Learning to Estimate Two Dense Depths from LiDAR and Event Data[⋆]

Vincent Brebion[0000−0001−7429−0547], Julien Moreau[0000−0001−5008−9232], and Franck Davoine[0000−0002−8587−6997]

Heudiasyc (Heuristics and Diagnosis of Complex Systems) Laboratory, CNRS, Université de technologie de Compiègne (UTC), 60319 Compiègne Cedex, France {vincent.brebion,julien.moreau,franck.davoine}@hds.utc.fr

Abstract. Event cameras do not produce images, but rather a continuous flow of events, which encode changes of illumination for each pixel independently and asynchronously. While they output temporally rich information, they lack any depth information which could facilitate their use with other sensors. LiDARs can provide this depth information, but are by nature very sparse, which makes the depth-to-event association more complex. Furthermore, as events represent changes of illumination, they might also represent changes of depth; associating them with a single depth is therefore inadequate. In this work, we propose to address these issues by fusing information from an event camera and a LiDAR using a learning-based approach to estimate accurate dense depth maps. To solve the "potential change of depth" problem, we propose here to estimate two depth maps at each step: one "before" the events happen, and one "after" the events happen. We further propose to use this pair of depths to compute a depth difference for each event, to give them more context. We train and evaluate our network, ALED, on both synthetic and real driving sequences, and show that it is able to predict dense depths with an error reduction of up to 61% compared to the current state of the art. We also demonstrate the quality of our 2-depths-to-event association, and the usefulness of the depth difference information. Finally, we release SLED, a novel synthetic dataset comprising events, LiDAR point clouds, RGB images, and dense depth maps.

Keywords: Sensor fusion · Machine learning · Dense depth estimation.

1 Introduction

Rather than accumulating light to create images, event cameras perceive changes of illumination for each pixel independently and asynchronously. Thanks to their high temporal resolution (in the order of the µs) and high dynamic range, event cameras are a sensor of choice for dynamic applications in complex environments

---

(fast motions, extreme illumination conditions), where traditional cameras reach their limits.

LiDAR sensors offer accurate but sparse 3D information of their surrounding environment. They are a key component for autonomous navigation, helping to solve multiple problems, e.g., obstacle detection and tracking, SLAM, etc. Yet, their sparsity often constitutes a limiting factor. While 64- or 128-channel LiDARs are starting to be commercialized, they come at a significantly high cost, and are still not as dense as cameras.

In this work, we focus on the fusion of LiDAR and event data, which we consider as a dual problem: (1) LiDAR depths densification and (2) events-depths association. Regarding problem (1), we are interested in densifying the LiDAR data using the events as a guide. As a result, dense depth maps are obtained, which allow for a dense 3D perception of the observed scene. As for problem (2), we are interested in associating a depth to each event. By doing so, each event can be projected in 3D, and then even be backprojected in 2D in another vision sensor. For a fully calibrated and synced setup, this process would allow for the superimposition of events and RGB images, a task which is only possible at the moment through the use of specific low-resolution frame+events cameras like the DAVIS240C [2].

Estimating dense depth maps from sparse LiDAR data is a well-regarded problem as it solves the sparsity drawback of the LiDAR while keeping its metric scale. However, using events to densify depth maps (i.e., problem (1)) might be inaccurately seen as a task that inherently includes problem (2), as corresponding depths for the events could be taken from the dense depth map. We argue in this work that, as each event represents a change in illumination, it might also represent a change in depth. As such, two depths should be associated to each event, and we will therefore compute two depth maps: one before the events happen, and one after they happen.

As an answer to these issues, we propose in this work a learning-based fusion method for estimating pairs of dense depth maps from events and sparse LiDAR data. Our main contributions are as follows:

– We propose to revise the principle of associating a single depth to each event to rather estimate two depths: one "before" the event happens, and one "after". Following this, we introduce the notion of "depth change map", to give more context to each event.
– We propose a novel convolutional network, the ALED (Asynchronous LiDAR and Events Depths densification) network, able to fuse asynchronous events and LiDAR data, and to estimate the two dense depth maps from them, while surpassing state-of-the-art accuracy.
– We finally build and share a high-definition simulated dataset, the SLED (Synthetic LiDAR Events Depths) dataset, used as part of the training of the network and its evaluation.

If the reader is interested, supplementary material, the SLED dataset, source codes, as well as videos showcasing results on both simulated and real data are all available at https://vbrebion.github.io/ALED.

2   Related Work

2.1   LiDAR Densification

LiDAR sensors only produce sparse point clouds, which is challenging for numerous applications (3D reconstruction, object detection, SLAM, etc). As a consequence, LiDAR depth completion is a subject that has been widely studied in the literature.

Some authors try to obtain dense depth maps while only relying on the sparse data from the LiDAR. These methods either use machine learning [4,14,39] or traditional image processing operations [19].

The most successful approaches use a secondary modality as a guide for the densification process. While most of these approaches employ a RGB camera as the secondary sensor [7,14,15,42], other authors have proposed using alternative modalities, such as stereo cameras [23] or more recently event cameras [5].

2.2   Fusion of Events and Other Modalities

Due to their relative youth, the literature on the fusion of data from event cameras with other sensors is quite sparse.

Most of the investigations focused on the fusion of events and frames, thanks to sensors offering both modalities like the DAVIS camera [2]. These works include frame interpolation and deblurring [26,27,31], feature tracking [8,20], object detection [3,16,38], or even steering prediction [13].

In the past few years, a few authors have started investigating the fusion of events and LiDAR data. Explored issues include calibration [35,36], and very recently, point clouds enhancement with events [21] and LiDAR densification [5].

2.3   Depth Estimations with Events

Several approaches have been proposed in order to estimate sparse or dense depth maps by using a single event camera. Kim et al. [17] used probabilistic filters to simultaneously estimate the motion of the camera, reconstruct a log intensity image of the observed scene, and construct a sparse inverse depth map. Zhu et al. [44] used a convolutional network to jointly predict depth and egomotion, by trying to minimize the amount of motion blur in the accumulated events. Hidalgo-Carrió et al. [12] were the first to estimate dense depth maps from a monocular event camera, through the use of a recurrent convolutional network.

In parallel, other authors have advocated for the use of a secondary sensor to help the depth estimation. Schraml et al. [32,33] and Nam et al. [25] used two event cameras, and estimated depths by creating images of accumulated events for each camera and applying stereo matching. While [32,33] used traditional model-based approaches, [25] entirely relied on learning-based networks: an attention-based network to construct detailed events representations, then a convolutional network for depth map inference. Other authors have also combined the event camera with a RGB sensor; Gehrig et al. [9] for instance designed

a recurrent network to fuse asynchronous data and estimate dense depths from them. Finally, some authors have also used depth sensors in direct combination with event cameras. Weikersdorfer et al. [41] used an RGB-D camera to obtain dense depths, and used the depth-augmented events to perform SLAM. Li et al. [21] used a LiDAR sensor to associate a depth to each event through the use of a Voronoi diagram and a set of heuritic rules. Cui et al. [5] also employed a LiDAR, to derive dense depth maps by using 3D geometric information.

3   Depth Change Map: Two Depths per Event

The fusion of depths and events is a problem that can have two different goals: (1) obtaining dense depth maps from events and sparse LiDAR data, or (2) determining a depth for each event. While problem (1) can be interpreted as a LiDAR densification method guided by the events, we argue here for problem (2) that associating a single depth to an event is inadequate.

By definition, an event represents a significant change in illumination observed by a given pixel. Under motion, observed events can either originate from (a) texture changes inside an object; or from (b) the contour of an object. In case (a), associating a single depth to these events can be coherent, as depth inside an object should be subject to little variation. However, doing so in case (b) is erroneous, as the events are likely to denote also a depth change.

Instead, we propose to estimate two dense depth maps: one before the events happen, which we will denote D_{bf} in the rest of this article, and one after the events happen, which we will denote D_{af}. We can then formulate the depth change map as $D_{af} - D_{bf}$ and compare the two depths d_{bf} and d_{af} for each pixel. Three meaningful cases can be distinguished:

1. $d_{af} - d_{bf} \approx 0$: the pixel is located in an area where depths do not vary much, i.e., inside an object;
2. $d_{af} - d_{bf} \gg 0$: the pixel was at the edge of an object, and is now on an object further away;
3. $d_{af} - d_{bf} \ll 0$: the pixel was located on a far object, and is now at the edge of a closer object.

The depth difference information given by the depth change map can especially help events processing to differentiate real objects from artifacts such as shadows and even noise. An illustration of some possibilities offered by the depth change map on events is given in Fig. 1. Other applications could also take advantage of the pair of depth maps D_{bf} and D_{af}: ego-motion and speed estimation, objects clustering, scene flow, etc.

4   Method

4.1   The ALED Network

Inspired by the Recurrent Asynchronous Multimodal Network (RAMNet) architecture of Gehrig et al. [9] for RGB and events fusion, we propose here a fully
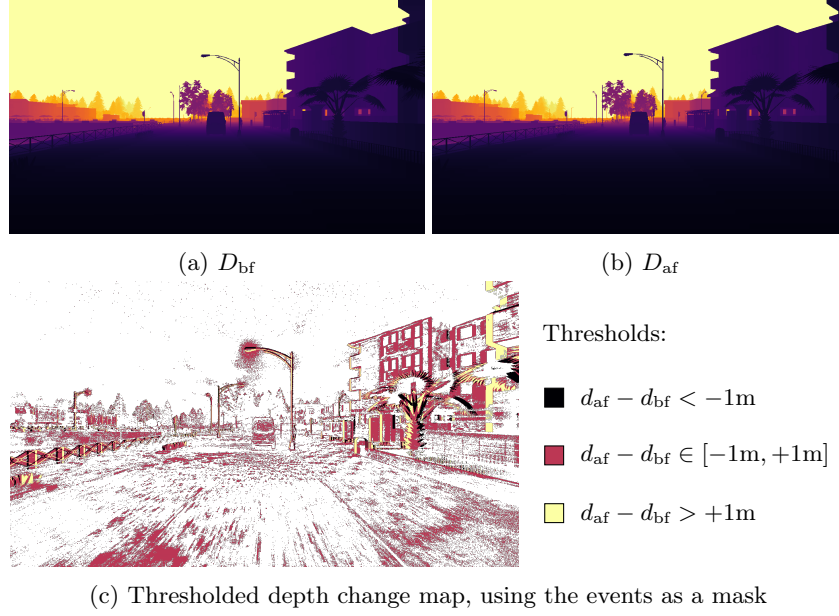
(a) D_{bf} (b) D_{af}

Thresholds:

■ $d_{\mathrm{af}} - d_{\mathrm{bf}} < -1\mathrm{m}$

■ $d_{\mathrm{af}} - d_{\mathrm{bf}} \in [-1\mathrm{m}, +1\mathrm{m}]$

□ $d_{\mathrm{af}} - d_{\mathrm{bf}} > +1\mathrm{m}$

(c) Thresholded depth change map, using the events as a mask

Fig. 1: Example of the importance of the depth change map for each event on the "Town01_00" sequence from our SLED dataset. Notice how simple thresholds on this depth difference help distinguishing the events linked to the contour of real objects from the events corresponding to the texture of the road, the halo from the street lamp, or even the noisy events in the sky.

convolutional recurrent network to estimate dense depths from asynchronous LiDAR and events data. We call it the ALED Net, for Asynchronous LiDAR and Events Depths densification network.

Our network can be decomposed in two main parts: an encoder, tasked with fusing asynchronous events and LiDAR features at different scales, and a decoder, tasked with interpreting the fused features for estimating dense depths.

In its encoder part, illustrated in Fig. 2, the LiDAR and events inputs are fed independently. Both of them go through an encoding head, computing a first feature map of 32 channels while keeping their original height and width. Convolutional encoders (in the form of ResNet Basic Blocks [11]) are then used to compute feature maps at scales 1/2, 1/4, and 1/8, doubling the number of channels every time. Each of these feature maps is then used as the input of convolutional gated recurrent unit (convGRU) blocks [34], updating its corresponding state. Since these states are shared between the LiDAR and events encoders, both parts of the network can update them asynchronously.

In its decoder part, illustrated in Fig. 3, the convGRU state at the lowest scale first goes through two residual blocks. Then, for each following scale, the decoded feature map from the previous scale is upscaled by using convex upsampling [37]. While a simple bilinear upsampling was used in RAMNet [9], convex upsampling
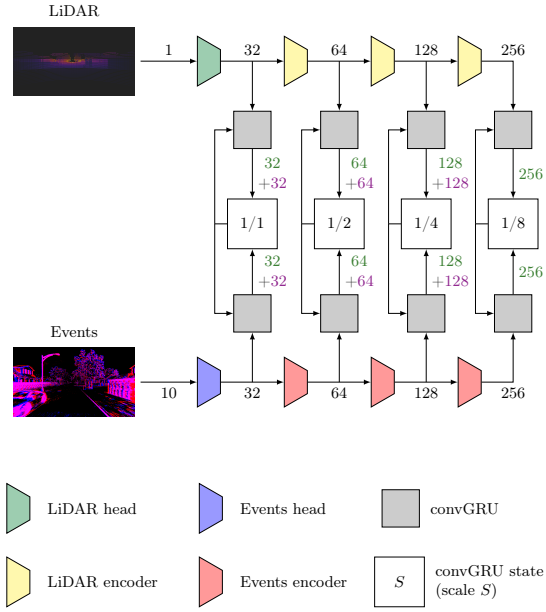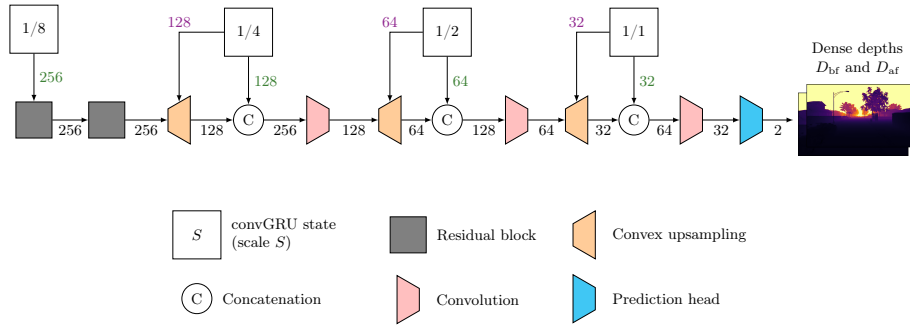
Fig. 2: The encoder part of the network.

Fig. 3: The decoder part of the network.

allows our network to learn how to upscale features from a lower scale, using information from a higher scale. We propose to design the convGRU such that the first half of its state (in purple in Fig. 2 and 3) guides the convex upsampling. Fusion of the upsampled decoded features and the state from the current scale is then performed by concatenating the output of the convex upsampling block with the remaining half of the convGRU state (in green in Fig. 2 and 3), and by applying a convolution to reduce the number of channels. After the last scale, a prediction head is used to obtain the two final depth maps, D_{bf} and D_{af}, in the form of a two-channel tensor, at the same full resolution as the events input.

Regarding the implementation, both encoding heads use a kernel size of 5. LiDAR and events encoders use a kernel size of 5, with stride 2. Both the convGRU and residual blocks use a kernel size of 3. The convolutions in the convex upsampling blocks use a kernel size of 5, while the convolutions following the concatenations use a kernel size of 1. Finally, the prediction layer also uses a kernel size of 1. Convolutions are followed by a PReLU activation function [10], and instance normalization is used in the ResNet encoders as proposed by Pan et al. [28]. In total, the network contains 26 millions of trainable parameters.

4.2   Data Representation

Events We use Discretized Event Volumes [44] as the input representation for the events. We follow the formulation of Perot et al. [29], where the Discretized Event Volume V for input events $\{e_i = (x_i, y_i, p_i, t_i)\}_{i=1}^N$ is described as:

$$V_{t,p,y,x} = \sum_{e_i, x_i=x, y_i=y, p_i=p} \max(0, 1 - |t - t_i^*|) \tag{1}$$

$$t_i^* = (B - 1)\frac{t_i - t_0}{t_N - t_0} \tag{2}$$

where (x, y) is the position of the event, t its timestamp, and p its polarity.

In our experiments, we set $B = 5$ bins, and concatenate the negative and positive polarity bins along the first dimension, resulting in a tensor of shape $(10, H, W)$.

LiDAR and Depths LiDAR data is fed to the network as a 1-channel depth image. To do so, each LiDAR point cloud is projected onto the image plane of the event camera. Pixels where one or more LiDAR points fall into are given as value the lowest depth. Pixels without any LiDAR point are given a value of 0. For an easier learning, the LiDAR projection and ground truth images are normalized between 0 and 1 based on the maximum LiDAR range (200m in the case of our synthetic SLED dataset, 100m in the case of the MVSEC dataset [43]).

4.3   Loss functions

To train our network, we combine the use of two losses: a pixel-wise ℓ_1 loss $\mathcal{L}_{\mathrm{pw}}$, and a multiscale gradient matching loss $\mathcal{L}_{\mathrm{msg}}$.

The pixel-wise ℓ_1 loss operates as the main supervision loss, applied on both the "before" and the "after" depth maps, and is defined as follows:

$$\mathcal{L}_{\mathrm{pw}} = \sum_{x,y} \left\| D(x,y) - \hat{D}(x,y) \right\|_1 \tag{3}$$

where D and $\hat{D}$ are the respectively the estimated and ground truth depth maps.

However, when supervised by the ℓ_1 loss alone, the network tends to produce blurry and non-smooth depth images. To solve this issue, we use here a multiscale gradient matching loss inspired by [40], also applied on both the "before" and the "after" depth maps, and defined as

$$\mathcal{L}_{\mathrm{msg}} = \sum_{h \in \{1,2,4,8,16\}} \sum_{x,y} \left\| \mathbf{g}[D](x,y,h) - \mathbf{g}[\hat{D}](x,y,h) \right\|_2 \tag{4}$$

with the discrete gradient function $\mathbf{g}$ of an image f defined as

$$\mathbf{g}[f](x,y,h) = (f(x+h,y) - f(x,y); f(x,y+h) - f(x,y))^T \tag{5}$$

This loss helps regulating the depth results, by making depth discontinuities more prominent, and by smoothing homogeneous regions.

Our total loss $\mathcal{L}$ for a sequence of length T is finally defined as

$$\mathcal{L} = \sum_{t=0}^{T} \sum_{\mathrm{bf,af}} (\mathcal{L}_{\mathrm{pw}}^t + \alpha \mathcal{L}_{\mathrm{msg}}^t) \tag{6}$$

where α is a weight parameter for the multiscale gradient matching loss.

In our experiments, we observed that giving too much importance to the multiscale gradient matching loss early in the training makes the network unable to derive correct depth estimates. Therefore, we always set $\alpha = 0.1$ during the first epoch of training, to force the network to use mainly the ℓ_1 loss and converge towards good initial depth estimates. For the remaining epochs, we set $\alpha = 1$.

5   The SLED Dataset

In order to train and evaluate the proposed network, we require a dataset containing both events, LiDAR point clouds, as well as a dense ground truth on depths. While we can use the MVSEC dataset [43] for low-resolution cameras, its ground truth is constructed by accumulating point clouds from a LiDAR sensor, a solution which introduces errors in case of moving objects. A similar dataset does not exist for sensors of higher resolution.

For these reasons, we use the CARLA simulator [6] (version 0.9.14) to generate a dataset with perfect synchronization and calibration of the sensors, and perfect ground truth depth. We call it SLED, for Synthetic LiDAR Events Depths dataset. It is composed of 160 sequences of 10 seconds each, for a total of more than 20 minutes of data. These sequences are recorded on the *Town01* to *Town07*

Table 1: Detailed content of our SLED dataset containing 160 sequences of 10 seconds each.

Map	Set	Environment	Features	Night seq.	Day seq.
Town01	Test	Town	Small buildings, bridges, distant forests and mountains, palm trees	4	16
Town02	Train	Town	Small buildings, plazas, forest road	4	16
Town03	Test	City	Tall and small buildings, roundabouts, tunnel, aerial railway	4	16
Town04	Val.	Town	Small buildings, highway, parking, lake, forests and mountains	4	16
Town05	Train	City	Tall buildings, parking, aerial beltway and railway	4	16
Town06	Train	Suburban	Small buildings, U-turns, distant hills	4	16
Town07	Train	Countryside	Barns, grain silos, fields, mountain road	4	16
Town10HD	Train	City	Buildings, monuments and sculptures, playgrounds, seaside	4	16

and *Town10HD* maps (20 sequences for each map), each sequence starting from a different geographic location. By doing so, a wide range of environments is represented within the dataset, as detailed in Table 1.

Each sequence contains a 1280×720 event camera, a 40-channel LiDAR, and a 1280×720 depth camera which is perfectly aligned with the event camera. Both the event data and the depth images are recorded at 200Hz, while the LiDAR is configured to run at 10Hz. RGB images (1280×720) are also provided at a 30Hz rate, aligned with the event-based sensor. The LiDAR sensor is configured with a maximum range of 200 meters. For realism and diversity purposes, AI-controlled vehicles and pedestrians are added to the simulation. Sun altitude also varies, resulting for each map in 4 night recordings, and the other 16 recordings ranging from early morning (where the sun can be directly in front of the camera) to midday (where the sun is at its apogee). Varying cloudiness conditions are also used, adding more or less texture to the sky, and making shadows more diverse.

We also configure the event camera in CARLA to use a linear intensity scale rather than the default logarithmic one, making the events produced by the simulator more realistic. More details on that topic, as well as an overview of the data contained in the dataset, are given in the Supplementary Material.

6 Evaluation

6.1 Dense Depths

On the SLED Dataset For training on the SLED dataset, we use the Adam optimizer [18] with a learning rate of 10^{-4} and a batch size of 4, and train for a total of 50 epochs. To augment input data, we randomly crop it to 608×608, and apply random horizontal flipping.

Numerical results on the testing set are presented in the "Dense depths errors" column of Table 2. Evaluations are conducted on *Town01* and *Town03* maps, which contain challenging environments with many unique features (bridges, tunnels, . . .) that are not present in the training maps. For the max range of 200 meters, ALED estimates depth maps with an average absolute error slightly over 4.5 meters for *Town01*, and around 5 meters for *Town03*. The respective average relative error is around 18% for *Town01*, and around 22% for *Town03*. A first element that can explain these errors is that the LiDAR has a small

Table 2: Errors on the testing set of the SLED dataset for various cutoff depth distances. From left to right: average absolute and relative depth errors on both the "before" D_{bf} and "after" D_{af} depth maps; average absolute depth errors when associating a depth to each event; average absolute depth difference errors and percentage of correctly classified events based on this depth difference.

| Map | Cutoff | Dense depths errors | | | | Sparse depths errors | | | | Depth change map errors | |
| | | On D_{bf} | | On D_{af} | | On D_{bf} | | On D_{af} | | Absolute error | Correctly classified events |
		Raw	Rel.	Raw	Rel.	NN	ALED$_{\mathrm{S}}$	NN	ALED$_{\mathrm{S}}$		(with a threshold of ±1m)
	10m	1.24m	20.99%	1.37m	23.60%	1.32m	1.46m	2.24m	1.79m	2.11m	90.27%
	20m	2.08m	23.06%	2.27m	25.48%	1.51m	1.84m	2.53m	2.15m	3.18m	85.07%
Town01	30m	2.72m	23.76%	2.92m	26.03%	1.71m	2.37m	2.83m	2.67m	3.88m	81.68%
	100m	4.25m	24.01%	4.51m	26.07%	2.40m	3.48m	3.91m	3.95m	5.12m	77.48%
	200m	4.53m	17.20%	4.81m	18.66%	7.86m	5.44m	9.76m	6.23m	7.36m	75.54%
	10m	2.00m	28.91%	2.09m	30.11%	0.47m	0.56m	0.67m	0.66m	1.14m	93.70%
	20m	2.85m	29.91%	2.97m	31.15%	0.64m	0.75m	1.12m	0.87m	2.54m	87.16%
Town03	30m	3.33m	29.10%	3.45m	30.24%	0.92m	1.11m	1.61m	1.26m	3.23m	83.71%
	100m	4.60m	27.37%	4.77m	28.42%	1.88m	2.55m	3.17m	2.88m	4.47m	78.50%
	200m	4.86m	21.50%	5.03m	22.33%	4.43m	3.60m	5.93m	4.10m	6.20m	77.23%
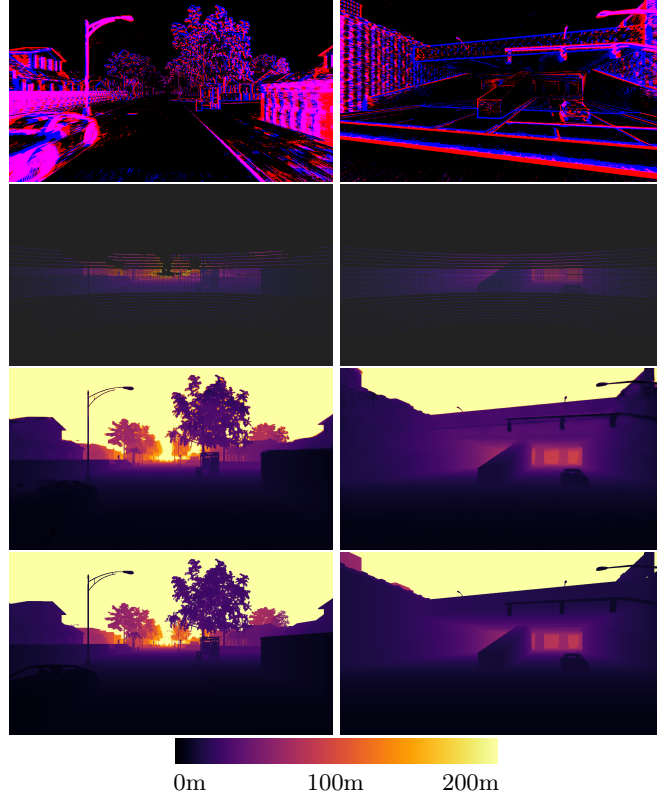
Fig. 4: Two qualitative results on our synthetic SLED dataset, on *Town01* (left) and *Town02* (right). From top to bottom: events, LiDAR, predicted depth map, ground truth, color scale.

vertical coverage of the image: close ground objects or the top of close buildings are not reached by the LiDAR, meaning that accurate depth estimations for these objects are complex to achieve. In opposition, sky pixels (for which ALED produces good results) are only accounted for at the full 200m cutoff. These observations can be correlated to the larger relative errors observed for close cutoff distances than for the full 200m cutoff. It can also be observed that errors on the "after" depth maps are slightly higher. This is to be expected, as the network has to make use of the events to estimate the movement and propagate the depths accordingly. If the reader is interested, results for each sequence of the testing set are given in the Supplementary Material.

Qualitative results are given in Fig. 4. They showcase the ability of the network to estimate accurate depths for the whole image, by using events as a guide for the areas the LiDAR sensor can not reach. This is particularly visible for the trees and the light pole for the left column, or the ceiling of the tunnel for the right column. If the reader is interested, more visual results are given in the Supplementary Material and in the videos linked in the introduction.

On the MVSEC Dataset In order to be able to compare our results with the other approaches in the literature, we also train and evaluate ALED on the MVSEC dataset [43]. We conduct our evaluation under three different sets of weights from different training setups described below:

- ALED$_S$: the network is only trained in simulation, on proposed SLED dataset;
- ALED$_R$: the network is only trained on real data, on the MVSEC dataset;
- ALED$_{S \to R}$: the network is first trained on the SLED dataset, then fined-tuned on the MVSEC dataset.

We use a batch size of 4 and a learning rate of 10^{-4} (10^{-5} when fine-tuning). 50 epochs are used when training the network from zero, 5 when fine-tuning it. We also augment input data, by randomly cropping it to 256×256, and by applying random horizontal flipping. Numerical results are given in Table 3.

Comparing our three sets of trained weights, it appears clearly that training on synthetic data before fine-tuning the network on the MVSEC dataset ($S \to R$) produces the best results. Training from zero on the MVSEC dataset (R) is not as good as the $S \to R$ variant due to the limited data available for training. Finally, training solely on the SLED dataset (S) produces the worst results, due to the large differences in terms of resolution and LiDAR models between the two datasets, and as simulation is not a perfect reproduction of real data.

Proposed ALED$_{S \to R}$ network greatly outperforms all the other approaches of the state of the art. Most impressive results are obtained with distant cutoff depths, where fewer LiDAR points are available: our network is still able to infer accurate depths, while reference methods show large errors. Compared to the frames+events EvT$^+$ method of Sabater et al. [30], we improve the error by $\{-2.7\%, 13.4\%, 31.3\%\}$ at minimum and by $\{60.6\%, 58.8\%, 57.0\%\}$ at maximum for each of the $\{10m, 20m, 30m\}$ cutoff distances respectively. Compared to the LiDAR+events method of Cui et al. [5], this improvement is of $\{32.7\%, 17.4\%, 56.7\%\}$ at minimum and of $\{59.7\%, 39.9\%, 79.1\%\}$ at maximum.

Table 3: Average absolute depth errors (in meters) on the MVSEC dataset for various cutoff depth distances. This evaluation is performed on the "before" depth map D_{bf}, to be consistent with the methods we compare ourselves to.

| Recording | Cutoff | Event-based | | Event- and frame-based | | LiDAR- and event-based | | | |
		Zhu et al. [44]	E2Depth [12]	RAMNet [9]	EvT$^+$ [30]	Cui et al. [5]	ALED$_\mathrm{S}$	ALED$_\mathrm{R}$	ALED$_{\mathrm{S}\to\mathrm{R}}$
Outdoor day 1	10m	2.72	1.85	1.39	1.27	1.24	1.54	0.91	**0.50**
	20m	3.84	2.64	2.17	1.94	1.28	2.55	1.22	**0.80**
	30m	4.40	3.13	2.76	2.37	4.87	3.18	1.43	**1.02**
	50m	-	-	-	-	-	3.79	1.67	**1.31**
	100m	-	-	-	-	-	4.08	1.96	**1.60**
Outdoor night 1	10m	3.13	3.38	2.50	**1.48**	2.26	2.24	1.75	1.52
	20m	4.02	3.82	3.19	2.09	2.19	3.32	2.10	**1.81**
	30m	4.89	4.46	3.82	2.84	4.50	3.82	2.25	**1.95**
	50m	-	-	-	-	-	4.31	2.44	**2.20**
	100m	-	-	-	-	-	4.62	2.73	**2.54**
Outdoor night 2	10m	2.19	1.67	1.21	1.48	1.88	1.94	1.19	**1.09**
	20m	3.15	2.63	2.31	2.13	2.14	2.82	1.65	**1.49**
	30m	3.92	3.58	3.28	2.92	4.67	3.22	1.81	**1.64**
	50m	-	-	-	-	-	3.58	1.95	**1.80**
	100m	-	-	-	-	-	3.78	2.11	**1.97**
Outdoor night 3	10m	2.86	1.42	1.01	1.40	1.78	1.76	0.85	**0.81**
	20m	4.46	2.33	2.34	2.05	1.93	2.43	1.25	**1.16**
	30m	5.05	3.18	3.43	2.79	4.55	2.78	1.42	**1.33**
	50m	-	-	-	-	-	3.12	1.57	**1.51**
	100m	-	-	-	-	-	3.31	1.73	**1.66**

(a) Reference image   (b) Events input   (c) LiDAR input

(d) ALED$_\mathrm{S}$ (e) ALED$_\mathrm{R}$ (f) ALED$_{\mathrm{S}\to\mathrm{R}}$

(g) RAMNet [9]   (h) Ground truth   (i) Scale
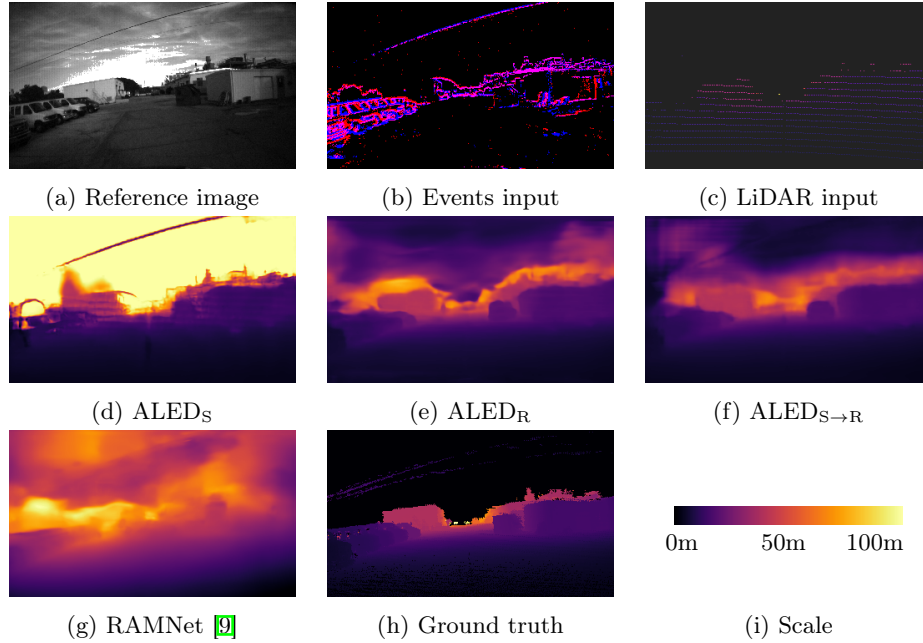
0m   50m   100m

Fig. 5: Qualitative results on "Outdoor day 1" from the MVSEC dataset with real data.

Qualitative results are presented in Fig. 5. All three ALED variants produce results which are visually close to the ground truth for ground objects. Since the MVSEC dataset lacks ground truth depth for the sky, and since the rare elements to have a ground truth for this part of the image are close buildings, close trees, or power lines, the network cannot learn to derive correct depth estimations for the corresponding pixels, leading to the purple blobs in the upper parts of Fig. 5e and 5f. Only the S variant is able to predict accurate values for sky areas (as our SLED dataset contains valid ground truth depths for all pixels), but has more difficulties for ground objects due to the lack of fine-tuning. Between the R and $S \rightarrow R$ variants, improvement can still be seen, for instance for the edges of the objects of Fig. 5e and 5f, which are less uneven in the $S \rightarrow R$ variant. Finally, when comparing our results to RAMNet, we can clearly observe that our method provides in all cases more accurate depth maps, where object boundaries are more prominent, and where estimated depths are closer to the ground truth. This observation further demonstrates that the use of a LiDAR input — even if very sparse — is of great help for obtaining accurate dense depth maps.

6.2   Associating a Depth to Each Event

As stated in Sections 1 and 3, our goal is not only to estimate dense depth maps, but also to associate two depths to each event, allowing for their 3D reprojection and depth difference analysis. Evaluation of the depth association to each event on proposed SLED dataset is given in the "Sparse depths errors" column of Table 2.

The sparse event-LiDAR fusion literature is limited to the method of Li et al. [21]. However, their approach only considers one depth per event, and is intended for a Road Side Unit (RSU) application (i.e., their event camera is fixed and evaluation is conducted on a specific dataset). Therefore, we decided to compare ourselves to a more naive (and faster) baseline: the Nearest Neighbor (NN) approach, where each event is given the depth of its closest LiDAR point. As the Nearest Neighbor approach can not infer correct depths for events which are too far from a LiDAR scan, and so as to provide a fair comparison, we only consider the events between the bottom and top LiDAR scans.

As displayed in Table 2, in the "before" D_{bf} case, depending on the map and the cutoff distance, best results are shared between the NN approach and our ALED$_{\text{S}}$ network. These results can be explained by the fact that our network is more likely to commit large errors for events at the boundary of close objects, as it might estimate that they should be given the depth of the more distant background. On the contrary, the NN approach will always attribute the depth of the closest LiDAR point, and will therefore commit more frequent but smaller errors. In the "after" D_{af} case, despite this potential source of error, our network ALED$_{\text{S}}$ nearly always obtains the best results, as it has correctly learned the temporal propagation of the depths, a task which cannot be completed natively with the NN approach. We also remind here that these results are given for the parts of the image where LiDAR data is available: the NN method would not

be able to derive correct estimations for the other parts of the image. Numerical results on each sequence of the dataset are given in Supplementary Material.

6.3 Depth Difference

We finally estimate the quality of our "two depths per event" approach through the depth change map, as presented in Section 3. We perform two evaluations on our SLED dataset: (1) the average error on the depth change map $D_{af} - D_{bf}$ compared to the true depth changes, and (2) the percentage of correctly classified events when using a difference threshold of 1m on the depth change map. Results of these evaluations are given in the "Depth change map errors" column of Table 2. Numerical results on each sequence of the dataset, as well as some visual results, are given in Supplementary Material.

   We can observe here that, despite a significant absolute error on the depth change maps, events can still be classified correctly, with a rate of success over 75% on *Town01*, and over 77% on *Town03*. We remind here that the ALED network is not trained on these depth change map and classification tasks. As such, we believe that, while even more accurate individual depth maps could improve both the depth change map and classification errors, further improvements could be brought by designing a network specifically dedicated to these tasks.

7   Conclusion

In this article, a novel learning-based approach for estimating dense depth maps from asynchronous LiDAR and event-based data has been proposed. A novel "two depths per event" notion has also been proposed, to solve the issue of events representing a change of depth. A synthetic multimodal dataset has also been recorded, to train and evaluate our method. Multiple evaluations on our synthetic and a real driving datasets have been performed to show the relevance of our contributions. In particular, on the MVSEC dataset, an improvement of up to 79.1% compared to the current LiDAR-and-events state of the art has been achieved, on complex daytime and nighttime recordings.

   In hindsight, further improvements could be brought to the method. Attention-based networks provide state-of-the-art results in numerous vision-based applications, and could improve the fusion of the event and LiDAR modalities. Making the network predict directly sparse depths for each event could also potentially provide better results for the depth to event inference. This could be achieved by using sparse convolutional networks [24] for instance, and could be subject to future work. The use of the Event Volume with a fixed time window as the input representation for the events in our network could also be revised, as it can become ill-suited under large motions. A solution could be to use an alternative representation, such as TORE Volumes [1], or to use adaptive accumulation times using methods such as the one proposed by Liu and Delbrück [22]. Finally, the recording of a real dataset with a high resolution event camera could also be considered, to complete the possibilities offered by the low-resolution MVSEC dataset.

References

1. Baldwin, R.W., Liu, R., Almatrafi, M.B., Asari, V.K., Hirakawa, K.: Time-ordered recent event (TORE) volumes for event cameras. IEEE Transactions on Pattern Analysis and Machine Intelligence (2021)
2. Brandli, C., Berner, R., Yang, M., Liu, S.C., Delbruck, T.: A 240 × 180 130 dB 3 μs latency global shutter spatiotemporal vision sensor. IEEE Journal of Solid-State Circuits **49**, 2333–2341 (2014)
3. Cao, H., Chen, G., Xia, J., Zhuang, G., Knoll, A.: Fusion-based feature attention gate component for vehicle detection based on event camera. IEEE Sensors Journal **21**, 24540–24548 (2021)
4. Chodosh, N., Wang, C., Lucey, S.: Deep convolutional compressed sensing for Li-DAR depth completion. In: Asian Conference on Computer Vision. pp. 499–513 (2018)
5. Cui, M., Zhu, Y., Liu, Y., Liu, Y.M., Chen, G., Huang, K.: Dense depth-map estimation based on fusion of event camera and sparse LiDAR. IEEE Transactions on Instrumentation and Measurement **71**, 1–11 (2022)
6. Dosovitskiy, A., Ros, G., Codevilla, F., López, A., Koltun, V.: CARLA: An open urban driving simulator. In: Proceedings of the 1st Annual Conference on Robot Learning. pp. 1–16 (2017)
7. Gansbeke, W.V., Neven, D., Brabandere, B.D., Gool, L.V.: Sparse and noisy Li-DAR completion with RGB guidance and uncertainty. 2019 16th International Conference on Machine Vision Applications (MVA) pp. 1–6 (2019)
8. Gehrig, D., Rebecq, H., Gallego, G., Scaramuzza, D.: EKLT: Asynchronous photo-metric feature tracking using events and frames. International Journal of Computer Vision **128**, 601–618 (2019)
9. Gehrig, D., Rüegg, M., Gehrig, M., Hidalgo-Carrió, J., Scaramuzza, D.: Combining events and frames using recurrent asynchronous multimodal networks for monocular depth prediction. IEEE Robotics and Automation Letters **6**, 2822–2829 (2021)
10. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. 2015 IEEE International Conference on Computer Vision (ICCV) pp. 1026–1034 (2015)
11. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) pp. 770–778 (2016)
12. Hidalgo-Carrió, J., Gehrig, D., Scaramuzza, D.: Learning monocular dense depth from events. 2020 International Conference on 3D Vision (3DV) pp. 534–542 (2020)
13. Hu, Y., Binas, J., Neil, D., Liu, S.C., Delbrück, T.: DDD20 end-to-end event camera driving dataset: Fusing frames and events with deep learning for improved steering prediction. 2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC) pp. 1–6 (2020)
14. Huang, Z., Fan, J., Cheng, S., Yi, S., Wang, X., Li, H.: HMS-Net: Hierarchical multi-scale sparsity-invariant network for sparse depth completion. IEEE Transactions on Image Processing **29**, 3429–3441 (2020)
15. Jaritz, M., de Charette, R., Wirbel, É., Perrotton, X., Nashashibi, F.: Sparse and dense data with CNNs: Depth completion and semantic segmentation. 2018 International Conference on 3D Vision (3DV) pp. 52–60 (2018)
16. Jiang, Z., Xia, P., Huang, K., Stechele, W., Chen, G., Bing, Z., Knoll, A.: Mixed frame-/event-driven fast pedestrian detection. 2019 International Conference on Robotics and Automation (ICRA) pp. 8332–8338 (2019)

17. Kim, H., Leutenegger, S., Davison, A.J.: Real-time 3D reconstruction and 6-DoF tracking with an event camera. In: European Conference on Computer Vision (2016)
18. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. CoRR **abs/1412.6980** (2015)
19. Ku, J., Harakeh, A., Waslander, S.L.: In defense of classical image processing: Fast depth completion on the CPU. 2018 15th Conference on Computer and Robot Vision (CRV) pp. 16–22 (2018)
20. Kueng, B., Mueggler, E., Gallego, G., Scaramuzza, D.: Low-latency visual odometry using event-based feature tracks. 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) pp. 16–23 (2016)
21. Li, B., Meng, H., Zhu, Y., Song, R., Cui, M., Chen, G., Huang, K.: Enhancing 3-D LiDAR point clouds with event-based camera. IEEE Transactions on Instrumentation and Measurement **70**, 1–12 (2021)
22. Liu, M., Delbrück, T.: Adaptive time-slice block-matching optical flow algorithm for dynamic vision sensors. In: British Machine Vision Conference (2018)
23. Maddern, W.P., Newman, P.: Real-time probabilistic fusion of sparse 3D LIDAR and dense stereo. 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) pp. 2181–2188 (2016)
24. Messikommer, N., Gehrig, D., Loquercio, A., Scaramuzza, D.: Event-based asynchronous sparse convolutional networks. In: European Conference on Computer Vision (2020)
25. Nam, Y., Isfahani, S.M.M., Yoon, K.J., Choi, J.: Stereo depth from events cameras: Concentrate and focus on the future. 2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 6104–6113 (2022)
26. Paikin, G., Ater, Y., Shaul, R., Soloveichik, E.: EFI-Net: Video frame interpolation from fusion of events and frames. 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW) pp. 1291–1301 (2021)
27. Pan, L., Scheerlinck, C., Yu, X., Hartley, R.I., Liu, M., Dai, Y.: Bringing a blurry frame alive at high frame-rate with an event camera. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 6813–6822 (2019)
28. Pan, X., Luo, P., Shi, J., Tang, X.: Two at once: Enhancing learning and generalization capacities via IBN-Net. In: European Conference on Computer Vision (2018)
29. Perot, E., de Tournemire, P., Nitti, D.O., Masci, J., Sironi, A.: Learning to detect objects with a 1 megapixel event camera. 34th Conference on Neural Information Processing Systems (NeurIPS 2020) (2020)
30. Sabater, A., Montesano, L., Murillo, A.C.: Event transformer+. A multi-purpose solution for efficient event data processing. ArXiv **abs/2211.12222** (2022)
31. Scheerlinck, C., Barnes, N., Mahony, R.E.: Continuous-time intensity estimation using event cameras. In: Asian Conference on Computer Vision. pp. 308–324 (2018)
32. Schraml, S., Belbachir, A.N., Bischof, H.: An event-driven stereo system for real-time 3-D 360° panoramic vision. IEEE Transactions on Industrial Electronics **63**, 418–428 (2016)
33. Schraml, S., Belbachir, A.N., Milosevic, N., Schön, P.: Dynamic stereo vision system for real-time tracking. Proceedings of 2010 IEEE International Symposium on Circuits and Systems pp. 1409–1412 (2010)
34. Siam, M., Valipour, S., Jägersand, M., Ray, N.: Convolutional gated recurrent networks for video segmentation. 2017 IEEE International Conference on Image Processing (ICIP) pp. 3090–3094 (2017)

35. Song, R., Jiang, Z., Li, Y., Shan, Y., Huang, K.: Calibration of event-based camera and 3D LiDAR. 2018 WRC Symposium on Advanced Robotics and Automation (WRC SARA) pp. 289–295 (2018)
36. Ta, K., Bruggemann, D., Brodermann, T., Sakaridis, C., Gool, L.V.: L2E: Lasers to events for 6-DoF extrinsic calibration of lidars and event cameras. ArXiv **abs/2207.01009** (2022)
37. Teed, Z., Deng, J.: RAFT: Recurrent all-pairs field transforms for optical flow. In: European Conference on Computer Vision. pp. 402–419 (2020)
38. Tomy, A., Paigwar, A., Mann, K.S., Renzaglia, A., Laugier, C.: Fusing event-based and RGB camera for robust object detection in adverse conditions. 2022 International Conference on Robotics and Automation (ICRA) pp. 933–939 (2022)
39. Uhrig, J., Schneider, N., Schneider, L., Franke, U., Brox, T., Geiger, A.: Sparsity invariant CNNs. 2017 International Conference on 3D Vision (3DV) pp. 11–20 (2017)
40. Ummenhofer, B., Zhou, H., Uhrig, J., Mayer, N., Ilg, E., Dosovitskiy, A., Brox, T.: DeMoN: Depth and motion network for learning monocular stereo. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR) pp. 5622–5631 (2017)
41. Weikersdorfer, D., Adrian, D.B., Cremers, D., Conradt, J.: Event-based 3D SLAM with a depth-augmented dynamic vision sensor. 2014 IEEE International Conference on Robotics and Automation (ICRA) pp. 359–364 (2014)
42. Xu, Y., Zhu, X., Shi, J., Zhang, G., Bao, H., Li, H.: Depth completion from sparse LiDAR data with depth-normal constraints. 2019 IEEE/CVF International Conference on Computer Vision (ICCV) pp. 2811–2820 (2019)
43. Zhu, A.Z., Thakur, D., Özaslan, T., Pfrommer, B., Kumar, V.R., Daniilidis, K.: The multivehicle stereo event camera dataset: An event camera dataset for 3D perception. IEEE Robotics and Automation Letters **3**, 2032–2039 (2018)
44. Zhu, A.Z., Yuan, L., Chaney, K., Daniilidis, K.: Unsupervised event-based learning of optical flow, depth, and egomotion. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) pp. 989–997 (2019)