision (3DV)*. 1, 2, 4, 6, 7

Girdhar, R., Carreira, J., Doersch, C., and Zisserman, A. (2019). Video Action Transformer Network. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 7

Gu, X., Wang, Y., Wu, C., Lee, Y. J., and Wang, P. (2019). HPLFlowNet: Hierarchical Permutohedral Lattice FlowNet for Scene Flow Estimation on Large-scale Point Clouds. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 1, 3

Hu, L., Zhao, R., Ding, Z., Ma, L., Shi, B., Xiong, R., and Huang, T. (2022). Optical Flow Estimation for Spiking Camera. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2

Hui, T.-W., Tang, X., and Loy, C. C. (2018). LiteFlowNet: A Lightweight Convolutional Neural Network for Optical Flow Estimation. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*. 2

Ilg, E., Mayer, N., Saikia, T., Keuper, M., Dosovitskiy, A., and Brox, T. (2017). FlowNet 2.0: Evolution of Optical Flow Estimation with Deep Networks. In *IEEE In-*

ternational Conference on Computer Vision and Pattern Recognition (CVPR). 2

Kittenplon, Y., Eldar, Y. C., and Raviv, D. (2021). FlowStep3D: Model Unrolling for Self-Supervised Scene Flow Estimation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 1, 2, 3, 4, 6, 7, 8

Lee, C., Kosta, A. K., and Roy, K. (2021). Fusion-FlowNet: Energy-Efficient Optical Flow Estimation using Sensor Fusion and Deep Fused Spiking-Analog Network Architectures. *arXiv preprint arXiv:2103.10592*. 2

Lee, C., Kosta, A. K., Zhu, A. Z., Chaney, K., Daniilidis, K., and Roy, K. (2020). Spike-FlowNet: Event-based Optical Flow Estimation with Energy-Efficient Hybrid Neural Networks. In *European Conference on Computer Vision (ECCV)*. 2

Liu, H., Lu, T., Xu, Y., Liu, J., Li, W., and Chen, L. (2022). CamLiFlow: Bidirectional Camera-LiDAR Fusion for Joint Optical Flow and Scene Flow Estimation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2, 3

Liu, X., Qi, C. R., and Guibas, L. J. (2019). FlowNet3D: Learning Scene Flow in 3D Point Clouds. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 1, 3, 4

Low, W. F., Gao, Z., Xiang, C., and Ramesh, B. (2020). SOFEA: A Non-Iterative and Robust Optical Flow Estimation Algorithm for Dynamic Vision Sensors. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 1, 2

Low, W. F., Sonthalia, A., Gao, Z., van Schaik, A., and Ramesh, B. (2021). Superevents: Towards Native Semantic Segmentation for Event-based Cameras. In *International Conference on Neuromorphic Systems (ICONS)*. 2

Ma, W.-C., Wang, S., Hu, R., Xiong, Y., and Urtasun, R. (2019). Deep Rigid Instance Scene Flow. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 1

Mayer, N., Ilg, E., Hausser, P., Fischer, P., Cremers, D., Dosovitskiy, A., and Brox, T. (2016). ALarge Dataset to Train Convolutional Networks for Disparity, Optical Flow, and Scene Flow Estimation. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*. 6, 7, 8

Menze, M. and Geiger, A. (2015). Object Scene Flow for Autonomous Vehicles. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*. 1

Puy, G., Boulch, A., and Marlet, R. (2020). FLOT: Scene Flow on Point Clouds Guided by Optimal Transport. In *European Conference on Computer Vision (ECCV)*. 1, 3

Qi, C. R., Yi, L., Su, H., and Guibas, L. J. (2017). PointNet++: Deep Hierarchical Feature Learning on Point Sets in a Metric Space. *Advances in Neural Information Processing Systems (NeurIPS)*. 3

Rishav, R., Battrawy, R., Schuster, R., Wasenmüller, O., and Stricker, D. (2020). DeepLiDARFlow: A Deep Learning Architecture For Scene Flow Estimation Using Monocular Camera and Sparse LiDAR. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2, 3

Saxena, R., Schuster, R., Wasenmüller, O., and Stricker, D. (2019). PWOC-3D: Deep Occlusion-Aware End-to-End Scene Flow Estimation. *IEEE International Conference on Intelligent Vehicles Symposium (IV)*. 1

Schuster, R., Wasenmüller, O., Kuschk, G., Bailer, C., and Stricker, D. (2018). SceneFlowFields: Dense Interpolation of Sparse Scene Flow Correspondences. In *IEEE Winter Conference on Applications of Computer Vision (WACV)*. 1

Su, H., Jampani, V., Sun, D., Maji, S., Kalogerakis, E., Yang, M.-H., and Kautz, J. (2018). SPLATNet: Sparse Lattice Networks for Point Cloud Processing. In *IEEE conference on computer vision and pattern recognition (CVPR)*. 3

Sun, D., Yang, X., Liu, M.-Y., and Kautz, J. (2018). PWC-Net: CNNs for Optical Flow Using Pyramid, Warping, and Cost Volume. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*. 2

Teed, Z. and Deng, J. (2020). RAFT: Recurrent All-Pairs Field Transforms for Optical Flow. In *European conference on computer vision (ECCV)*. 2, 3

Teed, Z. and Deng, J. (2021). RAFT-3D: Scene Flow using Rigid-Motion Embeddings. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2

Wei, Y., Wang, Z., Rao, Y., Lu, J., and Zhou, J. (2021). PV-RAFT: Point-Voxel Correlation Fields for Scene Flow Estimation of Point Clouds. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 1, 3

Weinzaepfel, P., Revaud, J., Harchaoui, Z., and Schmid, C. (2013). DeepFlow: Large Displacement Optical Flow with Deep Matching. In *IEEE International Conference on Computer Vision (ICCV)*. 2

Wu, W., Qi, Z., and Fuxin, L. (2019). PointConv: Deep Convolutional Networks on 3D Point Clouds. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 3

Wu, W., Wang, Z. Y., Li, Z., Liu, W., and Fuxin, L. (2020). PointPWC-Net: Cost Volume on Point Clouds for (Self-) Supervised Scene Flow Estimation. In *European Conference on Computer Vision (ECCV)*. 1, 3

Xu, J., Ranftl, R., and Koltun, V. (2017). Accurate Optical Flow via Direct Cost Volume Processing. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2

Zhu, A. Z., Yuan, L., Chaney, K., and Daniilidis, K. (2018). EV-FlowNet: Self-Supervised Optical Flow Estimation for Event-based Cameras. In *Proceedings of Robotics: Science and Systems (RSS)*. 1, 2

Zhu, A. Z., Yuan, L., Chaney, K., and Daniilidis, K. (2019). Unsupervised Event-based Learning of Optical Flow, Depth, and Egomotion. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*. 4, 5