International Journal of Computer Vision (2021) 129:900–920
https://doi.org/10.1007/s11263-020-01410-2

Learning to Reconstruct HDR Images from Events, with Applications to Depth and Flow Prediction

Mohammad Mostafavi[1] · **Lin Wang**[2] · **Kuk-Jin Yoon**[2]

Received: 12 September 2019 / Accepted: 26 November 2020 / Published online: 5 January 2021

Abstract
Event cameras have numerous advantages over traditional cameras, such as low latency, high temporal resolution, and high dynamic range (HDR). We initially investigate the potential of creating intensity images/videos from an adjustable portion of the event data stream via event-based conditional generative adversarial networks (cGANs). Using the proposed framework, we further show the versatility of our method in directly handling similar supervised tasks, such as optical flow and depth prediction. Stacks of space-time coordinates of events are used as the inputs while the proposed framework is trained to predict either the intensity images, optical flows, or depth outputs according to the target task. We further demonstrate the unique capability of our approach in generating HDR images even under extreme illumination conditions, creating non-blurred images under rapid motion, and generating very high frame rate videos up to the temporal resolution of event cameras. The proposed framework is evaluated using a publicly available real-world dataset and a synthetic dataset we prepared by utilizing an event camera simulator.

Keywords Event camera · Conditional generative adversarial network · Image and video reconstruction · High dynamic range · High frame rate · Optical flow · Depth

1 Introduction

Event cameras are bio-inspired vision sensors that mimic the human eye in receiving visual information (Lichtsteiner et al. 2008). Whereas traditional cameras transmit intensity frames at a fixed rate, event cameras transmit the changes in intensity at the time of the changes, in the form of asynchronous events that deliver the space-time coordinates of the intensity changes. It has been stated that event cameras, in principle, can transfer all the information needed to reconstruct an image or a full video stream (Rebecq et al. 2017; Bardow et al. 2016; Reinbacher et al. 2016). However, this claim has never been thoroughly substantiated. Motivated by the recent advances in deep learning in image reconstruction and translation, we initially address the problem of generating intensity images or videos from an adjustable portion of the event data stream via camera-based conditional generative adversarial networks (cGANs) (Isola et al. 2017). Using the proposed framework, we further investigate its potential for generating optical flow and depth images from events.

Our method follows end-to-end supervised learning and accomplishes the target task by flexibly varying the training data. Furthermore, our method can generate high-frame-rate and high-dynamic-range (HDR) videos without any motion blur, solely from event data. To the best of our knowledge, our work is one of the first attempts to generate high-quality, non-blurry, high-frame-rate, HDR images and videos from pure events, even under fast camera or scene movements or extreme illumination conditions. Although optical flow and depth can also be synthesized in this manner, we focus more on generating intensity images/videos. This is because (1) they naturally include more gradient information and

Communicated by Takayuki Okatani.

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/s11263-020-01410-2.

✉ Kuk-Jin Yoon
kjyoon@kaist.ac.kr

Mohammad Mostafavi
mostafavi@gist.ac.kr

Lin Wang
wanglin@kaist.ac.kr

[1] Computer Vision Lab., GIST, Gwangju, Korea

[2] Visual Intelligence Lab., KAIST, Daejeon, Korea

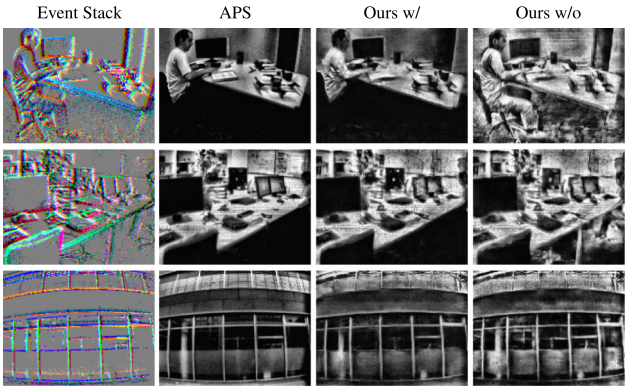

Fig. 1 Recovery of intensity images solely from event data. We can either reconstruct intensity images/videos with the same details as those of an active pixel sensor (APS) or reconstruct further details that the APS cannot reach. This is obtained by training either with (w/) the events in absolute black or white regions or without (w/o) such events. Excessive constructed areas can be found in the top row near the person's leg, in the areas under the table in the second row, or in the details inside the windows of the last row. The event stacks are visualized as pseudo-color images for reference throughout this paper

also vary spatially within an image or temporally among the sequence of images and (2) motion blur and sharpness are not highly critical features in optical flow or depth estimation. For these reasons, we first fully focus on data preparation for the intensity image/video generation. Then, we describe the event stacking details and network architecture. Finally, we show the usefulness of our method in direct depth and optical flow generation (Fig. 1).

We first propose an event-based image translation framework that generates quality images from events better than those generated by active pixel sensor (APS) frames and other previous methods. To feed events to the proposed framework, we also propose two novel event stacking methods: stacking based on time (SBT) and stacking based on the number of events (SBN). In particular, we designed these methods by considering how event streams are processed. In this way, we can obtain high-frame-rate and HDR representations with no motion blur, which, in contrast, is impossible for normal cameras.

It turns out that it is possible to generate a video with up to 1 million frames per second (FPS) in terms of the reconstruction frequency using the proposed stacking methods. It should be noted that the differences between two consecutive frames may not be quite visible in a high-frame-rate video reconstruction setting. This is because the two consecutive event stacks used for reconstructing these images will only have one single event different between them. Therefore, in practical applications, at least a few hundred events should be different. Furthermore, this is inherently different from a high-speed camera with 1 million FPS, which can capture all the motion happening during 1 μs.

This paper is an extended version of our previous work (Wang et al. 2019) and contains additional parts including the investigation of different data terms, numerical comparisons with more recent methods, new experiments on temporal consistency, and the possibility of extending the work using color event cameras. Furthermore, we show the versatility of the proposed framework in directly estimating depth and optical flow without the need to first reconstruct the intensity images. Furthermore, to verify the robustness of the framework, we conducted intensive experiments for its evaluation and comparison with other methods using a publicly available real-world dataset captured under real-world indoor, outdoor, and driving conditions (Bardow et al. 2016; Mueggler et al. 2017; Zhu et al. 2018), together with a simulated event dataset we prepared using an event camera simulator (Rebecq et al. 2018).

2 Related Works

Early attempts to reconstruct intensity images involved statistical methods such as interconnected maps (Cook et al. 2011), probabilistic filters (Kim et al. 2008), or patch-based sparse dictionaries (Barua et al. 2016). Higher quality results were obtained by utilizing regularization terms (Bardow et al. 2016) in which the intensity image and the motion field are jointly estimated for generic motion using a convex optimization scheme.

Reinbacher et al. (2016) introduced a variational denoising framework that iteratively filters incoming events. They guided the events through a manifold regarding their timestamps to reconstruct the image. Compared with the method of Bardow et al. (2016), their method yields more grayscale variations in untextured areas and recovers more details, and their graphics processing unit (GPU)-based algorithm can be executed in real time. Measurements and simulations on event cameras with RGBW color filters were proposed by Moeys et al. (2017). They first presented a naive method that requires the initial APS image from the event camera to be updated with the incoming events and then an iterative scheme creates a regularized image by solving the Poisson equation about the divergence of the intensity image, which produces better outputs than those of the naive method.

Following the trend in deep learning-based solutions, event-based vision has also experienced many success stories. A hybrid intensity and event fusion method proposed in Shedligeri et al. (2018) partially utilizes deep learning to obtain better intensity predictions. However, it is not at par with the most recent approaches proposed in Wang et al. (2019); Rebecq et al. (2019), where relatively realistic outputs were synthesized in challenging fast movements or HDR scenes using pure events. An experimental comparison between these two methods is provided in Sect. 4.4.

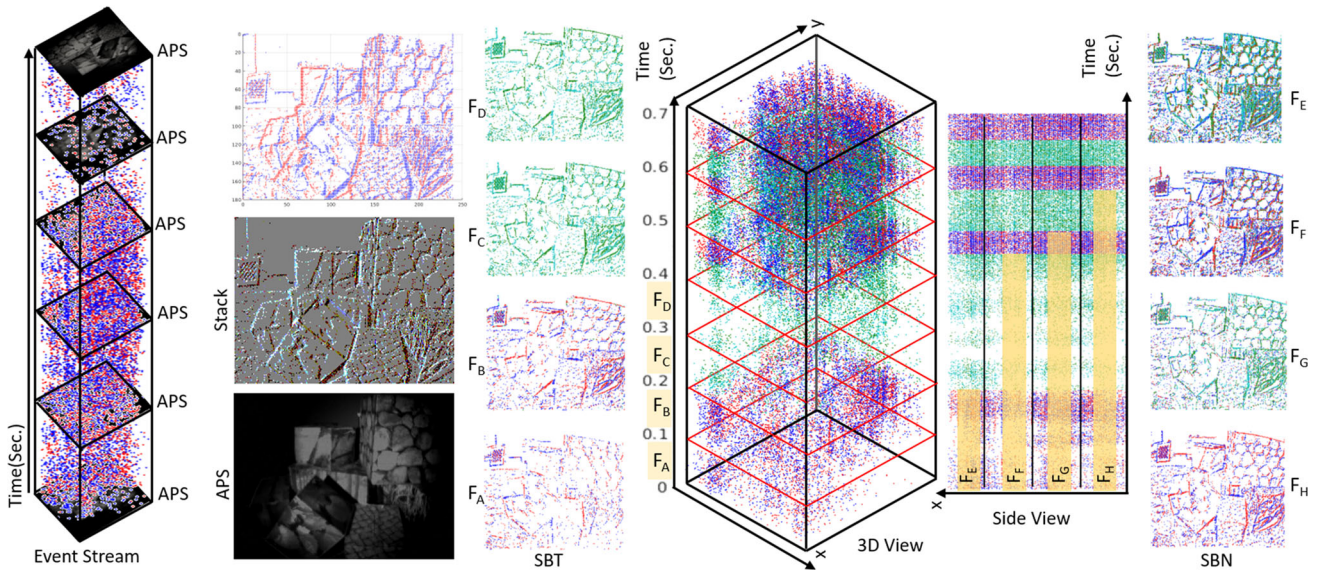

Fig. 2 The event stream and construction of the stacks by SBT and SBN. Two main color tuples of (red(+) and blue(-)) and (green(+) and cyan(-)) express the event polarity (plus or minus) throughout this paper. In the 3D view, two types of stacking (SBT on the left and SBN on the right) are shown using the yellow highlighted time. The 3D view followed by its side view is color coded with (red and blue) and (green and cyan) periodically (every 5000 events) for better visualization. All the images and plotted data are from the "hdr_boxes" sequence of Mueggler et al. (2017) (Color figure online)

Zhu et al. (2018) introduced an unsupervised method for optical flow estimation using the APS frames as a supervisory signal in the training phase. They obtained better results by changing their event representation while using extra loss functions on the motion blur and stereo similarity (Zhu et al. 2019). In another self-supervised method, evenly cascaded networks are utilized for dense depth and optical flow predictions (Ye et al. 2018). Kim et al. (2016) proposed a real-time three-dimensional (3D) reconstruction method that can recover the six-degrees-of-freedom (6DoF) motion together with depth and intensity reconstruction through probabilistic filters. However, in their method, depth estimation relies on the intensity image reconstruction. A multi-view stereo scheme for estimating 3D structures from an event camera under a known trajectory was presented in Rebecq et al. (2018). The authors introduced a space-sweep method to directly perform 3D reconstruction by utilizing the sparsity of the event stream. A unified framework to calculate motion, depth, and optical flow was presented in Gallego et al. (2018); it minimizes the objective function by aligning the trajectories of the event points in three dimensions over a short time interval and maximizes the contrast of the image of warped events. The depth is estimated by selecting regions with a large variance over the disparity space image.

Moreover, event data alone or accompanied with APS images have also been used in training convolutional neural networks for other tasks such as steering prediction (Moeys et al. 2016), self-driving cars (Binas et al. 2017; Maqueda et al. 2018), 6DoF pose relocalization [4], or supervised object detection (Chen 2018). A more detailed survey on event-based vision is available in Gallego et al. (2019). Conditional GANs, on the other hand, have been used in many applications such as image prediction from a normal map (Wang and Gupta 2016), future frame prediction (Mathieu et al. 2015), image generation from sparse annotations (Karacan et al. 2016), or style transfer (Isola et al. 2017; Atapour-Abarghouei and Breckon 2018; Ledig et al. 2017; Li and Wand 2016; Zhu et al. 2017). The key strength of cGANs is that there is no need to tailor the loss function in regard to the given specific tasks, and they can generally adapt their own learned loss to the data domain where they are trained. Actually, there is no qualitative study showing the effectiveness of cGANs on event data, and, therefore, we investigate this potential in this paper.

3 Proposed Method

In this section, we describe the proposed framework for reconstructing HDR and non-blurry images and videos from events. To this end, we exploit currently available deep learning based models, such as cGANs, as potential solutions for event-based intensity image reconstruction.

However, since event data are quite different from conventional frame-based images, how to feed event data as inputs to a deep neural network remains a challenge. Hence, in the

following Sect. 3.1, we first propose two event stacking methods that can provide off-the-shelf inputs to neural networks. Then, in Sect. 3.2, we will describe the proposed novel yet simple cGAN framework.

3.1 Event Stacking

In an event camera, each event e is represented as a tuple (u, v, t, p), where u and v are the pixel coordinates, t is the timestamp of the event, and $p = \pm 1$ is the polarity of the event, which is the sign of the brightness change ($p = 0$ for no event). These events are shown as a stream on the left-hand side of Fig. 2. On the basis of the frame rate of the intensity camera, we synchronize the APS images and asynchronous events in between two consecutive APS frames.

To feed the event data input to the network, we require new representations of the event data. One simple way to do this is to form the 3D event volume as $p(u, v, t)$ for some time duration, ensuring enough event data for the image reconstruction. We denote the temporal resolution of an event camera by δt, the time duration by t_d, and the size of the 3D volume by (w, h, n), where w and h represent the spatial resolution of an event camera and $n = t_d/\delta t$. This is equivalent to having an n-channel image input for the network.

This representation preserves all the information about the events. However, the problem is that the number of channels is very large. For example, when t_d is set to 10 ms, then n is about 10K, which is extraordinarily large since the temporal resolution of an event camera is approximately 1 μs. For this reason, we construct the 3D event volume with a small n by forming each channel via merging and stacking the events within a small time interval.

Event stacking is the initial step to make a tensor-like representation suitable for deep learning-based architectures, and it can be done in different ways, but, usually, the temporal information of the event is necessarily sacrificed in return.

3.1.1 Stacking Based on Time (SBT)

In this approach, the streaming events in between the time references of two consecutive intensity images (APS) of the event camera, denoted as Δt, are merged. However, not all events are merged into a single frame. Instead, the time duration of the event stream is divided into n equal-scale portions, and then n grayscale frames, $S_p^i(u, v)$, $i = 1, 2, .., n$, are formed by merging the events in each time interval of $[\frac{(i-1)\Delta t}{n}, \frac{i\Delta t}{n}]$. $S_p^i(u, v)$ is the sum of the polarity (p) values at (u, v). These n grayscale frames are concatenated again to form one stack $S_p(u, v, i) = S_p^i(u, v)$, $i = 1, 2, .., n$, which is fed to the network as the input. As mentioned earlier, this stacking method loses the time information of events within the time interval $\frac{\Delta t}{n}$. However, the stack itself, as a sequence of frames from one to n, still holds the temporal information to some extent. Therefore, a larger n can keep more temporal information.

Figure 2 illustrates how to merge and stack the events. When $n = 3$ (*i.e.*let@tokeneonedot, frames F_A, F_B, and F_C are stacked into one stack), the stack can be visualized as a pseudo-color frame, as shown in the left-hand side of Fig. 2 above the APS image. Based on the time shown at the event manifold in the middle of Fig. 2, starting from zero on the 3D view, the location of the APS image is around the location of the third red rectangle near 0.03 s (the frame rate of the APS image is 33 FPS).

3.1.2 Stacking Based on the Number of Events (SBN)

Unfortunately, SBT has an intrinsic limitation that comes from the event camera, which is the lack of events when there is no movement of the scene or the camera. When the event data within the time interval are not enough for the image reconstruction, it is inevitably difficult to obtain good HDR images. This is the case for the fourth and fifth frames of the event stream at the left-hand side of Fig. 2. Furthermore, another flaw comes from the case of having too many events in one time frame as in the third time frame.

SBN coincides more with the nature of an event camera, which is being asynchronous to time, and can overcome the aforementioned limitations of SBT. In this method, a frame is formed by merging the events according to the number of incoming events, as illustrated in Fig. 2. The first N_e events are merged into frame 1 and the next N_e events into frame 2, and this is continued up to frame n to create one stack of n frames. Then, this n-frame stack containing nN_e events in total is used as an input to the network.

This method guarantees enough rich event data to reconstruct images depending on the N_e value. F_E, F_F, F_G, and F_H in Fig. 2 are the frames corresponding to the different numbers of events, *i.e.*let@tokeneonedot, N_e, $2N_e$, $3N_e$, and $4N_e$, respectively. Since we count the total number of events per stack with time, we can adaptively adjust the number of events in each frame and also in one stack. In the special case, where several positive or negative events land on the same pixel location in a stack, only the latest event is considered. This means that all previous events on that location will not be considered in the stack. Therefore, choosing the correct number of events per stack is important, although the network can, to some degree, handle excessive or missing events, as shown in Fig. 9. Further details and ablation studies on SBN and SBT are available in Wang et al. (2019).

When the events in the time interval $[i - \Delta t, i]$ are used for one input stack for the image $I(i)$ in a video, the next input stack for the image $I(i + t_s)$ in the video can be constructed by using the events in the time interval $[i - \Delta t', i + t_s]$ (for SBT $\Delta t' = \Delta t - t_s$), with the time shift t_s. Then, the frame rate of the output video becomes $\frac{1}{t_s}$. It is also worth noting

that two stacks have a large time overlap $[i - \Delta t', i]$ with the duration $\Delta t'$. If $\Delta t' >> t_s$, the temporal consistency is naturally enforced for the nearby frames.

3.1.3 Further Event Representations

Although many new event representations have been proposed, we show that our network can reconstruct intensity images and perform depth and optical flow estimations from the proposed simple yet effective stacking methods: SBN and SBT. Employing further event stacking algorithms, *e.g*let@tokeneonedot, (Zhu et al. 2018, 2019), might be straightforward; however, slight changes are explicitly required to the first few layers of our network depending on the chosen event representation. We summarize the similarities and differences of our event stacking algorithm to the event representations used in E2VID (Rebecq et al. 2019) or depth and optical flow estimations (Zhu et al. 2018, 2019), as shown in Table 1. We compare the dimensions of each event representation, and describe its characteristics in keeping temporal and polarity information.

The per polarity summation (Zhu et al. 2018) uses a $4 \times H \times W$ tensor representation, where H and W are the input height and width of the sensors, respectively. They sum all the events based on their polarity into two separate channels and keep the latest timestamp and report it in the next two channels based on the event polarity.

The dimensions of our event stack is $C \times H \times W$, where C is the number of channels for each stack. We mainly use $C = 3$ in our experiments as it can be easily visualized, saved, or concatenated as a 3 channel RGB image but are not limited to it. The number of channels C, can be considered similar to the notation of bins (B) in a spatio-temporal voxel grid with $B \times H \times W$ dimensions (Zhu et al. 2019). The number of such bins (B) or channels (C) can be chosen arbitrarily. For instance, E2Vid (Rebecq et al. 2019) uses a spatio-temporal voxel grid with B=10 bins, and our method uses C=3 channels.

Like the voxel grids, we also preserve the temporal information by using multiple channels, which can reveal the temporal relation and flow between the events in the channels as visualized in Fig. 15. Unlike the voxel grid, which discards the event polarity by summing them, we keep the polarities of the latest event per pixel. However, previous polarity information might be replaced by newer events landing in a pixel.

3.2 Network Architectures

In this section, we describe the proposed generator and discriminator structures inspired by the works in Li and Wand (2016); Yi et al. (2017). Details of the architectures including the size of each layer can be found in Figs. 3 and 4.

3.2.1 Generator Architecture

The core of the event-to-image translation is how to map a sparse event input to a dense HDR output with details, while sharing the same structural image features such as edges, corners, and blobs. The encoder-decoder network is the most used network for image-to-image translation tasks. The input is continuously downsampled through the network and then upsampled back to obtain the translated result. Since, in the event-to-image translation problem, there is a huge amount of important high-frequency information from the event data passing through the network, it is likely to lose detailed features of the events during this process and induce noise to the outputs. For that reason, we consider the similar approaches proposed in Isola et al. (2017), wherein we further add skip connections to the U-net (Ronneberger et al. 2015) network structure in Reinbacher et al. (2016).

In Figs. 3 and 4, each gray block presents a convolutional layer where the first two numbers show the filter size and the last number is the number of filters. Each box includes a batch norm followed by a leaky rectified linear unit layer, with a slope of 0.2. All convolutions use 4×4 spatial filters with a stride of 2. The decoder gets upsampled by a factor of 2, and, in return, the encoder and the discriminator get downsampled also by a factor of 2. In the last layer of the decoder, a convolution is applied to map the number of output channels back to 1 followed by a tanh function. The first three layers of the decoder also have a dropout layer with a rate of 50%. The networks were all initialized from a Gaussian distribution with a mean of 0 and a standard deviation of 0.02 and trained from scratch. We utilized the Adam optimizer (Kingma and Ba 2015), with a learning rate of 0.0002 and momentum parameters of $\beta_1 = 0.5$ and $\beta_2 = 0.999$. The networks were trained with a batch size of 10 for 150 epochs.

3.2.2 Discriminator Architecture

Our discriminator can be considered as a method for minimizing the style transfer loss between the events and the intensity images. Mathematically, the objective function is defined as

$$L_{cGAN}(G, D) = E_{e,g}[\log D(e, g)] + E_{e,\epsilon}[\log (1 - D(e, G(e, \epsilon)))]. \quad (1)$$

where e indicates the original event, g indicates the target image (namely, the ground truth (GT)), and ϵ indicates the random noise vector. G tries to produce the output $G(e, \epsilon)$, which cannot be distinguished from the GT image g, whereas the discriminator D is adversarially trained to detect the 'fake' image of the generator. Here, for the regularization,

Table 1 Comparing different event representations used for intensity (Rebecq et al. 2019), depth or optical flow estimation methods (Zhu et al. 2018, 2019), where H,W,C and B are the input height, input width, number of designated channels and number of designated bins, respectively

Representation	Dimensions	Temporal information	Description
Per polarity sum (Zhu et al. 2018)	$4\times H\times W$	Discarded	Sum of each polarity
Voxel grid (Zhu et al. 2019; Rebecq et al. 2019)	$B\times H\times W$	Kept (B bins)	Voxel of sum of events
SBN, SBT	$C\times H\times W$	Kept (C channels)	latest event at N_e/C, $\Delta t/C$

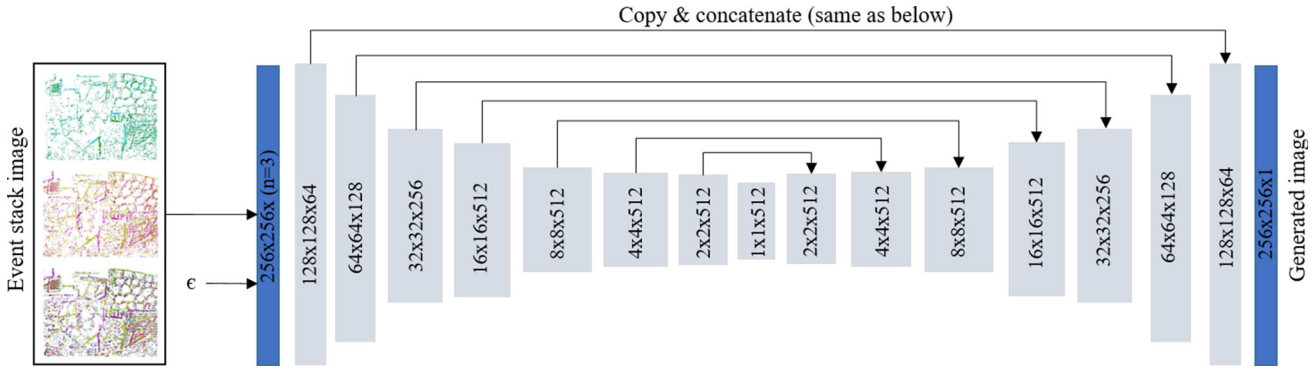

Fig. 3 The generator network: A U-net architecture Ronneberger et al. (2015); Isola et al. (2017) (with skip connections) that takes an input with a dimension of $256\times256\times n$ ($n = 3$ for this example) together with a random noise vector ϵ. The gray boxes correspond to multi-channel feature maps where the number of channels is denoted inside each box. The first two numbers indicate the filter sizes followed by the number of filters

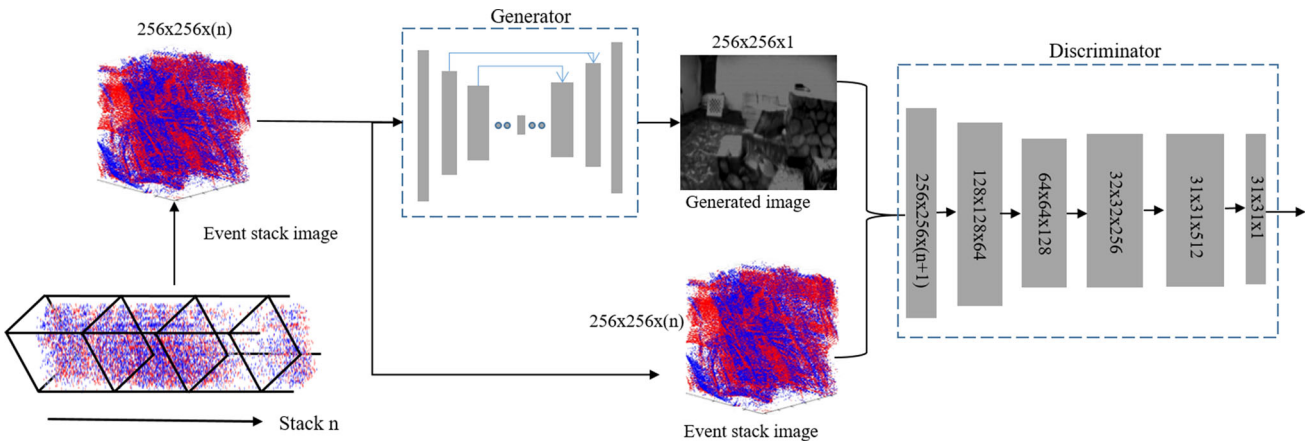

Fig. 4 The proposed framework with the generator and discriminator networks. Our discriminator network is similar to PatchGAN (Yi et al. 2017), which takes two images (the original APS image and the image generated by the generator from the events). The discriminator takes both the input events and the generated image and discriminates whether the generated image is from the ground-truth APS images or from a generated image

the $\mathcal{L}_1$ norm is used to reduce the blurring and is defined as

$$L_{\mathcal{L}_1}(G) = E_{e,g,\epsilon}[\|g - G(e,\epsilon)\|_1]. \tag{2}$$

This $\mathcal{L}_1$ norm is aimed to make the discriminator focus more on the high-frequency structure of the images generated from the events. Eventually, the objective is to estimate the total loss from the event-to-image translation as

$$G^* = \arg\min_G \max_D [L_{cGAN}(G, D) + \lambda L_{\mathcal{L}_1}(G)]. \tag{3}$$

where λ is a parameter for adjusting the learning rate. With the noise ϵ, the network could learn a mapping from event e and $\epsilon \rightarrow g$, which could match the distribution based on events and help to produce more deterministic outputs.

3.3 Dataset Preparation

Following our main goal of having a versatile framework for synthesizing intensity images, optical flow, or depth, we prepared event stacks with the corresponding GT APS frame images, which were different depending on the task to learn.

3.3.1 Intensity Image Dataset

For the intensity image reconstruction, our training datasets were mainly prepared from three different sources. We created the first dataset by using the captured sequences in Mueggler et al. (2017), where many real-world scenes are included. We prepared the second dataset ourselves for various training and test purposes, which we have made publicly available for broader use of the research community. [1] Both of these datasets were captured using a DAVIS 240C camera and contain many indoor and outdoor scenes. The third dataset was generated from synthesized events by utilizing ESIM (Rebecq et al. 2018), an open-source event camera simulator. The real-world intensity data-set contains many indoor and outdoor scenes captured with various rotations and translations of the event camera. Our training data consisted of pairs of stacked events, as explained in Sect. 3.1, together with the APS frames from both the real-world scenes and the GT frames generated from ESIM.

Further evaluations of the intensity image reconstruction were performed on the multi-vehicle stereo event camera dataset (MVSEC) sequences (Zhu et al. 2018) together with the parts of the real-world dataset that were not used in the training. The intensity images from the MVSEC test set can be synched with the GT depth and optical flow reconstructions as well. A sample event stack together with the intensity image from this dataset is presented in the bottom row of Fig 18(b). We used the real-world dataset for training the network. Hence, we carefully pre-processed the training data to prevent the proposed network from learning the unwanted properties of the APS frames. APS frames suffer from motion blur under rapid motion and have a limited dynamic range, resulting in loss of details, as shown in Fig. 1. Therefore, directly using the real APS frames as the GT is not a proper way for training the network since our goal is to produce HDR images without motion blur by fully exploiting the advantages of event cameras. For that reason, the events relevant to the black and white regions of the training data are removed from the input to make the network learn to generate HDR images from events.

More specifically, when making event stacks and APS image pairs, if the intensity of a specific pixel in the APS frame is equal to or lower than our low threshold (which we set to 1 for our experiments), we consider it to be an extreme case of lighting. Therefore, any event in that pixel location will be removed from the paired event stack. In contrast, if the pixel value is greater than our high threshold (which we set to 254 in our experiments), all events landing in that pixel location will not be considered in its dual-event stack. Furthermore, the APS images are classified as blurred and non-blurred through manual inspection. If the APS frame is blurry, its joint event stack with that APS frame will not be used for the training. However, human inspection is not our sole reference for selecting good images. We further use the Blind/Referenceless Image Spatial Quality Evaluator (BRISQUE) (Mittal et al. 2012) to filter out lower quality images with relatively low scores. Internally, BRISQUE utilizes normalized luminance coefficients to quantify the *naturalness* of images. We used the public available implementation of BRISQUE from OpenCV [2] [54]. The simulated sequences are mainly generated from ESIM, where events are produced while a virtual camera moves freely in all directions to capture different scenes in the given images. Since the events and APS images are generated from a controlled simulation environment, the APS frames are counted directly as the GT for the image reconstruction. Therefore, the refinement of the training data as mentioned above is not required for simulated datasets.

3.3.2 Depth and Optical Flow Datasets

We mainly followed the procedure in the previous section and used two series of data for estimating the depth and optical flow. One was based on the simulator (ESIM), and the other was from the real-world sequences captured by a stereo pair of DAVIS 346B cameras mounted on multiple vehicles (MVSEC) (Zhu et al. 2018). For the simulated datasets, we created 15 different nonplanar scenes using computer graphics software. We placed different 3D objects such as cubes, spheres, and animals on a surface with different textures on the objects and the floor of the scene. Some samples are shown in Figs. 5 and 16 of Sect. 5. Using the SBN method, we created stacks of three channels using 60K events per stack. Through this process, we created 8500 pairs of event stacks and optical flow or depth pairs. We set the simulator parameters such that the rate, optical flow, and depth creation of the image were all the same and its FPS was 50. The optical flow was set on the basis of the difference between two consecutive frames.

The MVSEC dataset contains recorded outputs of multiple sensors, including stereo event cameras, motion capture systems, and LiDAR depth sensors mounted on multiple moving vehicles to capture different day/night and indoor/outdoor scenes. The dataset contains GT depth maps from the LiDAR

[1] Our dataset is publicly available at https://github.com/wl082013/ESIM_dataset.

[2] imported from OpenCV: cv::quality::QualityBRISQUE.

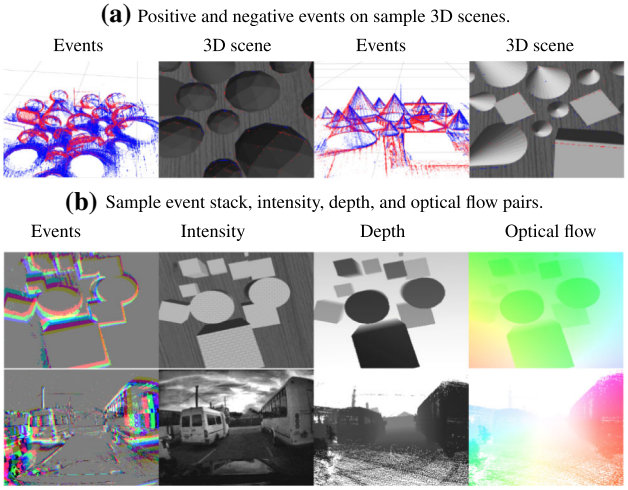

Fig. 5 **a** Sample nonplanar scenes used for the simulator depth and optical flow training using ESIM. The positive (red) and negative (blue) events being triggered on the 3D scene are shown together with a sample intensity image from that scene. **b** Examples of the event stack, intensity image, depth, and optical flow from ESIM and of the real-world scenes from MVSEC. The 3D scenes were visualized using Blender and the RViz 3D visualization tool (Community 2018), [51] (Color figure online)

depth sensors where the GT optical flow can be calculated using the depth map and the GT rotation and translation of the camera (available in the dataset) (Zhu et al. 2018).

It should be noted that the GT in the dataset sometimes contains mismatches between the events and the depth, which can be a source of error in the training process. Since the flow is generated on the basis of the depth map, it will also contain artifacts. The network expects that, if it sees visible events in the input stack, there should be a relation for it in the paired intensity, depth, or flow. Therefore, following the intensity reconstruction scheme, we slightly removed the effect of these artifacts by omitting the extreme training samples that did not represent a one-to-one presentation of the events to the depth or intensity.

Furthermore, the car hood was cropped to prevent similar errors due to the reflection of light on the shiny surface of the hood. A sample training set is shown in the last row of Fig. 5. For the real-world dataset on the depth and optical flow tests, we used the outdoor_day_2 sequence of the MVSEC dataset (more than 12K frames) and tested it on the outdoor_day_1 and indoor_flying sequences for both the depth and the optical flow. Stacking was performed by utilizing SBN, with 60K events and three frames per stack. The output results are shown in Figs. 17 and 16 of Sect. 5.

4 Experiments on Image Reconstruction

Experiment setting. In this section, we present both the qualitative and the quantitative results for the event-to-image reconstruction using the datasets mentioned in Sect. 3.3.1. We mainly focus on how the stacking methods or the network parameters affect the reconstruction of the intensity images under different conditions. Meanwhile, we compare our outputs with those of the existing methods. Depth and optical flow will be evaluated separately in the following Sect. 5. We stacked 60K events to create event images together with their corresponding APS images on the basis of the precise timestamps.

We tested our method on both scenes with normal illumination and HDR scenes. From both the real-world and the simulated datasets, we randomly chose 1K event images together with their corresponding APS (GT) images.

It is worth noting that, since the real-world dataset does not include GT images for training and testing, we used its *APS images as the GT* for training purposes. However, the APS images themselves suffer from motion blur and low dynamic range. Thus, using the APS images might not be the best way for training and also for evaluating the results. For this reason, we prepared the training APS images as described in Sect. 3.3.1 and evaluated the results (compared the results with the APS images) using the structured similarity indexing method (SSIM) and the feature similarity (FSIM) method (Zhang et al. 2011). On the other hand, to assess the similarity between the GT and the generated images on the synthetic datasets, we matched each GT image with the corresponding reconstructed image with the closest timestamp, as mentioned in Scheerlinck et al. (2018). The SSIM, FSIM, and peak signal-to-noise ratio (PSNR) were adopted to evaluate the non-HDR scenes and the scenes with high-quality GT images.

Interestingly, since the proposed method can fully exploit the advantages of event cameras such as their high temporal resolution and HDR, it can generate more visually acceptable HDR images than APS images and very high frame rate videos, as mentioned in Sect. 3.1, thus greatly increasing the usefulness of the proposed method. We qualitatively show the outputs of our approach on the challenging real-world sequences in Fig. 6.

4.1 SBT vs. SBN

We compared the performances of the two event stacking methods, *i.e*let@tokeneonedot, SBT and SBN, on the real-world intensity datasets. The 17K stacked event images and the APS image pairs were used for the training, where we set Δt for SBT to 0.03 s and the number of events in one stack to 60K for SBN (*e.g*let@tokeneonedot, 20K events per frame if n=3); the number of frames (n) in one stack was set to 3 for both methods to show the effect of a stacking method. Note that the given numbers were an approximate number of events per stack and that, if they were not divisible by the number of channels (*e.g*let@tokeneonedot, 5K events with 3

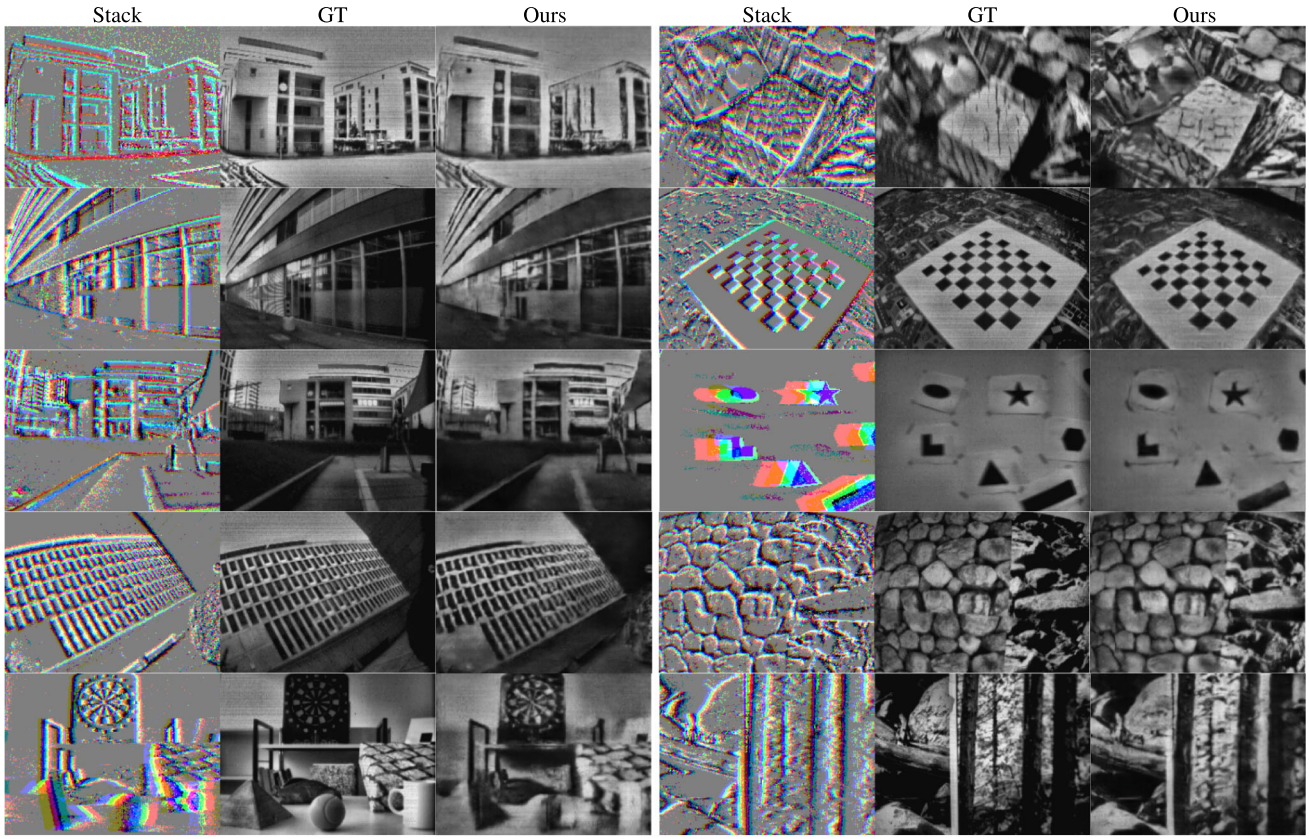

Fig. 6 Reconstruction of relatively realistic intensity images of real-world scenes from pure events using our proposed method. We were able to reconstruct images with comparable quality to that of APS frames in multiple scenarios with different camera speed and movements, lighting conditions, and number of triggered events in indoor and outdoor scenes

frames), the greatest integer less than or equal to that division (floor) was considered.

Table 2 shows the quantitative results of using the SBN and SBT stacking methods on the real-world dataset. SBN produced better results in general owing to its asynchronous behavior to time, as explained in Sect. 3.1. SBN includes events of the whole scene rather than the latest movements of a specific time slot, which results in better reconstructed intensity images. For a qualitative comparison, refer to Wang et al. (2019). However, it should be pointed out that SBN may be less useful for static background scenes in which a fast-moving object (*e.g*let@tokeneonedot, a car) passes by and consumes all the events considered for a stack. In such cases, SBT can still recover the background information but may lose the details of the fast-moving object.

Table 2 Quantitative evaluation of SBN and SBT ($n = 3$) on our simulated event dataset

	PSNR	FSIM	SSIM
SBN	25.12±3.18	0.88±0.06	0.80±0.12
SBT	22.92±2.92	0.85±0.07	0.75±0.14

SBN quantitatively performed better in all of the metrics

4.2 Evaluation on the Simulated Datasets

In Sect. 4.1, we investigated the performance potential of our method on different stacking methods. SBN can produce higher quality outputs than those of SBT, and, therefore, we used it to stack the events and to show the effectiveness of our method on the simulated datasets. Since ESIM produces noise-free APS images with the corresponding events for a given image, the APS images can be regarded as the GT, thus making the quantitative evaluations more feasible. In addition, we chose the number of channels as $n = 3$ following the performance evaluation on the number of events vs. the number of frames in Wang et al. (2019). The total number of events in one stack was set to 60K.

The proposed network is usually trained with a fixed number of events per stack. However, it can also be trained with multiple events per stack, such that it can be more robust to a diverse number of input events. Furthermore, once trained with a specific number of events, the network can still be used for slightly further or less number of events

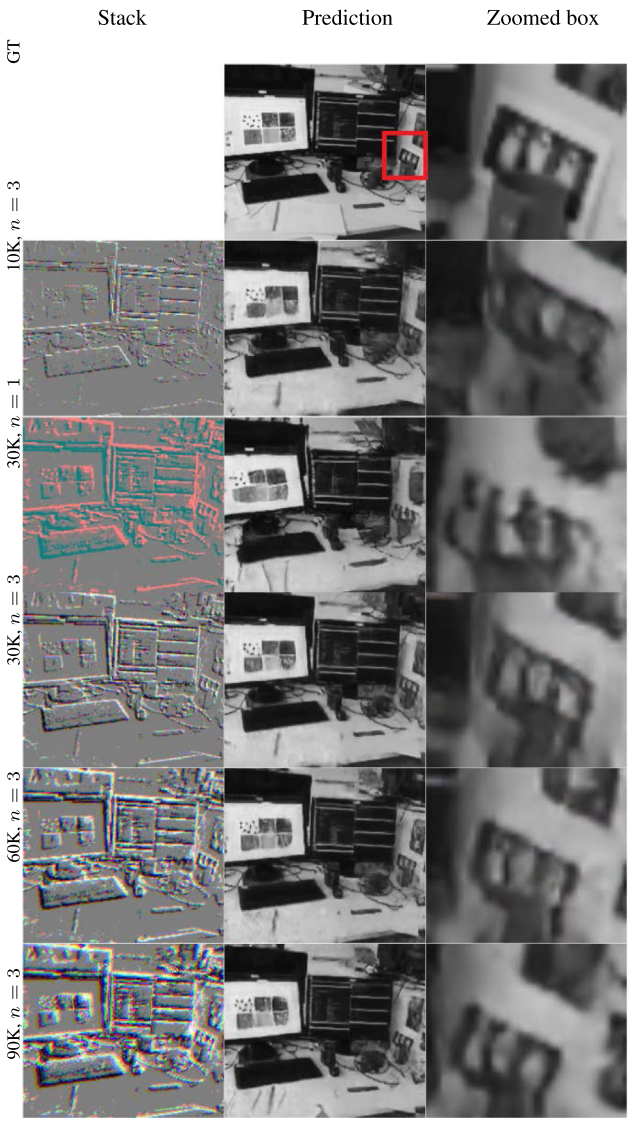

Fig. 7 Qualitative presentation of the effect of changing the number of **a** frames per stack and **b** total number of events per stack. When keeping the events fixed (30K events) and changing the number of frames from $n = 1$ to $n = 3$ (case a), we can see that $n = 3$ frames has a better output. When changing the total number of events per stack to 10K, 30K, 60K, and 90K while having a fixed number of 3 frames (case b), we can see that adding more events up to 30–60K improves the quality but adding more does not really help and might also degrade the quality partially. The GT together with a zoomed version of the prediction/GT enclosed by the red box is provided for comparison. The stacking method is SBN

in a stack. To show this feature, we trained our network using $\sim$ 5000 events stacked as three frames ($\sim$ 1660 events per frame) on the real-world event dataset. We then tested the "HDR_boxes" scene using different numbers of events stacked (1K, 2.5K, 5K, and 10K events), as shown in the top rows of Fig. 9. The test results showed that the network can reconstruct reasonable outputs with different numbers of events in a stack at the testing phase.

However, too few events force the network to hallucinate some results on flat locations with fewer events, as shown in Fig. 9. These hallucinations tend to be from the GAN term, as mentioned in Sect. 4.3, and they are in the form of high-frequency outputs such as lines and textures. Providing an extra number of events at the testing phase does work also, although it may lead to lower quality outputs. If too many events are fed to a stack, there is a chance of overwriting the previous events, which will produce erroneous outputs. Although the network itself can tolerate an excessive number of events to some extent, if more events are planned to be used, it is better to increase the number of frames in each stack. For our experiments, once the network was trained with E events in a stack, the minimum number of events required to provide acceptable reconstructions for testing should be in the range of $E/2$ to a maximum of $E \times 2$ events.

We further show the effect of the number of channels and events per stack on the quality of the reconstruction in Fig. 7. In our experiments, we changed multiple factors including the number of frames per stack (1 and 3) and the total number of events in each stack (10K, 30K, 60K, and 90K events). Remarkably, we observed that the performance increased by increasing the number of events in a stack only if we had enough frames available to accommodate the extra events (*e.g.*let@tokeneonedot, $n = 3$ can include more events); otherwise (*e.g.*let@tokeneonedot, for $n = 1$), the performance dropped as more events were added because there is a probability that newer events will overwrite the previous events.

Figure 10 shows some of our intensity image reconstruction results. It can be seen in the figure that the reconstructed intensity images are very close to the GT images. Since there exists no real-world noise in the events and the training set covers a wide variety of scenes, the reconstructed images are more realistic. Furthermore, the camera movements are not parallel to the scene gradients, which allows changes in the scene to be sensed and events to be easily triggered. Moreover, some of the APS images are in focus compared with the real-world reconstructed images, which has a positive impact on the reconstruction of sharp and realistic images.

4.3 Impact of the Loss Function

In this section, we discuss the results of the systematic ablation study we conducted on different combinations of loss functions that affect the quality of reconstructed intensity images. The quantitative results are shown in Table 3, whereas the qualitative results are shown in Fig. 8. In terms of PSNR, the $\mathcal{L}_1$ norm obtained higher values. However, we used cGAN + $\mathcal{L}_1$ throughout our experiments since it obtained a better perceptual (LPIPS) score in the ablation study, as shown in Table 3. A higher PSNR does not always mean better output quality. It just presents the similarity to APS images (used as the GT) suffering from noise, motion

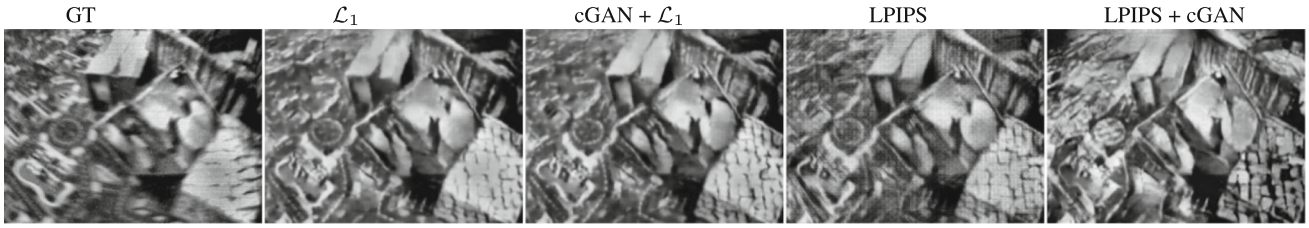

Fig. 8 Effect of the different loss functions on a sample real-world input. $\mathcal{L}_1$ does not preserve the high-frequency details. cGAN+$\mathcal{L}_1$ preserves the high-frequency details without adding unwanted artifacts. Unwanted artifacts were added when the learned perceptual image patch similarity (LPIPS) function or the combination of cGAN+LPIPS were utilized

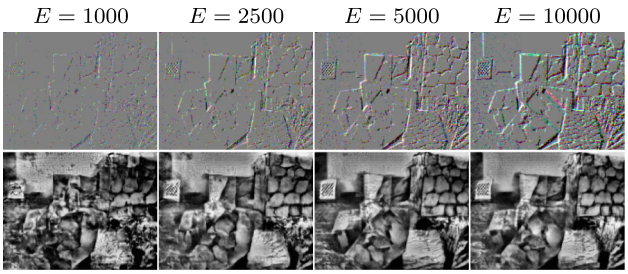

Fig. 9 Stacking different numbers of events as an input stack of three frames trained with $E = 5000$ events. Although the algorithm can create outputs with a very low number of events, empirically, however, feeding more than $E/2$ and less than $E \times 2$ events per stack leads to better output results. Overfeeding with an excessive number of events does not necessarily enhance the reconstructed outputs

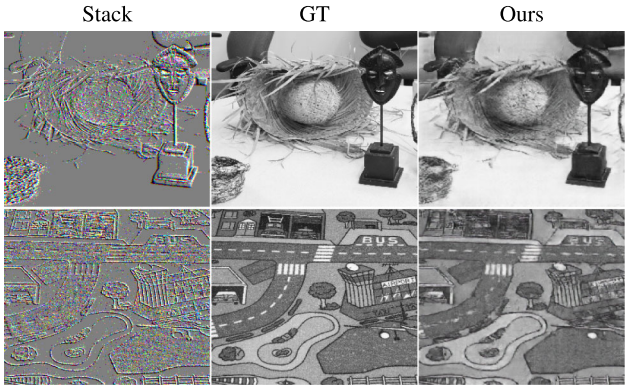

Fig. 10 Reconstruction of relatively realistic intensity images of simulated scenes from pure events using our proposed method without the presence of real-world noise

blur, and low dynamic range. For example, a higher PSNR means that the result with $\mathcal{L}_1$ in Fig. 8 is more similar to the low-quality APS image. Since we aimed to reconstruct images to make them look more realistic and better than APS images (used as the GT), we did not use $\mathcal{L}_1$ but used cGAN+$\mathcal{L}_1$ instead. Moreover, the $\mathcal{L}_1$ norm by itself smoothens the image and averages out the details.

Furthermore, using LPIPS as the loss function generates slightly better PSNR values; however, in terms of the LPIPS similarity, it obtains worse results. Additionally, in our experiments, the LPIPS loss caused high-frequency artifacts as a fuzzy pattern, which is not visually pleasing. Adding the LPIPS loss and the adversarial loss together does not improve the performance visually or qualitatively, most probably because the model cannot converge easily between these two terms. The effectiveness of LPIPS has been studied in many image translation and super-resolution methods (Oudeeraa et al. 2019; Wang et al. 2020). Note that L1 loss is a pixel-wise loss where each pixel in the generated image is directly compared with each pixel in the GT image. Using L1 shows improved performance and convergence. However, L1 loss may ignore the high-frequency details. The GAN loss helps generate more realistic images as it predicts the probability that real images are relatively more realistic than fake ones. Thus, in our method, using L1 loss and GAN loss together overcomes the difficulty of only learning perceptual quality. Overall, cGAN+$\mathcal{L}_1$ showed the best perceptual performance among the loss terms while producing the most reasonable image outputs. Therefore, we used it throughout our experiments and its results unless otherwise explicitly stated (Figs. 9, 10).

Although our GAN-based method shows visually pleasing outputs and has better perceptual quality than that of other loss terms, it tends to obtain lower PSNR values. We argue that this is because the most dominant difference between

Table 3 Ablation of the effect of the standard $\mathcal{L}_1$ loss function, cGAN, LPIPS, and cGAN+LPIPS on event reconstruction quality

	$\mathcal{L}_1$	cGAN+$\mathcal{L}_1$	LPIPS	cGAN + LPIPS
LPIPS (↓)	0.426	**0.395**	0.419	0.439
PSNR (↑)	**20.84**	18.57	18.63	17.12
SSIM (↑)	**0.644**	0.52	0.51	0.46

Table 4 Comparison between our event-to-intensity reconstruction method that incorporates GANs (EG) and the E2Vid method using statistical (SSIM and MSE) and perceptual (LPIPS) metrics

Sequence	SSIM (↑)			MSE (↓)			LPIPS (↓)		
	E2Vid	EG w/	EG w/o	E2Vid	EG w/	EG w/o	E2Vid	EG w/	EG w/o
dynamic_6dof	**0.50**	0.48	0.41	0.08	**0.03**	0.05	**0.43**	0.45	0.50
boxes_6dof	**0.63**	0.45	0.43	0.04	**0.03**	0.04	**0.36**	0.48	0.51
poster_6dof	**0.68**	0.61	0.57	0.04	**0.01**	0.02	**0.32**	0.42	0.45
shapes_6dof	0.44	**0.56**	0.48	0.10	**0.03**	0.05	0.53	**0.51**	0.58
office_zigzag	0.50	**0.67**	0.54	0.05	**0.01**	0.02	0.44	**0.36**	0.43
slider_depth	**0.61**	0.54	0.47	0.06	**0.02**	0.03	0.42	0.42	0.47
calibration	0.52	**0.67**	0.52	0.04	**0.01**	0.02	0.47	**0.42**	0.49
Avg.	0.55	**0.57**	0.49	0.06	**0.02**	0.03	**0.42**	0.43	0.49

We used both of our training settings with (w/) and without (w/o) extreme cases of motion blur and low-dynamic-range locations. Our method showed better performance in terms of SSIM in three of the sequences and also on average. Although not trained for LPIPS, our method still achieved comparable LPIPS scores in some of the sequences. MSE might not be a great metric, and all methods were almost comparable in performance. As expected, "Ours w/" mimics APS frames better than "Ours w/o" using all the metrics. The best performance is bolded, and the runner-up is underlined

the reconstructed images and the real intensity images is the high-frequency information, where the reconstructed images obtained by minimizing the pixel-wise errors lacked high-frequency details. The simplest way for a discriminator to distinguish reconstructed images from real intensity images could be simply by inspecting the presence of high-frequency components in a given image, and the simplest way for a generator to fool the discriminator would be to put arbitrary high-frequency noise into the resulting images.

4.4 Comparison with Relevant Works

We quantitatively compared our method with the deep learning based Events-to-Video (E2Vid) method (Rebecq et al. 2019) and further show a qualitative comparison in Fig. 11. Furthermore, in comparison with previous optimization-based methods, we predicted the intensity images from the *face, jumping, and ball* sequences (Bardow et al. 2016) and compared them with the results of Munda et al. (2018) and Bardow et al. (2016) and report them in Table 5. We followed the convention of comparing with statistical methods using the BRISQUE score (lower is better) because no GT image was available for these sequences.

As E2Vid is a concurrent work, we first point out the similarities and differences between it and our work. E2Vid is designed for video reconstruction from events and is quite similar to our method in different aspects. The main architecture of both is U-net (Ronneberger et al. 2015) with skip connections, and they both follow almost the same architectural details in decoding and encoding. As for the differences, E2Vid uses a recurrent connection to propagate the intensity information over time since its priority is video reconstruction. Therefore, the input information is also different in comparison.

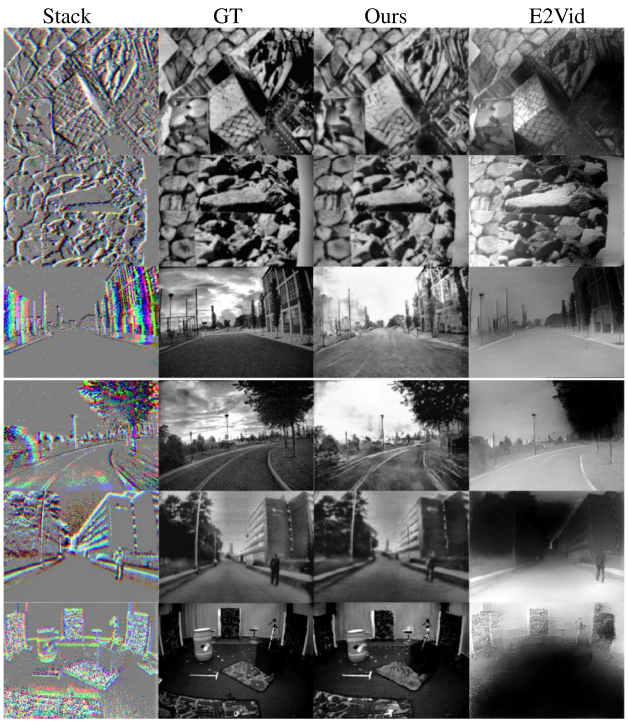

Fig. 11 Visual comparison between our event-to-intensity reconstruction method and the E2Vid method (Rebecq et al. 2019). Both methods were able to create high-quality outputs. Unlike E2Vid, our reconstructed images did not create dark regions, which could be a result of previous error propagation in their method

However, when looking at the results of E2Vid, we can find faded black or white regions following the video. This could be due to the propagated error from the recurrent network. Moreover, the central loss function of these two methods is based on a learned perceptual image patch similarity (LPIPS) (Zhang et al. 2018) distance. Instead of finding the normal

Table 5 Quantitative comparison of our method (EG) with those of Bardow et al. (2016) and Munda et al. (2018)

Sequence	Face	Jumping	Ball
Bardow (Bardow et al. 2016)	22.27±8.81	29.39±7.27	29.37±9.61
Munda (Munda et al. 2018)	27.29±7.27	48.18±6.70	34.98±9.31
EG	**14.15**±4.22	**16.69**±4.32	**13.37**±5.25

The reported numbers are the mean and standard deviation of the BRISQUE measure applied to all reconstructed frames of the sequences. Our method showed better BRISQUE scores for all sequences

$\mathcal{L}_2$ distance between an input image and its GT pair, LPIPS feeds an input image to another pretrained neural network, *e.g*let@tokeneonedot, a VGG net (Simonyan and Zisserman 2015). Then, it calculates the $\mathcal{L}_2$ distance between that feature and the features of the GT obtained in the same manner. The logic for training with such a criterion is that, if the features of the two images are perceptually similar, then, those two images will also be alike. In contrast, we mainly utilize a conditional GAN together with the $\mathcal{L}_1$ norm to minimize the distance. To quantitatively relate E2Vid with our method in a uniform representation, we followed the selected sequences together with the evaluation metrics: SSIM (higher is better), mean squared error (MSE; lower is better), and LPIPS (lower is better). As mentioned previously, these real-world images suffer from extreme cases of low dynamic range and motion blur if we consider the APS frames as the GT. To discuss this aspect, in our comparison, we used two different training schemes: one is trained with such extreme cases (Ours w/) and the other one without such samples (Ours w/o). In extreme cases, the network either learns to blur images in similar situations or does not consider events in locations where the dynamic range is low. A sample of such cases is shown in Fig. 1. It should be noted that E2Vid was trained on the simulated events only, whereas both of our methods were trained using our real-world training set mentioned in Sect. 3.3.

Table 4 compares the performances of these two methods in multiple real-world scenes. Depending on the sequence, both of our training methods obtained similar numbers in terms of SSIM; however, our method (w/) achieved a slightly higher score on average, which is the same case as that in MSE. However, for LPIPS, we obtained a slightly lower score in comparison; however, it should be noted that, although we did not train on the basis of the LPIPS score, we still had sequences that showed comparable scores.

4.5 Events to HDR Image

Event stacks have rich information for HDR image reconstruction. In many cases, some parts of the scene are not visible in the APS image because of its low dynamic range, although many events might be triggered, such as the region

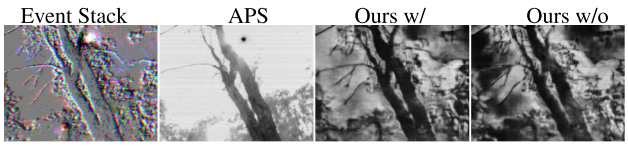

Fig. 12 HDR imaging against direct sunlight (extreme illumination) sequence (Scheerlinck et al. 2018). The predicted output trained under extreme light conditions (w/) is sharper without any unwanted black areas than that trained without such conditions (w/o)

under the table in Fig. 1 or the checkerboard pattern at the top-left part of the stacked image in Figs. 2 and 13 . Those examples are from dark illumination scenes, but normal cameras also fail in excessive illumination, such as that shown in Fig. 12.

We took one further step in reconstructing images, as shown in Figs. 13 and 12 , where we show the ability to reconstruct images in HDR scenes and to recover details that cannot be recovered using APS frames. This reflects one advantage of the event camera, namely, the higher dynamic range. The reconstructed images can preserve the structure of the scene in all areas. The trained network using SBN performs better by producing superior results in HDR scenes as well. This characteristic can be very useful for autonomous driving or tracking applications under extreme illumination conditions. One example is when a car enters a tunnel; standard cameras can get blacked out owing to low illumination or when exiting the tunnel. The camera can become saturated by the excessive amount of light suddenly flooding inside the aperture. The same example holds for a robot vacuum cleaner going under a piece of furniture while moving on the floor and experiencing major illumination changes.

The proposed method creates well-structured images even with a small number of events in the input image as in the time t_0, which is the starting state of the sequence in Fig. 13. Unlike the method of Reinbacher et al. (2016), our method is not related to any previous state; therefore, from a few events, our method can create acceptable outputs. Furthermore, since their method uses the previous states, it sometimes propagates previous erroneous parts to its next frames as a shadow-like or a hollow artifact.

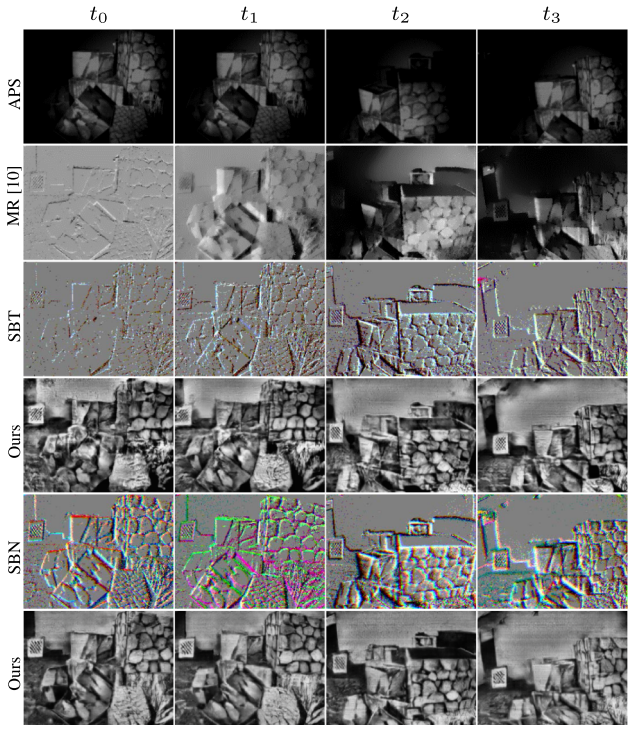

Fig. 13 Examples of HDR image generation. Reconstruction from the "hdr_boxes" scene (Mueggler et al. 2017). Our SBT and SBN results clearly recovered more HDR details from the start of the sequence without blur or any shadow such as artifacts. A slight difference in performance between SBT and SBN is visible

4.6 Temporal Consistency

Our method is mainly designed for reconstructing intensity images from events, not from sequences, to be considered as a consistent video. However, when events have overlapping common data in nearby frames in a sequence, they will be more consistent over time. To quantitatively evaluate the consistency of the reconstructed intensity images in a sequence, we used the temporal stability metric from Lai et al. (2018). This metric is based on the flow warping error between two consecutive frames of (F_t, F_{t+1}), as defined in Eq. 4.

$$E_{warp}(F_t, F_{t+1}) = \frac{1}{\Sigma_{i=1}^{N} M_t^{(i)}} \Sigma_{i=1}^{N} M_t^{(i)} ||F_t^{(i)} - \hat{F}_{t+1}^{(i)}||_2^2. \quad (4)$$

Here, $\hat{F}_{t+1}$ is the warped frame of F_{t+1} and $M_t \in \{0, 1\}$ is the non-occlusion mask based on Ruder et al. (2016), to ensure that the calculations are applied only over the non-occluded regions. This error is averaged over all the frames in each sequence to give the final error. We used the same sequences shown in Table 4 and report the warping error for our proposed method trained with (w/) and without (w/o) the extreme cases in Table 6. The GT optical flow was calculated on the basis of the APS frames. Note that the APS frames themselves also have a small but non-zero warping error, and we report them as a reference. We also calculated the

Table 7 Impact of different loss functions on the depth and optical flow estimation using the simulated datasets

	cGAN+$\mathcal{L}_1$	cGAN+$\mathcal{L}_2$	$\mathcal{L}_1$	$\mathcal{L}_2$
AEE	0.102	0.108	0.103	0.105
Outlier	0.132	0.143	0.128	0.110
Abs Rel	0.047	0.053	0.04	0.036
RMSE log	0.025	0.027	0.017	0.013
SIlog	0.044	0.044	0.016	0.013
Acc: $\delta < 1.25$	0.929	0.925	0.965	0.968
Acc: $\delta < 1.25^2$	0.962	0.962	0.986	0.989
Acc: $\delta < 1.25^3$	0.977	0.978	0.993	0.995

Outlier is the percentage of points having an average end-point error (AEE) of >3 pixels

Table 6 Temporal consistency error on diverse real-world sequences

$E_{warp}(\downarrow)$	APS	EG w/	EG w/o	EG w/ +BP	EG w/o +BP	E2Vid (Rebecq et al. 2019)
dynamic_6dof	0.61	7.45	16.88	4.07	7.23	8.78
boxes_6dof	1.81	19.54	23.93	11.42	12.8	15.69
poster_6dof	1.1	14.5	27.61	9.23	14.25	17.74
shapes_6dof	0.44	2.45	7.58	1.55	2.61	16.66
office_zigzag	0.08	1.31	1.75	0.64	0.77	0.72
slider_depth	0.02	0.33	0.47	0.13	0.16	0.19
calibration	0.36	5.49	6.14	3.05	3.24	2.99
Average	0.63	7.29	12.05	4.3	5.86	8.97

We quantitatively evaluated both of our training methods (EG) with extreme cases (w/) and without including such cases (w/o) in the training. The reported numbers can be further reduced to the numbers in parentheses by applying blind post-processing (BP) methods (Lai et al. 2018). EG w/ shows comparable results to those of E2Vid on average, and EG w/o shows a slightly higher loss. By applying BP to both of our training settings (w/ and w/o), our method showed temporally consistent results that were better than those of E2Vid

temporal loss in Eq. 4 on the sequences from E2Vid and showed the results in Table 6. For consistency with Table 4, the stacking method is SBN without any overlapping events.

Table 6 shows that, compared with the warping errors obtained from the source of our optical flow calculations (the APS frames), the temporal consistency loss obtained by our method was higher, although the calculated loss was within an acceptable and comparable range to that of E2Vid. Unfortunately, areas in the event stack, where no events are fired, are prone to fluctuations when moving from one reconstructed frame to another. This may not be ideal when creating a video; however, there are blind postprocessing (BP) methods such as the method of Lai et al. (2018), which, when given the input event stack and the reconstructed output, can further refine the outputs. We also leveraged that algorithm for our reconstructed images and report them as EG (w/ or w/o) + BP.

Our method took about $65ms$ on average to synthesize a three-frame stack holding $E = 20,000$ events with a spatial size of 256×256 pixels when testing on a single Nvidia Titan-Xp GPU. The additional postprocessing step using the same settings took 4 ms. Note that the blind postprocessing method takes the event stacks as inputs together with the reconstructed outputs and makes more temporally consistent outputs. Therefore, if we aim for higher frame rates, which creates event stacks with high overlapped shares, the BP can still be applied. Furthermore, if the reconstruction method creates many black regions like what EG w/ or E2Vid does, warping a black region to another black region will result in a lower warp error, indicating that the sequence of images is consistent. Although this statement is true, it should be considered together with the image reconstruction quality metrics when comparing the two methods. In this setting, EG w/o that creates more changes based on the event stacks will have a higher warp error as the intensity of these regions will not be consistent over time.

4.7 Color Event Cameras

Following the intensity image reconstruction from a color event camera (Moeys et al. 2017), we also show the potential of our method to synthesize color images as well. We used the sequences from Scheerlinck et al. (2019), which were captured using a color DAVIS 346 event camera that utilizes a Bayer pattern filter on specific pixels, making them more sensitive to certain colors. This creates four separate streams of red, green, blue, and green again in a combined stream, sacrificing the spatial resolution to a quarter of the original grayscale size. Each sub-stream is given separately to our network to predict each color channel accordingly, which is combined back together to make a color output. The input events, together with the reconstructed outputs using the previous training for real-world grayscale intensity images, are visualized in Fig. 14 for reference. Our method can produce colors similar to those of the GT images.

Fig. 14 Presentation of the capability of our method in reconstructing color images from color events. Utilizing the data from a color event camera (Scheerlinck et al. 2019), we separately predicted the RGBG events and combined them as the color image. Separate input channels are shown in regard to their color together with the combined color output and GT

5 Experiments on Direct Depth and Optical Flow

In this section, we present the experimental results on the direct depth and optical flow estimations. The geometric and temporal interrelations between events make it possible to recover the depth and optical flow directly from the event streams without any additional requirements. Therefore, there is no need to reconstruct intensity images first and then recover the flow or depth as a downstream application. Generally, as events are sparse, a one-to-one mapping of the depth or flow estimation is also sparse. However, our proposed framework can achieve a dense estimation.

5.1 Optical Flow and Depth Estimation

First, we show that our proposed event stacking methods (*i.e.*let@tokeneonedot, SBT and SBN) can also be used for optical flow and depth estimation. To this end, we show in Fig. 15 that the moving objects and the background scene leave timely coded traces of the movement in each different stack. These traces are both from the global or rigid flow, referring to the camera movement. The local flow refers to the movement of the objects inside the scene. Note that the optical flow is coded on the basis of the direction and length of the per-pixel movement to the hue and saturation, respectively, throughout the rest of this paper.

We show a simple planar scene with the same size as that of black filled circles having the same distance as that of the background, which is extended by adding three horizontal bars shaping a staircase in the second scene. In the simulated sequences, the camera moves freely in 6DoF over the scene. In Fig. 15, the dot pattern (filled black circles) on the surface creates positive (red) and negative (blue) events depending on the movement of the camera and on the topology of the

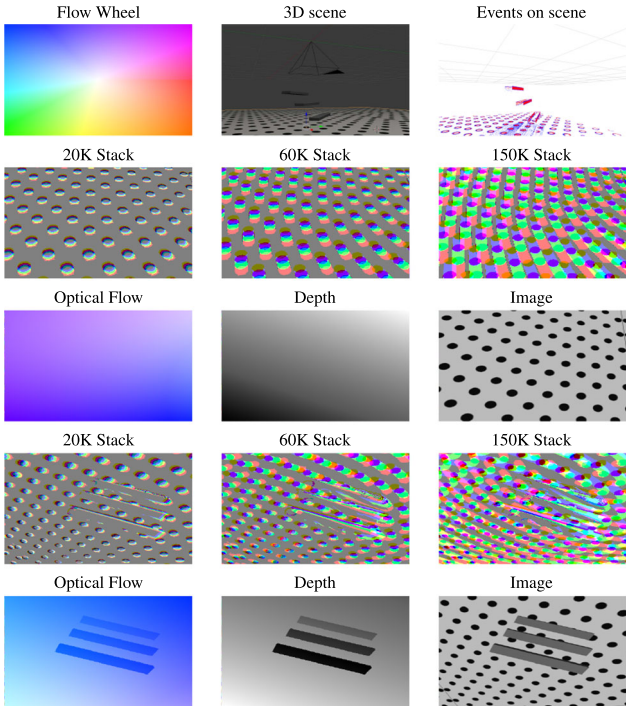

Fig. 15 Logic behind the optical flow and depth estimation in a stack of events for 20K, 60K, and 150K events in a three-channel SBN stack over a planar scene. The following tail of events visualizes the relative motion between an observer and a scene over time, which is called the optical flow. The depth can also be inferred in regard to the different sizes of the tails in different areas of the stacked events. The original grayscale image of the scene at the end of the stacking period is also presented for better visual comparison

scene. The samples show how each segment of the image has moved in respect to its previous location.

Various numbers of events including 20K events, 60K events, and even 150K events were added to the three-channel stacks. Note that, in our typical experiments, we do not use the stacks with 150K events; we only utilized them to show the relation of the stacked events to the optical flow and depth estimation. Although we mainly show three-channel stacks in our figures, our method can be designed and trained using any number of channels as input stacks, as explained in the previous sections. For a specific dot in Fig. 15 (*e.g.*let@tokeneonedot, in the extreme case of 150K events in a three-channel stack), its current location is usually dark purple and its tail colored in green and orange shows its previous locations. This indicates that the timestamp of purple is bigger than those of the other two colors: $T_{purple} > T_{green} > T_{orange}$.

On the basis of the event stacks, we can see the camera moving upward and slighting to the right. It is worth noting that too few events have the risk of losing the optical flow in specific regions and that too many events have the risk of overriding previous events and destroying the path of the optical flow, resulting in wrong estimations.

Depth is closely related to optical flow in terms of the logic behind the estimation process. Imagine two lines sweeping in the horizontal direction with the same speed and both parallel to the image plane but one is closer to and the other one is farther from the image. The line that is closer to the camera will create many more events in the same period because it changes more pixel values on its edge along its sweep line and will have a longer tail of events. If we accumulate them at a specified time, this can be used as a measure to estimate the depth. As shown in Fig. 15, the longer tail of events are for the dots close to the camera and the smaller ones are for those far from it. The same principle applies to more complicated scenes.

5.2 Impact of the Loss Function

The simulated datasets mentioned in Sect. 3.1 were used to perform our ablation study of the loss function on the depth and optical flow. Fig. 16 shows the outputs of the depth and flow reconstruction for a sample test scene using the different loss functions. The quantitative comparisons are also shown in Table 7. Like in the intensity image reconstruction, the $\mathcal{L}_1$ and $\mathcal{L}_2$ terms by themselves do an excellent job in predicting the overall dense flow of the scene; however, they tend to blur out the fine details. On the other hand, cGAN was able to recover very fine details with much clearer edges; however, it might create fine high-frequency edges in some regions. The reported numbers are the average endpoint error (AEE) computed in pixels, representing the distance between the end points of the predicted (f') and the GT (f) flow vectors:

$$AEE = \frac{1}{n}\Sigma\|\overrightarrow{f} - \overrightarrow{f'}\|_2. \tag{5}$$

The percentage of outliers reported in the table is considered as the number of points with AEE >3 pixels. For a quantitative scale-invariant depth metric, we used the accuracy (Acc), scale invariant logarithmic error (SILog), absolute relative distance (AbsRel), and logarithmic mean squared error (RMSELog):

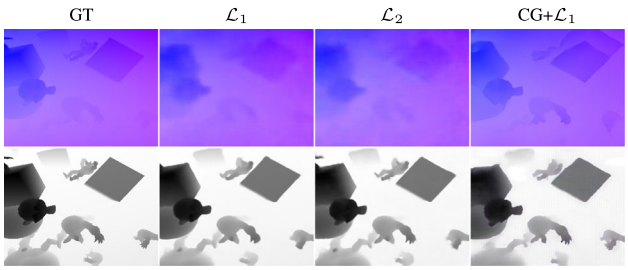

Fig. 16 Estimation of the depth (bottom) and optical flow (top) using different optimization functions on the simulated inputs. Using GANs helps in preventing fading details

Table 8 Quantitative comparison of the optical flow prediction

	outdoor_day_1				indoor_flight_1				indoor_flight_2				indoor_flight_3			
	(Zhu et al. 2019)	(Zhu et al. 2018)	Ours	Ours M	(Zhu et al. 2019)	(Zhu et al. 2018)	Ours	Ours M	(Zhu et al. 2019)	(Zhu et al. 2018)	Ours	Ours M	(Zhu et al. 2019)	(Zhu et al. 2018)	Ours	Ours M
AEE ↓	0.32	0.49	0.23	0.17	0.58	1.03	0.28	0.20	1.02	1.72	0.30	0.23	0.87	1.53	0.31	0.23
Outlier ↓	0.0	0.2	0.1	0.1	0.0	2.2	0.1	0.1	4.0	15.1	0.1	0.1	3.0	11.9	0.1	0.1

Since our method predicts a dense estimation, we mask it with the regions where actual events are triggered and label it as "Ours M." Compared with E2FDE (Zhu et al. 2019) and EV-FlowNet (Zhu et al. 2018), which were also trained using only the sequences from outdoor_day_2, our method produced optical flow predictions with better quality

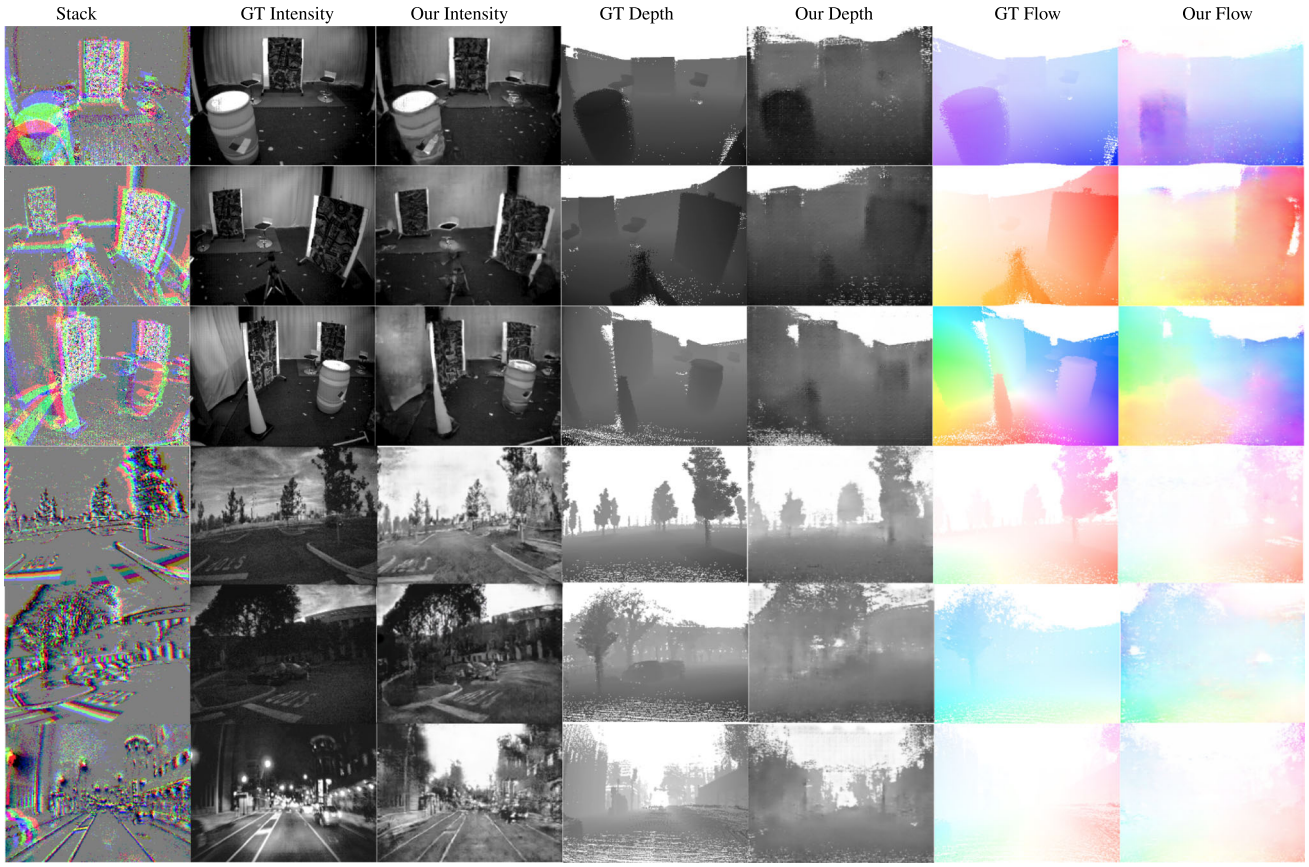

Fig. 17 Qualitative presentation of our intensity, depth, and optical flow estimations solely from events through minor modifications on the last layer of our proposed network using separate training settings. Our model is versatile as it can be generalized to other tasks by estimating the target task with high quality and with a dense representation

$$Acc = \%\ of\ d_i\ s.t.\ \max(\frac{d_i}{d_i'}, \frac{d_i'}{d_i}) = \sigma < th. \tag{6}$$

$$SILog = \frac{1}{n}\Sigma a_i^2 - \frac{1}{n^2}(\Sigma a_i)^2, a_i = \log d_i - \log d_i'. \tag{7}$$

$$AbsRel = \frac{1}{n}\Sigma \frac{\|d - d'\|}{d'}. \tag{8}$$

$$RMSELog = \sqrt{\frac{1}{n}\Sigma \|\log d - \log d'\|^2}. \tag{9}$$

On the basis of the quantitative results, we can see that cGAN $+\mathcal{L}_1$ reconstructed the output flow estimations better in the ESIM data, and, therefore, we utilized it in our experiments.

We used the real-world driving dataset MVSEC and separately created all three outputs of the intensity depth and optical flow from one input, namely the event stack. The results are shown in Fig. 17 for the challenging indoor, outdoor, day, and night scenes. The top three sequences from MVSEC were indoor_flight followed by outdoor_day1 and outdoor_night. One can see that our method can predict the

intensity, dense depth, or dense optical flow with high quality comparable to that of the GT data in each class.

5.3 Comparison to Relevant Works

We compared the depth and optical flow predictions of our approach with those of the recent methods such as E2FDE (Zhu et al. 2019), EV-FlowNet (Zhu et al. 2018), and ECN (Ye et al. 2018), which are well-designed deep learning-based frameworks. Following the same settings and using the same real-world dataset, we trained our network on the event sequence of outdoor_day_2 and test on outdoor_day_1 and three indoor_flight sequences from the MVSEC dataset. Table 8 shows the results for the optical flow estimation, where the comparison results with E2FDE and EV-FlowNet are listed. As ECN used 80% of the indoor scenes for the training and the other 20% and the outdoor_day_2 sequence for the testing, we could not compare our method with it. We also included a qualitative comparison with EV-FlowNet (Zhu et al. 2018), whose code is publicly available, in Fig. 19 to show the difference between these methods in terms of optical flow estimation.

Unlike E2FDE (Zhu et al. 2019) and EV-FlowNet (Zhu et al. 2018) in Table 8, our method reported the dense optical flow predictions even on regions where events were not triggered. Therefore, we also reported the performance of our algorithm by masking the flow results to locations where events existed, labeled as "Our M." In terms of the AEE, our method showed superior performance specifically on the outdoor_day_1 sequence, which was within the same domain of data used to train the network. We also obtained better results when predicting the optical flow of indoor_flight scenes, although they were cross-validated for the indoor domain. The masked outputs usually had even lower reported errors, showing the importance of the events triggered over the scene (Table 9).

Using the simulated datasets only for the training while validating the real-world results did not obtain comparable results; however, it is considered a common issue in cross-dataset validation. Sample outputs are shown in Fig. 18. Our simulated datasets for the depth and flow did contain 3D objects; however, they were all static objects with simple shapes in which the camera was only moving. Making such a dataset is more challenging and is out of the scope of this work. Therefore, the dataset does not include complicated details and has scenarios where the objects and camera are simultaneously moving. This may slightly improve the predictions as it is more similar to real-world dataset on this aspect.

One problem that prevents cross validation in general from the simulated datasets used for the training to the real-world dataset used in the testing and vice versa is the presence of noise, although it is not only limited to that. As

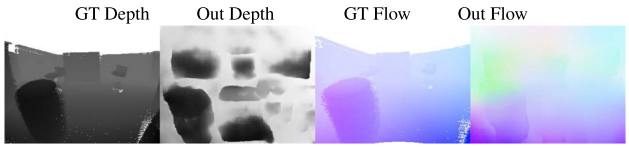

Fig. 18 Training using simulated events only and cross-evaluation on the real-world dataset. Unfortunately, the depth and flow outputs do not generalize well using only the simulated event data

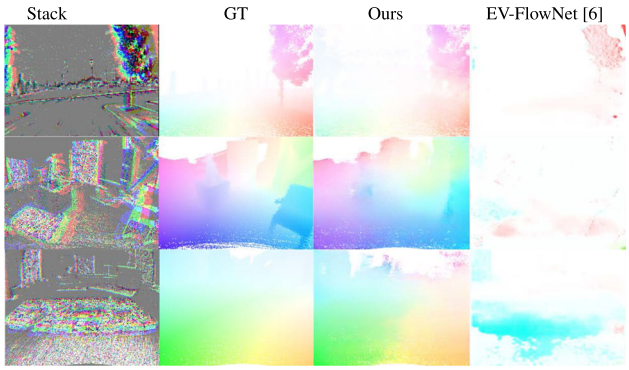

Fig. 19 Qualitative comparison between our method and EV-FlowNet (Zhu et al. 2018) in terms of optical flow estimation. We used a common color wheel for the comparison. We can see that our method can create dense relations of the optical flow, whereas EV-FlowNet creates sparse outputs

explained in Sect. 3.3, the mismatches between event stacks and their paired targets, namely, the intensity, depth, or flow data, change the predictions when such data are used in the training set. The event stack sometimes contains locations where events are available; however, there is no one-to-one relation between its paired target data. This makes the network learn to remove or ignore events in certain regions. These mismatches prevent our network from being reasonably fine-tuned to the real-world dataset after using it on the simulated datasets, and, therefore, training only on the real-world dataset had higher quality outputs.

Generally, by comparing the results of the outdoor scenes at day time to those at night time, we conclude that, to obtain a higher quality, the dataset used for the training should be very similar to the test dataset or at least should contain enough similar scenes and conditions to those contained in the latter dataset. Failure cases are mainly related to the mismatches between the triggered events and the intensity or depth sensor outputs in the dataset, which also affects the optical flow. The source of this can be lower dynamic range images and the accumulation of depth in moving objects sensed by the LiDAR depth sensor. Furthermore, on driving scenes, fast-moving cars can trigger many events as the rear tail, which saturates overlapping objects or blinds out the background scenes. The latter can be recovered partially using SBT. Far-away triggered events in distant buildings or clouds also create negative predictions if the training set does not include sufficient samples specifically for depth estimation.

Table 9 Quantitative comparison of the depth prediction

	outdoor_day				outdoor_night			
	(Zhu et al. 2019)	(Ye et al. 2018)	Ours	Ours M	(Zhu et al. 2019)	(Ye et al. 2018)	Ours	Ours M
Abs Rel ↓	0.36	0.33	0.29	0.30	0.37	0.39	0.34	0.36
RMSE log ↓	0.41	0.36	0.36	0.37	0.42	0.42	0.37	0.38
SIlog ↓	0.16	0.14	0.15	0.16	0.15	0.18	0.15	0.17
Acc: $\delta < 1.25$ ↑	0.46	0.97	0.68	0.80	0.45	0.95	0.59	0.67
Acc: $\delta < 1.25^2$ ↑	0.73	0.98	0.85	0.92	0.71	0.98	0.82	0.86
Acc: $\delta < 1.25^3$ ↑	0.88	0.99	0.93	0.96	0.86	0.99	0.94	0.96

We compared our method with E2FDE (Zhu et al. 2019) and ECN (Ye et al. 2018). "Ours M" represents the experiment results where the calculations were performed only on the triggered event locations. In comparison with the other methods, our method was able to generate comparable depth outputs

6 Conclusion

In this paper, we demonstrated how our cGAN-based approach could benefit from the properties of event cameras to accurately reconstruct HDR non-blurred intensity images and high-frame-rate videos from events. We first proposed two novel event stacking methods (*i.e*let@tokeneonedot, SBT and SBN) for both image and video reconstruction from events using the network. We then showed the advantages of using event cameras to generate high-dynamic-range images and high-frame-rate videos through experiments based on our datasets prepared from real-world sequences available online and using an event camera simulator. To show the robustness of our method, we compared our framework with other existing reconstruction methods. We showed that our method outperformed the other methods on the publicly available real-world dataset. We also confirmed that it is possible to generate high-dynamic-range images even under extreme illumination conditions and also non-blurred images under fast motion. Finally, we extended our proposed event stacking method and framework to estimate the dense depth and optical flow from events. The experimental results showed the versatility of our method for different tasks.

Acknowledgements This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (NRF 2018R1A2B3008640), the Next Generation Information Computing Development Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Science, ICT (NRF-2017M3C4A7069369) and Institute of Information & Communications Technology Planning & Evaluation (IITP) grant funded by Korea government (MSIT) (No.2020-0-00440, Development of Artificial Intelligence Technology that Continuously Improves Itself as the Situation Changes in the Real World).

References

Atapour-Abarghouei, A., & Breckon, T. P. (2018). Real-time monocular depth estimation using synthetic data with domain adaptation via image style transfer. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition,volume 18, page 1.

Bardow, P., Davison, A. J., & Leutenegger, S. (2016). Simultaneous optical flow and intensity estimation from an event camera. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 884–892.

Barua, S., Miyatani, Y., & Veeraraghavan, A. (2016). Direct face detection and video reconstruction from event cameras. In: *2016 IEEE winter conference on applications of computer vision (WACV)*, pp. 1–9. IEEE.

Benosman, R., Ieng, S. H., Clercq, C., Bartolozzi, C., & Srinivasan, M. (2012). Asynchronous frameless event-based optical flow. *Neural Networks*, *27*, 32–37.

Binas, J., Neil, D., Liu, S.-C., & Delbruck, T. (2017). DDD17: End-to-end davis driving dataset. arXiv preprint arXiv:1711.01458.

Chen, N. F. (2018). Pseudo-labels for supervised learning on dynamic vision sensor data, applied to object detection under ego-motion. In: *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pp. 644–653.

Community, B. O. (2018). Blender - a 3D modelling and rendering package. Stichting Blender Foundation, Amsterdam. Retrieved from http://www.blender.org

Cook, M., Gugelmann, L., Jug, F., Krautz, C., & Steger, A. (2011). Interacting maps for fast visual interpretation. In: *The 2011 international joint conference on neural networks (IJCNN)*, pp. 770–776. IEEE.

Gallego, G., Delbruck, T., Orchard, G., Bartolozzi, C., Taba, B., Censi, A., Leutenegger, S., Davison, A., Conradt, J., Daniilidis, K., Scaramuzza, D. (2019). Event-based Vision: A Survey. arXiv preprint arXiv:1904.08405.

Gallego, H., Rebecq, H., & Scaramuzza, D. (2018). A unifying contrast maximization framework for event cameras, with applications to motion, depth, and optical flow estimation. In: *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3867–3876).

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., & Bengio, Y. (2014). Generative adversarial nets. In: *Advances in neural information processing systems*, pp. 2672–2680.

Isola, P., Zhu, J.-Y., Zhou, T., & Efros, A. A. (2017). Image-to-image translation with conditional adversarial networks. arXiv preprint.

Karacan, L., Akata, Z., Erdem, A., & Erdem, E. (2016). Learning to generate images of outdoor scenes from attributes and semantic layouts. arXiv preprint arXiv:1612.00215.

Kim, H., Leutenegger, S., & Davison, A. J. (2016). Real-time 3d reconstruction and 6-dof tracking with an event camera. In: *European conference on computer vision*, pp. 349–364. Springer.

Kim, H., Handa, A., Benosman, R., Ieng, S.-H., & Davison, A. J. (2008). Simultaneous mosaicing and tracking with an event camera.J. *Solid State Circ*, *43*, 566–576.

Kingma, D., & Ba, J. (2015). Adam: A method for stochastic optimization. In the International Conference on Learning Representations (ICLR).

Lai, W. S., Huang, J. B., Wang, O., Shechtman, E., Yumer, E., & Yang, M. H. (2018). Learning blind video temporal consistency. In: *Proceedings of the European conference on computer vision (ECCV)*, pp. 170–185.

Ledig, C., Theis, L., Huszár, F., Caballero, J., Cunningham, A., Acosta, A., Aitken, A. P., Tejani, A., Totz, J., Wang, Z., et al. (2017). Photo-realistic single image super-resolution using a generative adversarial network. IEEE Conference on Computer Vision ;and Pattern Recognition (CVPR), volume 2, page 4.

Li, C., & Wand, M. (2016). Precomputed real-time texture synthesis with Markovian generative adversarial networks. In European Conference on Computer Vision, pp. 702–716. Springer.

Lichtsteiner, P., Posch, C., & Delbruck, T. (2008). A 128 × 128 120dB 15μs latency asynchronous temporal contrast vision sensor. *IEEE Journal of Solid-State Circuits*, *43*(2), 566–576.

Maqueda, A. I., Loquercio, A., Gallego, G., Garcıa, N., & Scaramuzza, D. (2018). Event-based vision meets deep learning on steering prediction for self-driving cars. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 5419–5427.

Mathieu, M., Couprie, C., & LeCun, Y. (2015). Deep multi-scale video prediction beyond mean square error. arXiv preprint arXiv:1511.05440.

Mittal, A., Moorthy, A. K., & Bovik, A. C. (2012). No-reference image quality assessment in the spatial domain. *IEEE Trans-actions on Image Processing*, *21*(12), 4695–4708.

Moeys, D. P., Corradi, F., Kerr, E., Vance, P., Das, G., Neil, D., Kerr, D., & Delbrück, T. (2016). Steering a predator robot using a mixed frame/event-driven convolutional neural network. In: *2016 Second international conference on event-based control, communication, and signal processing (EBCCSP)*, pp. 1–8. IEEE.

Moeys, D. P., Li, C., Martel, J. N., Bamford, S., Longinotti, L., Motsnyi, V., Bello, D. S. S., Delbruck, T. (2017). Color temporal contrast sensitivity in dynamic vision sensors. In: *IEEE international symposium on circuits and systems (ISCAS)*, 2017, pp. 1–4. IEEE.

Mueggler, E., Rebecq, H., Gallego, G., Delbruck, T., & Scaramuzza, D. (2017). The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and slam. *The International Journal of Robotics Research*, *36*(2), 142–149.

Munda, G., Reinbacher, C., & Pock, T. (2018). Real-time intensity-image reconstruction for event cameras using manifold regularisation. *International Journal of Computer Vision*, *126*(12), 1381–1393.

Nguyen, A., Do, T.-T., Caldwell, D. G., & Tsagarakis, N. G. Real-time 6dof pose relocalization for event cameras with stacked spatial lstm networks. arXiv preprint.

Open Source Computer Vision Library 2020.

Ouderaa, V. D., Tycho, F. A., & Worrall, D. E. (2019). Reversible gans for memory-efficient image-to-image translation. In: *IEEE conference on computer vision and pattern recognition (CVPR)*, pp. 4720–4728.

Rebecq, H., Gehrig, D., & Scaramuzza, D. (2018). Esim: An open event camera simulator. In: *Conference on robot learning*, pp. 969–982.

Rebecq, H., Ranftl, R., Koltun, V., & Scaramuzza, D. (2019). Events-to-video: Bringing modern computer vision to event cameras. In: *Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 3857–3866).

Rebecq, H., Gallego, G., Mueggler, E., & Scaramuzza, D. (2018). EMVS: Event-based multi-view stereo-3D reconstruction with an event camera in real-time. *International Journal of Computer Vision*, *126*(12), 1394–1414.

Rebecq, H., Horstschaefer, T., Gallego, G., & Scaramuzza, D. (2017). Evo: A geometric approach to event-based 6-dof parallel tracking and mapping in real time. *IEEE Robotics and Automation Letters*, *2*(2), 593–600.

Reinbacher, C., Graber, G., & Pock, T. (2016). Real-time intensity-image reconstruction for event cameras using manifold regularisation. arXiv preprint arXiv:1607.06283.

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In: *International conference on medical image computing and computer-assisted intervention*, pp. 234–241. Springer.

Ruder, M., Dosovitskiy, A., & Brox, T. (2016). Artistic style transfer for videos. In: *German conference on pattern recognition* (pp. 26-36). Springer, Cham.

rviz 3D visualization tool for ROS (2019). Retrieved from https://github.com/ros-visualization/rviz.

Scheerlinck, C., Barnes, N., & Mahony, R. (2018). Continuous-time intensity estimation using event cameras. arXiv preprint arXiv:1811.00386.

Scheerlinck, C., Rebecq, H., Stoffregen, T., Barnes, N., Mahony, R., & Scaramuzza, D. (2019). CED: Color event camera dataset. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops.

Shedligeri, P. A., Shah, K., Kumar, D., & Mitra, K. (2018). Photorealistic image reconstruction from hybrid intensity and event based sensor. arXiv preprint arXiv:1805.06140.

Simonyan, K., & Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition. In: *International conference on learning representations (ICLR)*.

Wang, X., & Gupta, A. (2016). Generative image modeling using style and structure adversarial networks. In: *European conference on computer vision*, pp. 318–335. Springer

Wang, Z., Chen, J., & CH, S. (2020). Hoi: Deep learning for image super-resolution: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*.

Wang, L., Mostafavi I, S. M., Ho, Y., & Yoon, K. (2019). Eventbased high dynamic range image and very high frame rate video generation using conditional generative adversarial networks. In: *IEEE conference on computer vision and pattern recognition (CVPR)*.

Wang, Z., Bovik, A. C., Sheikh, H. R., & Simoncelli, E. P. (2004). Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, *13*(4), 600–612.

Ye, C., Mitrokhin, A., Fermüller, C., Yorke, J. A., & Aloimonos, Y. (2018). Unsupervised learning of dense optical flow, depth and egomotion from sparse event data. arXiv preprint arXiv:1809.08625.

Yi, Z., Zhang, H. R., Tan, P., & Gong, M. (2017). Dualgan: Unsupervised dual learning for image-to-image translation. *ICCV*, 2868–2876.

Zhang, R., Isola, P., Efros, A. A., Shechtman, E., & Wang, O. (2018). The unreasonable effectiveness of deep features as a perceptual metric. In:*Proceedings of the IEEE conference on computer vision and pattern recognition* (pp. 586–595).

Zhang, L., Zhang, L., Mou, X., & Zhang, D. (2011). Fsim: A feature similarity index for image quality assessment. *IEEE Transactions on Image Processing*, *20*(8), 2378–2386.

Zhu, J.-Y., Park, T., Isola, P., & Efros, A. A. (2017). Unpaired image-to-image translation using cycle-consistent adversarial net-works. arXiv preprint.

Zhu, A. Z., Yuan, L., Chaney, K., & Daniilidis, K. (2018). Ev-flownet: Self-supervised optical flow estimation for event-based cameras. Proceedings of Robotics: Science and Systems

Zhu, A. Z., Yuan, L., Chaney, K., & Daniilidis, K. (2019). Unsupervised event-based learning of optical flow, depth, and egomotion. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 989-997).

Zhu, A. Z., Thakur, D., Ozaslan, T., Pfrommer, B., Kumar, V., & Daniilidis, K. (2018). The multi vehicle stereo event camera dataset: An event camera dataset for 3D perception. *IEEE Robotics and Automation Letters*, *3*(3), 2032–2039.