High Speed and High Dynamic Range Video with an Event Camera

Henri Rebecq, René Ranftl, Vladlen Koltun, and Davide Scaramuzza

Abstract—Event cameras are novel sensors that report brightness changes in the form of a stream of asynchronous "events" instead of intensity frames. They offer significant advantages with respect to conventional cameras: high temporal resolution, high dynamic range, and no motion blur. While the stream of events encodes in principle the complete visual signal, the reconstruction of an intensity image from a stream of events is an ill-posed problem in practice. Existing reconstruction approaches are based on hand-crafted priors and strong assumptions about the imaging process as well as the statistics of natural images. In this work we propose to learn to reconstruct intensity images from event streams directly from data instead of relying on any hand-crafted priors. We propose a novel recurrent network to reconstruct videos from a stream of events, and train it on a large amount of simulated event data. During training we propose to use a perceptual loss to encourage reconstructions to follow natural image statistics. We further extend our approach to synthesize color images from color event streams. Our quantitative experiments show that our network surpasses state-of-the-art reconstruction methods by a large margin in terms of image quality ($> 20\%$), while comfortably running in real-time. We show that the network is able to synthesize high framerate videos ($> 5,000$ frames per second) of high-speed phenomena (e.g., a bullet hitting an object) and is able to provide high dynamic range reconstructions in challenging lighting conditions. As an additional contribution, we demonstrate the effectiveness of our reconstructions as an intermediate representation for event data. We show that off-the-shelf computer vision algorithms can be applied to our reconstructions for tasks such as object classification and visual-inertial odometry and that this strategy consistently outperforms algorithms that were specifically designed for event data. We release the reconstruction code, a pre-trained model and the datasets to enable further research.

Index Terms—Event-based vision, dynamic vision sensor, video reconstruction, high speed, high dynamic range

Multimedia Material

Additional material (video, code and datasets) is available at http://rpg.ifi.uzh.ch/e2vid.

1 Introduction

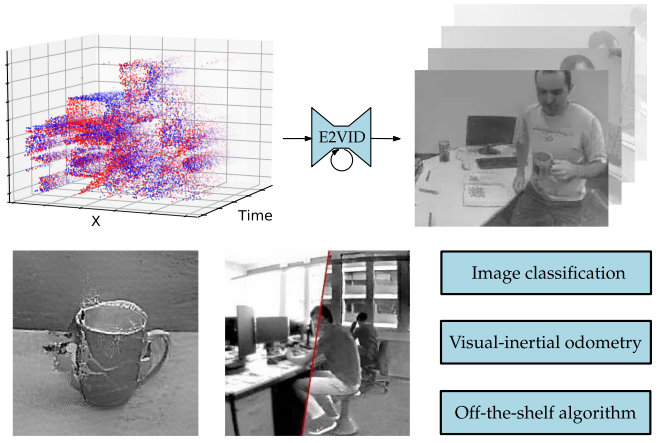
Event cameras are bio-inspired vision sensors that work radically differently from conventional cameras. Instead of capturing intensity images at a fixed rate, event cameras measure *changes* of intensity asynchronously at the time they occur. This results in a stream of *events*, which encode the time, location, and polarity (sign) of brightness changes (Fig. 2 - top). Event cameras such as the Dynamic Vision Sensor (DVS) [1] possess outstanding properties when compared to conventional cameras. They have a very high dynamic range (140 dB versus 60 dB), do not suffer from motion blur, and provide measurements with a microsecond temporal resolution. Event cameras thus provide a viable alternative (or complementary) sensor in conditions that are challenging for conventional cameras.

In theory, the stream of events contains the entire visual signal – in a highly compressed form – and could thus be decompressed to recover a video with arbitrarily high framerate and high dynamic range. However, real event cameras are noisy and differ significantly from the ideal camera model, which renders the reconstruction problem ill-posed. Naive integration of the event stream leads to very fast degradation of image quality due to accumulating noise. As a remedy, earlier works have proposed hand-crafted image priors to constrain the problem [2], [3], [4], [5]. However, these priors make strong assumptions about the statistics of natural images, leading to unrealistic reconstructions and artifacts. As a result, high-quality video reconstruction from event data has so far not been convincingly demonstrated.

In this work, we propose to bridge this gap by learning high-quality video reconstruction from sparse event data using a recurrent neural network (Fig. 1). In contrast to previous image reconstruction approaches [2], [4], [5], we do not embed handcrafted smoothness priors into our reconstruction framework. Instead, we learn video reconstruction from events using a large amount of simulated event data, and encourage the reconstructed images to have natural image statistics through a perceptual loss that operates on mid-level image features. Our network outperforms prior methods in terms of image quality by a large margin ($> 20\%$ improvement), demonstrating for the first time event-camera-based synthesized video sequences that are

- *H. Rebecq and D. Scaramuzza are with the Robotics and Perception Group, Department of Informatics and Department of Neuroinformatics, University of Zurich, Zurich, Switzerland and also with the Department of Neuroinformatics, ETH Zurich, Zurich, Switzerland. E-mail: {rebecq, sdavide}@ifi.uzh.ch.*
- *R. Ranftl and V. Koltun are with the Intelligent Systems Lab, Intel Labs, Santa Clara, CA 95054-1549 USA. E-mail: rene.ranftl@intel.com, vkoltun@gmail.com.*

Manuscript received 15 June 2019; revised 11 Oct. 2019; accepted 19 Dec. 2019. Date of publication 31 Dec. 2019; date of current version 5 May 2021. (Corresponding author: Henri Rebecq.)
Recommended for acceptance by B. Ommer.
Digital Object Identifier no. 10.1109/TPAMI.2019.2963386



(a) High framerate video (b) HDR video (c) Downstream applications

Fig. 1. Our network converts a spatio-temporal stream of events with microsecond temporal resolution (top left) into a high-quality video (top right). This enables synthesis of videos of high-speed phenomena such as a bullet piercing a mug (a), or scenes with high dynamic range (b). The reconstructions can also be used as input to off-the-shelf computer vision algorithms, thereby serving as an intermediate representation between event data and mainstream computer vision (c). The images in the figure were produced by the presented technique.

qualitatively on par with conventional cameras in terms of visual appearance. Our approach opens the door to a variety of applications, some of which we examine in this paper.

We explore the possibility of using an event camera to capture videos in scenarios that are challenging for conventional cameras. First, we show that our network can leverage the high temporal resolution of event data to synthesize high framerate ($> 5,000$ frames per second) videos of high-speed physical phenomena (Section 5.1). The resulting videos reveal details that are beyond the grasp of the naked eye or conventional cameras, which operate at a few hundred frames per second at best. Second, we show that our reconstructions preserve the high dynamic range of event cameras (Section 5.2), thus offering a viable alternative to conventional sensors in high dynamic range settings. We additionally present a simple strategy to synthesize color videos using a recent color event camera [6], without the need to retrain the network (Section 5.3).

Beyond pure imaging, we also consider using our approach for downstream applications. Since the output of an event camera is an asynchronous stream of events (a representation that is fundamentally different from natural images), existing computer vision techniques cannot be directly applied to this data. As a consequence, a number of algorithms have been specifically tailored to leverage event data, either processing the event stream in an event-by-event fashion [7], [8], [9], [10], [11], [12], or by building intermediate, "image-like" representations from event data [13], [14], [15], [15], [16]. In the spirit of the second category of methods, we explore the use of our image reconstructions as a novel representation for event data in Section 5.4. Specifically, we apply existing computer vision algorithms to images reconstructed from event data. We focus on object classification and visual-inertial odometry with event data, and show that this strategy consistently yields state of the art results in terms of accuracy. This suggests that high-quality reconstructions can be used as a bridge that brings the main stream of computer vision research to event cameras: mature algorithms, modern deep network architectures, and weights pretrained from large natural image datasets.

In summary, the contributions of this work are:

- A novel recurrent network that reconstructs video from a stream of events and outperforms the state of the art in terms of image quality by a large margin.
- We establish that networks trained from simulated event data generalize remarkably well to real events.
- Qualitative results showing that our method can be used in a variety of settings, such as high framerate video synthesis of high-speed phenomena (Section 5.1), reconstruction of high dynamic range video (Section 5.2), and reconstruction of color video (Section 5.3).
- Application of our method to two downstream problems: object classification and visual-inertial odometry from event data. Our method outperforms state-of-the-art algorithms designed specifically for event data in both applications.

2 Related Work

Because of its far reaching applications, events-to-video reconstruction is a popular topic in the event camera literature. The first evidence that it is possible to recover intensity information from event data was provided by [8] and [17], in the context of rotation estimation with an event camera. They showed how to reconstruct a single image from a large set of events collected by an event camera moving through a static scene and exploited the fact that every event provides one equation relating the intensity gradient and optic flow through brightness constancy [18]. Specifically, Cook et al. [8] used bio-inspired, interconnected networks to simultaneously recover intensity images, optic flow, and angular velocity from an event camera undergoing small rotations. Kim et al. [17] developed an Extended Kalman Filter to reconstruct a 2D panoramic gradient image (later upgraded to a full intensity frame by 2D Poisson integration) from a rotating event camera. They later extended their approach to static 3D scenes and 6 degrees-of-freedom (6DOF) camera motion [11]. Bardow *et al.* [2] proposed to estimate optic flow and intensity *simultaneously* from sliding windows of events through a variational energy minimization framework. They showed the first video reconstruction framework from events that is applicable to dynamic scenes. However, their energy minimization framework employs multiple hand-crafted regularizers, which can result in severe loss of detail in the reconstructions.

Recently, methods based on direct event integration have emerged. These approaches do not rely on any assumption about the scene structure or motion dynamics, and can naturally reconstruct videos at arbitrarily high framerates. Munda *et al.* [4] cast intensity reconstruction as an energy minimization problem defined on a manifold induced by the event timestamps. They combined direct event integration with total variation regularization and achieved real-time performance on the GPU. Scheerlinck et al. [5] proposed to filter the events with a high-pass filter prior to integration. They demonstrated video reconstruction results that are qualitatively comparable with [4] while being computationally more

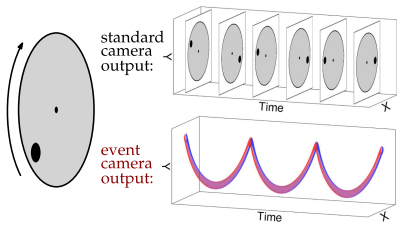


Fig. 2. Comparison of the output of a conventional camera and an event camera looking at a black disk on a rotating circle. While a conventional camera captures frames at a fixed rate, an event camera transmits the brightness changes continuously in the form of a spiral of events in space-time (red: positive events, blue: negative events). Figure inspired by [10].

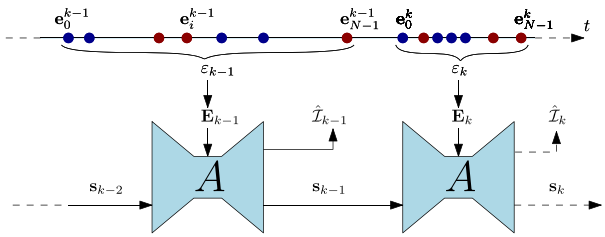


Fig. 3. Overview of our approach. The event stream (depicted as red/blue dots on the time axis) is split into windows ε_k containing multiple events. Each window is converted into a 3D event tensor $\mathbf{E}_k$ and passed through the network, together with the previous state $\mathbf{s}_{k-1}$ to generate a new image reconstruction $\hat{\mathcal{I}}_k$ and updated state $\mathbf{s}_k$. In this example, each window ε_k contains a fixed number of events $N = 7$.

efficient. While these approaches currently define the state-of-the-art, both suffer from artifacts which are inherent to direct event integration. The reconstructions suffer from "bleeding edges" caused by the fact that the contrast threshold (the minimum brightness change of a pixel to trigger an event) is neither constant nor uniform across the image plane. Additionally, pure integration of the events can in principle only recover intensity up to an unknown initial image $\mathcal{I}_0$ which causes "ghosting" effects where the trace of the initial image remains visible in the reconstructed sequence.

Barua *et al.* [3] proposed a learning-based approach to reconstruct intensity images from events. They used K-SVD [19] on simulated data to learn a dictionary that maps small patches of integrated events to an image gradient and used Poisson integration to recover the intensity image. In contrast, we do not reconstruct individual intensity images from small windows of events, but synthesize a temporally consistent video from a long stream of events (several seconds) using a recurrent network. Instead of mapping event patches to a dictionary of image gradients, we learn pixel-wise intensity estimation directly.

3 Video Reconstruction

An event camera consists of independent pixels that respond to changes in the spatio-temporal brightness signal $L(\mathbf{x}, t)$[1] and transmit the changes in the form of a stream of asynchronous events (Fig. 2). For an ideal sensor, an event $\mathbf{e}_i = (\mathbf{u}_i, t_i, p_i)$ is triggered at pixel $\mathbf{u}_i = (x_i, y_i)^T$ and time t_i when the brightness change since the last event at the pixel reaches a threshold $\pm C$. However, C is in reality neither constant nor uniform across the image plane. Rather, it strongly varies depending on factors such as the sign of the brightness change [12], the event rate (because of limited pixel bandwidth) [20], and the temperature [21]. Consequently, events cannot by directly integrated to recover accurate intensity images in practice.

3.1 Overview

Our goal is to translate a continuous stream of events into a sequence of images $\{\hat{\mathcal{I}}_k\}$, where $\hat{\mathcal{I}}_k \in [0,1]^{W \times H}$. To achieve this, we partition the incoming stream of events into sequential (non-overlapping) spatio-temporal windows $\varepsilon_k = \{\mathbf{e}_i\}$, for $i \in [0, N-1]$, each containing a fixed number N of events. The reconstruction function is implemented by a recurrent convolutional neural network, which maintains and updates an internal state $\mathbf{s}_k$ through time. For each new event sequence ε_k, we generate a new image $\hat{\mathcal{I}}_k$ using the network state $\mathbf{s}_{k-1}$ (see Fig. 3) and update the state $\mathbf{s}_k$. We train the network in supervised fashion, using a large amount of simulated event sequences with corresponding ground-truth images.

3.2 Event Representation

In order to be able to process the event stream using the convolutional recurrent network, we need to convert ε_k into a fixed-size tensor representation $\mathbf{E}_k$. A natural choice is to encode the events in a spatio-temporal voxel grid [22]. The duration $\Delta T = t_{N-1}^k - t_0^k$ spanned by the events in ε_k is discretized into B temporal bins. Every event distributes its polarity p_i to the two closest spatio-temporal voxels as follows:

$$\mathbf{E}(x_l, y_m, t_n) = \sum_{x_i = x_l y_i = y_m} p_i \max(0, 1 - |t_n - t_i^*|), \tag{1}$$

where $t_i^* \triangleq \frac{B-1}{\Delta T}(t_i - t_0)$ is the normalized event timestamp. We use $B = 5$ temporal bins. This value was chosen as the maximum value for which the training data fits in the GPU. We observed, however, that the value of B has limited influence on the reconstruction quality, even across different scenes and motions.

3.3 Training Data

Our network requires training data in the form of event sequences with corresponding ground-truth image sequences. However, there exists no large-scale dataset with event data and corresponding ground-truth images. Furthermore, images acquired by a conventional camera would provide poor ground truth in scenarios where event cameras excel, namely high dynamic range and high-speed scenes. For these reasons, we propose to train the network on synthetic event data, and show subsequently (in Section 4) that our network generalizes to real event data.

We use the event simulator ESIM [23], which allows simulating a large amount of event data reliably. ESIM renders images along the camera trajectory at high framerate, and interpolates the brightness signal at each pixel to approximate the continuous intensity signal needed to simulate an event camera. Consequently, ground-truth images $\mathcal{I}$ are readily available. We map MS-COCO images [24] to a 3D

1. Event cameras respond in fact to logarithmic brightness changes, i.e., $L = \log E$ where E is the *irradiance*.

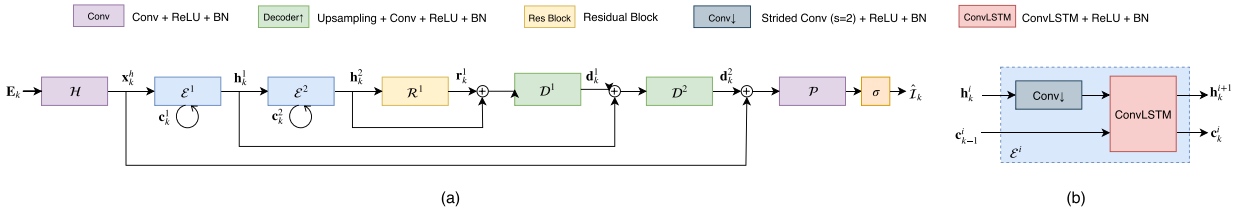


Fig. 4. We use a fully convolutional, UNet-like [26] architecture (a), composed of N_E recurrent encoder layers (b), followed by N_R residual blocks and N_E decoder layers, with skip connections between symmetric layers. Encoders are composed of a strided convolution (stride 2) followed by a ConvLSTM [27]. Decoder blocks perform bilinear upsampling followed by a convolution. ReLU activations and batch normalization [28] are used after each layer (except the last prediction layer, for which a sigmoid activation is used). In this diagram, $N_E = 2$ and $N_R = 1$.

plane and simulate the events triggered by random camera motion within this simple 3D scene. Using MS-COCO images allows capturing a much larger variety of scenes than is available in any existing event camera dataset. We set the camera sensor size to 240×180 pixels (to match the resolution of the DAVIS240C sensor used in our evaluation [25]). Note that inference can be performed at arbitrary resolutions since we will use a fully-convolutional network. Examples of generated synthetic event sequences are presented in the supplement.

We further enrich the training data by simulating a different set of positive and negative contrast thresholds for each simulated scene (sampled according to a normal distribution with mean 0.18 and standard deviation 0.03, values based on [17]). This data augmentation prevents the network from learning to naively integrate events, which would work well on noise-free, simulated data, but would generalize poorly to real event data (for which the assumption of a fixed contrast threshold does not hold).

We generate $1,000$ sequences of 2 seconds each, which results in approximately 35 minutes of simulated event data. Note that the simulated sequences contain only globally homographic motion (i.e. there is no independent motion in the simulated sequences). Nevertheless, our network generalizes surprisingly well to scenes with arbitrary motions, as will be shown in Sections 4 and 5.

3.4 Network Architecture

Our neural network is a recurrent, fully convolutional network that was inspired by the UNet [26] architecture. An overview is shown in Fig. 4. It is composed of a head layer ($\mathcal{H}$), followed by N_E recurrent encoder layers ($\mathcal{E}^i$), N_R residual blocks ($\mathcal{R}^j$), N_E decoder layers ($\mathcal{D}^l$), and a final image prediction layer ($\mathcal{P}$). Following [15], we use skip connections between symmetric encoder and decoder layers. The number of output channels is N_b for the head layer $\mathcal{H}$, and is doubled after each encoder layer (thus, the final encoder has $N_b \times 2^{N_E}$ output channels). The prediction layer performs a depthwise convolution (one output channel, kernel size 1), followed by a sigmoid layer to produce an image prediction. Encoder layers $\mathcal{E}^i$ (Fig. 4b) consist of a 2D downsampling convolution (kernel size: 5, stride: 2) followed by a ConvLSTM [27], with a kernel size of 3, and whose number of input and hidden layers is the same as the preceding downsampling convolution. Each encoder maintains a state $\mathbf{c}_k^i$ which is updated at every iteration, and initialized to zero at the first iteration ($k = 0$). The intermediate residual blocks [29] use a kernel size of 3. Each decoder layer consists of bilinear upsampling followed by a convolution with kernel size 5. Finally, we use the ReLU activation (for every layer except the final prediction) and batch normalization [28].

We used $N_E = 3$, $N_R = 2$, $N_b = 32$ and element-wise sum for the skip connection. In Section 6.1, we motivate these choices of hyperparameters as the result of a search over multiple network architectures. During training we unroll the network for L steps. We use $L = 40$.

Comparison with [30]. The proposed architecture differs from the preliminary version of this work [30] in several respects. First, unlike [30], we do not pass the last K reconstructed images as input to the network. In contrast, our network maintains an internal state (i.e. memory) – which it learns to update from an arbitrarily long sequence of past event tensors – instead of using only the last K images as memory. This also has the benefit of removing the parameter K. Second, instead of using a vanilla recurrent (RNN) architecture as [30], our network uses stacked ConvLSTM gates, which prevent vanishing gradients during backpropagation through time on long sequences. This leads to more stable behavior and increases the network's ability to deal with windows containing a variable number of events (Table 9). In addition, this enables training on significantly longer event sequences than [30] ($L = 40$ versus $L = 8$). In Section 6, we further study the benefits of this new architecture for temporal stability.

3.5 Loss

We use a combination of an image reconstruction loss and a temporal consistency loss.

Image Reconstruction Loss. The image reconstruction loss ensures that the reconstructed image is similar to the target image. While a direct pixel-wise loss such as the mean squared error (MSE) could be used, such losses are known to produce blurry images [31]. Instead, we use a perceptual loss (specifically, the calibrated perceptual loss LPIPS [32]). The perceptual loss passes the reconstructed image and the target image through a VGG network [33] that was trained on ImageNet [34], and averages the distances between VGG features across multiple layers. By minimizing LPIPS, our network effectively learns to endow the reconstructed images with natural statistics (i.e. with features close to those of natural images). Our reconstruction loss is computed as $\mathcal{L}_k^R = d(\hat{\mathcal{I}}_k, \mathcal{I}_k)$, where d denotes the LPIPS distance [32].

Temporal Consistency Loss. In our previous work [30], the network relied on the recurrent connection to naturally enforce temporal consistency between successive reconstructions. However, some temporal artifacts remained, notably some slight blinking that was especially noticeable in

homogeneous image regions. To address this issue, we introduce an explicit temporal consistency loss, the beneficial effect of which will be demonstrated in Section 6. Our temporal consistency loss is based on [35]. Given optical flow maps $\mathcal{F}_{k-1}^{k}$ between successive frames, the temporal loss is computed as the warping error between two successive reconstructions:

$$\mathcal{L}_k^{TC} = M_{k-1}^k \left\| \hat{\mathcal{I}}_k - \mathcal{W}_{k-1}^k(\hat{\mathcal{I}}_{k-1}) \right\|_1, \tag{2}$$

where $\mathcal{W}_{k-1}^k(\hat{\mathcal{I}}_{k-1})$ is the result of warping the reconstruction $\hat{\mathcal{I}}_{k-1}$ to $\hat{\mathcal{I}}_k$ using the optical flow $\mathcal{F}_{k-1}^k$, and $M_{k-1}^k = \exp(-\alpha \left\| \mathcal{I}_k - \mathcal{W}_{k-1}^k(\mathcal{I}_{k-1}) \right\|_2^2)$ is a weighting term that helps to mitigate the effect of occlusions (this term is small when the warping error in the ground truth images $\mathcal{I}_k$ is high, which happens predominantly at occlusions). We set $\alpha = 50$ in our experiments.

Note that the optical flow maps are only required at training time, but not at inference time. The final loss is a weighted sum of the reconstruction and temporal losses:

$$\mathcal{L} = \sum_{k=0}^{L} \mathcal{L}_k^R + \lambda_{TC} \sum_{k=L_0}^{L} \mathcal{L}_k^{TC}, \tag{3}$$

where $\lambda_{TC} = 5$ (this value was chosen empirically to balance the range of values taken by both losses), and $L_0 = 2$ (the first few samples of each sequence are ignored in the computation of the temporal loss to leave time for the reconstruction to converge).

3.6 Training Procedure

We split the synthetic sequences into 950 training sequences and 50 validation sequences. The input event tensors are normalized such that the mean and standard deviation of the nonzero values in each tensor is 0 and 1, respectively. We augment the training data using random 2D rotations (in the range of ± 20 degrees), horizontal and vertical flips, and random cropping (with a crop size of 128×128).

We implement our network using PyTorch [36] and use ADAM [37] with a learning rate of 0.0001. We use a batch size of 2 and train for 160 epochs ($320,000$ iterations).

3.7 Post-Processing

While the sigmoid activation guarantees that the resulting image prediction $\hat{\mathcal{I}}$ takes values between 0 and 1, we observe that the range of output values often does not span the entire range, i.e. the reconstructions can have low contrast. To remedy this, we rescale the image intensities using robust min/max normalization to get a final reconstruction $\hat{\mathcal{I}}^f$:

$$\hat{\mathcal{I}}^f = \frac{\hat{\mathcal{I}} - m}{M - m}, \tag{4}$$

where m and M are the 1 and 99 percent percentiles of $\hat{\mathcal{I}}$. Finally, $\hat{\mathcal{I}}^f$ is clipped to the range $[0, 1]$.

4 Evaluation

In this section, we present both quantitative and qualitative results on the fidelity of our reconstructions, and compare to recent methods [2], [4], [5]. We focus our evaluation on real event data. An evaluation on synthetic data can be found in supplementary material, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/TPAMI.2019.2963386.

We use event sequences from the Event Camera Dataset [38]. These sequences were recorded using a DAVIS240C sensor [25] moving in various environments. It contains events as well as ground-truth grayscale frames at a rate of 20 Hz. We remove redundant sequences (e.g. ones captured in the same scene) and those for which the frame quality is poor, leaving seven sequences in total that amount to $1,670$ ground-truth frames. For each sequence, we reconstruct a video from the events with our method and each baseline. For each ground-truth frame, we query the reconstructed image with the closest timestamp (tolerance of ± 1 ms).

Each reconstruction is then compared to the corresponding ground-truth frame according to several quality metrics. We apply local histogram equalization to every ground-truth frame and reconstructed frame prior to computing the error metrics (this way the intensity values lie in the same intensity range and are thus comparable). Note that the camera speed gradually increases in each sequence, leading to significant motion blur on the ground-truth frames towards the end of the sequences; we therefore exclude these fast sections in our quantitative evaluation. We also omit the first few seconds from each sequence, which leaves enough time for the baseline methods that are based on event integration to converge. Note that this works in favor of the baselines, as our method converges almost immediately (more details in Section 6).

We compare our approach against several state-of-the-art methods: [2] (which we denote as SOFIE for "Simultaneous Optic Flow and Intensity Estimation"), [5] (HF for "High-pass Filter"), and [4] (MR for "Manifold Regularization"), both in terms of image reconstruction quality and temporal consistency. For HF and MR, we used the code that was provided by the authors and manually tuned the parameters on the evaluated sequences to get the best results possible. For HF, we also applied a bilateral filter to the reconstructed images (with filter size $d = 5$ and $\sigma = 25$) in order to remove high-frequency noise, which improves the results of HF in all metrics. For SOFIE, we report qualitative results instead of quantitative results since we were not able to obtain satisfying reconstructions on our datasets using the code provided by the authors. We report three image quality metrics: mean squared error (MSE; lower is better), structural similarity (SSIM; higher is better) [39], and the calibrated perceptual loss (LPIPS; lower is better) [32]. In addition, we measure the temporal consistency of the reconstructed videos using the temporal loss introduced in Eq. (2). Note that computing the temporal loss requires optical flow maps between successive DAVIS frames, which we obtain with FlowNet2 [40].

Results and Discussion. The main quantitative results are presented in Table 1, and are supported by qualitative results in Figs. 5 and 6. Additional results are available in the supplementary material, available online. We also encourage the reader to watch the supplementary video, which conveys these results better than still images.

Our reconstruction method outperforms the state of the art by a large margin, with an average 24 percent increase in SSIM and a 22 percent decrease in LPIPS. Qualitatively, our method reconstructs small details remarkably well compared to the baselines (see the boxes in the first row of

TABLE 1
Comparison to State-of-the-Art Image Reconstruction Methods on the Event Camera Dataset [38]

Dataset	MSE			SSIM			LPIPS		
	HF	MR	Ours	HF	MR	Ours	HF	MR	Ours
dynamic_6dof	0.10	**0.05**	0.14	0.39	**0.52**	0.46	0.54	0.50	**0.46**
boxes_6dof	0.08	0.10	**0.04**	0.49	0.45	**0.62**	0.50	0.53	**0.38**
poster_6dof	0.07	**0.05**	0.06	0.49	0.54	**0.62**	0.45	0.52	**0.35**
shapes_6dof	0.09	0.19	**0.04**	0.50	0.51	**0.80**	0.61	0.64	**0.47**
office_zigzag	0.09	0.09	**0.03**	0.38	0.45	**0.54**	0.54	0.50	**0.41**
slider_depth	0.06	0.07	**0.05**	0.50	0.50	**0.58**	0.50	0.55	**0.44**
calibration	0.09	0.07	**0.02**	0.48	0.54	**0.70**	0.48	0.47	**0.36**
Mean	0.08	0.09	**0.05**	0.46	0.50	**0.62**	0.52	0.53	**0.41**

Our approach outperforms prior methods on almost all datasets and metrics by a large margin, with an average 24 percent increase in structural similarity (SSIM) and a 22 percent decrease in perceptual distance (LPIPS) compared to the best prior methods.

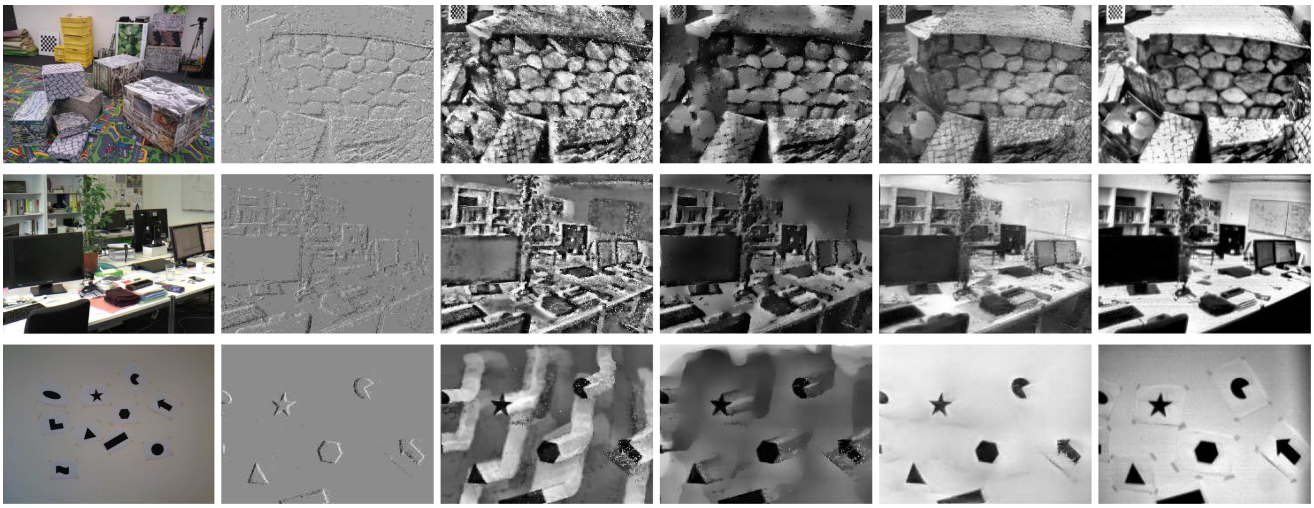


Fig. 5. Comparison of our method with MR and HF on sequences from [38]. Our network is able to reconstruct fine details well (textures in the first row), while avoiding common artifacts (e.g., the "bleeding edges" in the third row).

Fig. 5, for example). Furthermore, our method does not suffer from "ghosting" or "bleeding edges" artifacts that are present in other methods (particularly visible in the third row of Fig. 5). These artifacts result from (i) incorrectly estimated contrast thresholds and (ii) the fact that these methods can only estimate the image intensity up to some unknown initial intensity $\mathcal{I}_0$, the ghost of which can remain visible. We also compare our method to HF, MR, and SOFIE qualitatively using datasets and image reconstructions directly provided by the authors of [2], in Fig. 6. Once again, our network generates higher quality reconstructions, with finer details and less noise. In Section 5, we provide many more qualitative reconstruction results, and in particular show that our network is able to leverage the outstanding properties of events to reconstruct images in high-speed and high dynamic range scenarios.

Finally, Table 2 shows the temporal error (lower error means higher temporal consistency) for all methods, as well

Fig. 6. Qualitative comparison on the dataset introduced by [2]. Our method produces cleaner and more detailed results.

TABLE 2
Comparison of the Temporal Error (Eq. (2), Lower is Better) Between HF, MR, and Our Method

Dataset	Temporal Error			
	HF	MR	Ours	Ground truth
dynamic_6dof	3.32	1.91	**1.64**	0.53
boxes_6dof	3.37	1.79	**1.32**	0.59
poster_6dof	3.63	2.15	**1.77**	0.57
shapes_6dof	3.50	1.80	**1.48**	0.89
office_zigzag	3.18	1.58	**1.38**	0.48
slider_depth	2.14	1.62	**1.37**	0.70
calibration	2.72	1.52	**1.02**	0.62
Mean	3.12	1.77	**1.43**	0.63

Our video reconstructions have higher temporal consistency than the baselines.

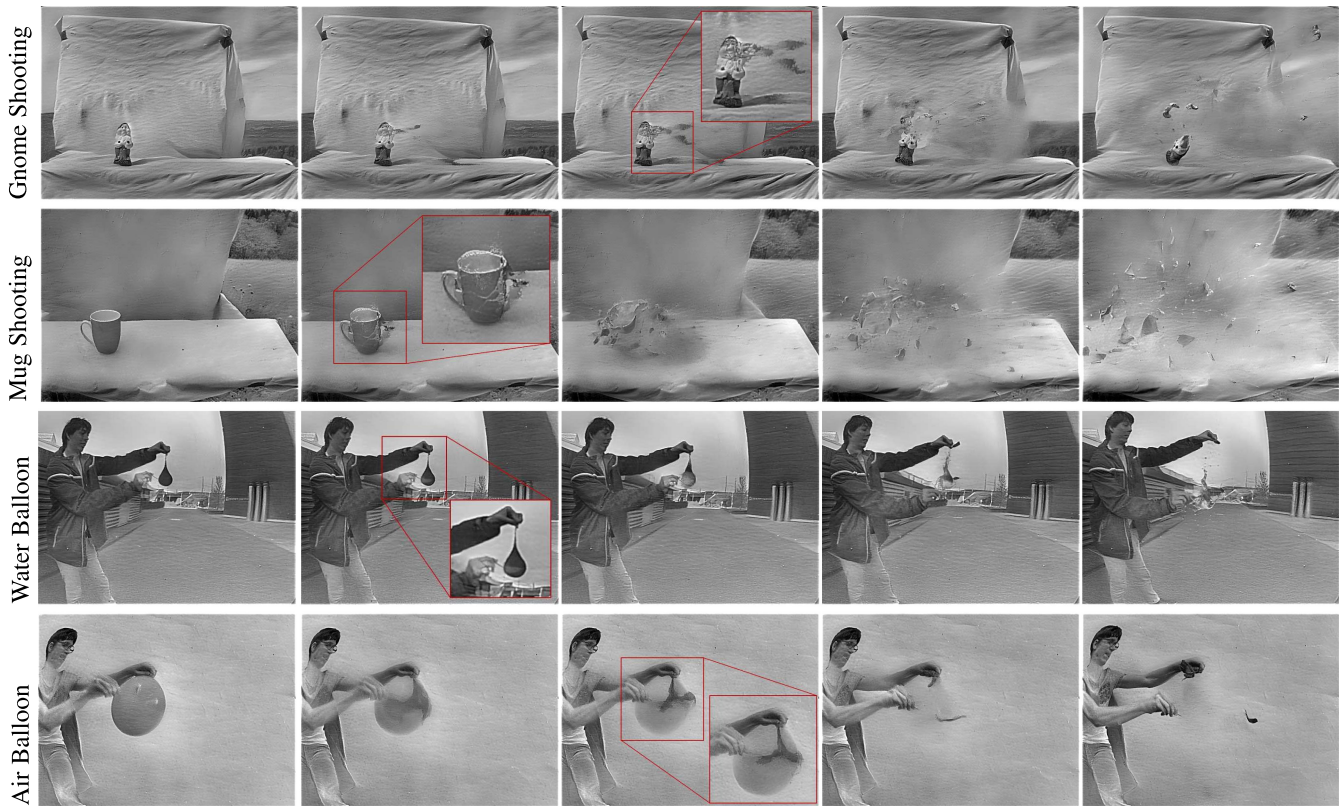


Fig. 7. Video reconstructions of high speed physical phenomena, synthesized at $> 5,000$ FPS with our approach. First two rows: shooting a garden gnome and a mug with a rifle. Last two rows: popping a water balloon and an air balloon with a needle. Our reconstructions reveal details invisible to the naked eye or a conventional consumer camera. In the first two rows, the trace of the bullet is clearly visible (the bullet itself was too fast for the event sensor to catch), and cracks in both objects are visible before the pieces fly apart. In the last two rows, the membrane of the balloons contracting away from the point where the needle hit is clearly visible.

as for the ground-truth sequences for reference. Note that the temporal loss is greater than zero on the ground-truth sequences because of small errors in optical flow estimation and occlusions. (It is still significantly lower than for all reconstruction methods.) Our method outperforms the competing approaches in terms of temporal consistency. We attribute this mostly to the temporal loss introduced in Section 3.5. In Section 6, we evaluate the effect of the temporal loss in an ablation study.

5 Applications

We now present some applications of our method. We synthesize high framerate videos of fast physical phenomena (Section 5.1), high dynamic range videos (Section 5.2), and color video (Section 5.3). Finally, we evaluate the effectiveness of our reconstructions as an intermediate representation that enables direct application of conventional vision algorithms to event data (Section 5.4).

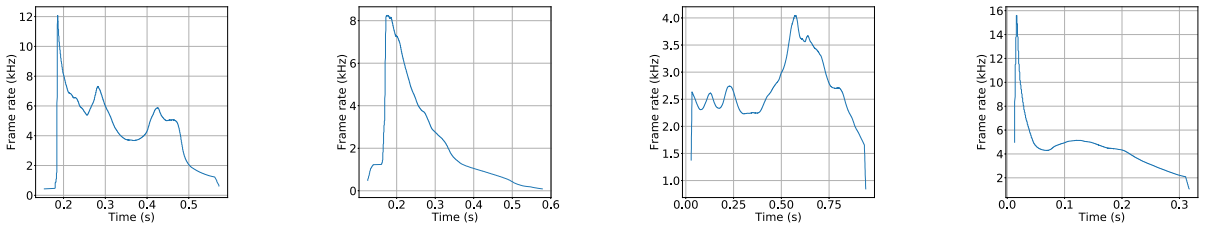
5.1 High Speed Video Reconstruction

Event cameras have a higher temporal resolution than conventional sensors (μs vs ms). In addition, the event stream is sparse by nature, which saves bandwidth. We now show that our method can decompress the event stream to reconstruct videos of fast motions with high framerate (thousands of frames per second).

Datasets. Since there exists no public event dataset containing fast physical phenomena, we recorded our own. We used the Samsung DVS Gen3 sensor [41], with VGA resolution. We recorded four sequences at daytime under bright sunlight (Fig. 7). The first two feature objects (plaster garden gnome, ceramic mug) being shot with a rifle (approximate muzzle velocity: 376 m/s). The last two sequences feature two balloons (filled with water and air, respectively) being popped with a needle. In order to reconstruct the background, each sequence starts with the camera being moved slightly, after which it is kept steady. We additionally recorded the first two sequences with a high-end mobile phone camera (Huawei P20 Pro) operating at 240 FPS[2].

High Framerate Video Synthesis From Events. We used a fixed number of events $N \simeq 10^4$ per window (exact values in Fig. 8), resulting in video reconstructions at only a few hundred FPS. While it would be possible to retrain the network with smaller window sizes to address the specific case of extremely high framerate video synthesis, it is in fact possible to arbitrarily increase the output framerate without retraining. To achieve that, we run multiple reconstructions in parallel, introducing a slight temporal shift (of D events) between each. We thus obtain a set of videos with different temporal offsets, then merged by reordering the frames from the individual videos. This yields a video with an arbitrarily high framerate, reaching multiple thousand FPS (Fig. 8). This temporal upsampling process may introduce

2. While the Huawei P20 Pro can in principle record at 960 FPS, it can only do so for a very short amount of time (0.2s), which made a synchronized recording impractical.



(a) Gnome $N = 10{,}000$, $D = 500$ (b) Mug $N = 12{,}000$, $D = 1000$ (c) Water balloon $N = 12{,}000$, $D = 1000$ (d) Air balloon $N = 20{,}000$, $D = 500$

Fig. 8. Reconstruction framerate for the high-speed sequences. The output framerate grows with the event rate and value of D, and varies between 1 kHz and 15 kHz.

some slight flickering, which is easily reduced using a simple filter [42].

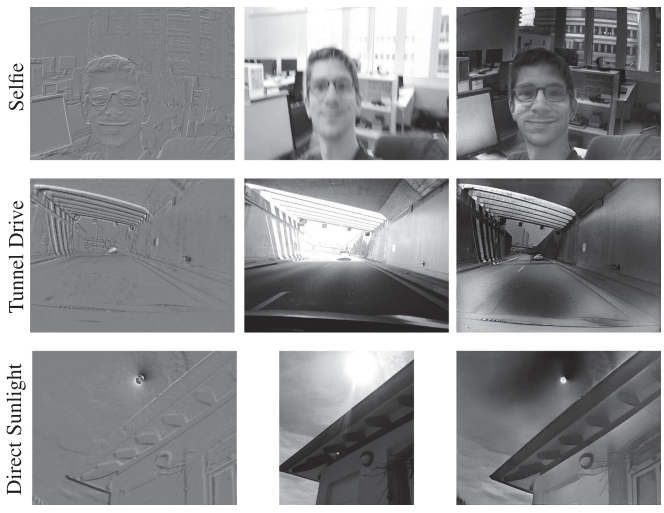
Results and Discussion. In the supplementary video, we show the synthesized, high framerate videos and compare them with the 240 FPS reference videos. Fig. 7 shows a few still frames from each sequence to convey the motion. At these extreme speeds, the event sensor is pushed to its limits: the events suffer from high noise and have many artefacts (e.g. readout artefacts). Nonetheless, our network performs well, revealing details that are invisible to the naked eye or a consumer camera. The output framerate (which varies with the event rate) is shown in Fig. 8, and consistently stays in the range of thousands of FPS: at least an order of magnitude above conventional consumer cameras. Note that the background is initially visible in the reconstructions, because we moved the sensor slightly at the start of each sequence to trigger events from the background. While our network tries to preserve the background throughout the sequence, it eventually decays away because of the prolonged absence of events from the background. We further study this decay effect in Fig. 16.

5.2 High Dynamic Range Reconstruction

Event cameras react to changes in log intensity [1], which endows them with much higher dynamic range than conventional cameras (140 dB versus 60 dB). The videos that our method synthesizes preserve the high dynamic range of the events. In Fig. 9, we support this claim by showing qualitative reconstruction results in a variety of challenging HDR scenes and compare our reconstructions to the corresponding frame from a conventional camera.

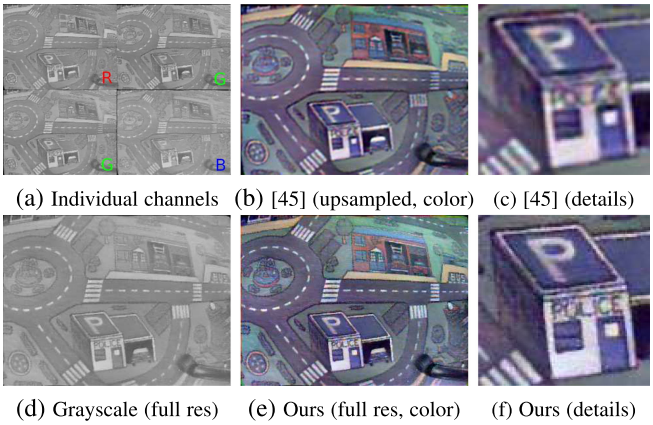
Datasets. There are many publicly available event camera datasets featuring HDR scenes [5], [38], [43]. However, the reference images in these datasets were taken by the DAVIS sensor [25], the quality of which falls short of state-of-the-art consumer cameras. To support a fair and up-to-date comparison, we recorded our own sequences in HDR scenes (indoors and outdoors), using a recent event camera (Samsung DVS Gen3) and a high-end smartphone for reference (Huawei P20 Pro). The two sensors were rigidly mounted to each other for recording, and the corresponding footage was geometrically and temporally aligned manually in post processing.

Results and Discussion. A comparison of our reconstruction results from event data and the frames from the conventional camera is presented in Fig. 9. The phone camera provides color images, which we converted to grayscale for easier visual comparison with our reconstructions. The first row of Fig. 9 shows a "selfie" sequence, recorded indoors with the sensors hand-held. While the window behind the main subject appears severely overexposed in the conventional frame (b), the events (a) capture the full dynamic range, which our network successfully leverages to reconstruct the entire scene. In addition, because of the camera shaking induced by the hand-held motion, the phone frame suffers from motion blur, which is not present in our reconstruction. The second row shows a driving sequence, recorded with both sensors placed on the windshield of a car driving out of a tunnel. Once again, the area outside of the tunnel is saturated in the conventional frame, while the events capture details both indoors and outdoors, which our reconstructions recover. Finally, the third row shows an outdoor example recorded with the sensors pointing directly at the sun on a bright day. In this extreme case, the events suffer from unusually high levels of noise (flickering events), which cause reconstruction artefacts such as the dark stain around the sun. Nonetheless, our reconstruction does not suffer from glare, and reveals the circular shape of the sun, which is lost in the frame. The video sequences, along with additional results on sequences from previously



(a) Events (b) Frame (c) Reconstruction

Fig. 9. Video reconstruction under challenging lighting. First row: Hand-held, indoor "selfie" sequence. Second row: Driving sequence recorded while driving out of a tunnel. Third row: Outdoor sequence recorded with the sensors pointing directly at the sun on a bright day. The frames from the consumer camera (Huawei P20 Pro) (b) suffer from under- or over-exposure, while the events (a) capture the whole dynamic range of the scene, which our method successfully recovers (c).



(a) Individual channels (b) [45] (upsampled, color) (c) [45] (details)

(d) Grayscale (full res) (e) Ours (full res, color) (f) Ours (details)

Fig. 10. Our color reconstruction approach. Color channels are reconstructed independently at quarter resolution (a), then upsampled and recombined into a low-quality color image (b,c). The latter is combined with a high-quality grayscale image (d) reconstructed using all the events (ignoring the CFA). The resulting color image (e,f) preserves fine details that are lost in the quarter resolution reconstruction [45] [compare (c) and (f)].

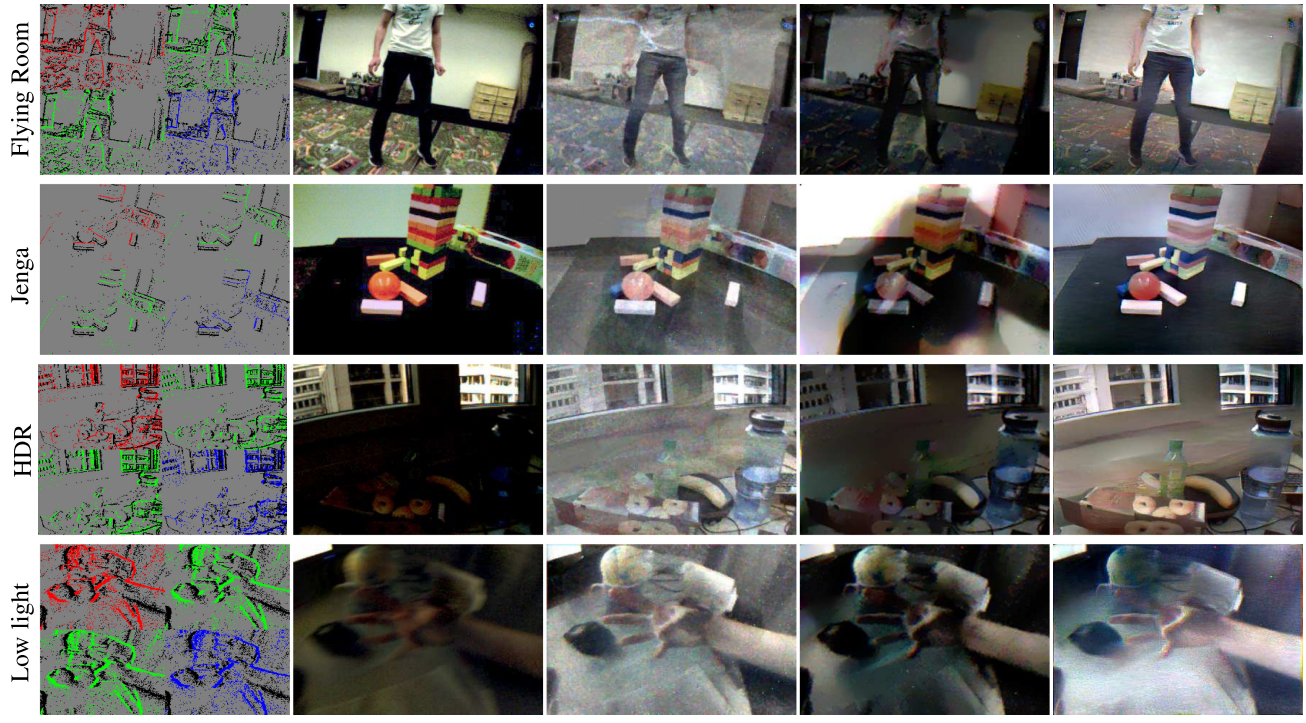
released datasets [5], [43], are available in the supplementary material, available online.

5.3 Color Video Reconstruction

Until recently, event cameras were mostly monochrome, producing events based on the variations in luminance [1], and discarding color information. This has changed with the recent introduction of a sensor that can perceive "color events", the Color-DAVIS346 [6]. The Color-DAVIS346 consists of an 8×6 mm CMOS chip equipped with a color array filter (CFA), forming an RGBG filter pattern. The pixels in the CFA are sensitive to the variation of their specific color filter, producing events that encode color information. Color reconstruction from such events was first shown by [44], where a single color image was recovered from a large set of events using a method similar to [17]. Later, [45] adapted existing monochrome video reconstruction methods [4], [30] to color by reconstructing the individual color channels independently, resulting in videos that have a quarter of the resolution of the sensor.

We now describe a simple method to perform color reconstruction from color event data at full resolution with our network, and then present qualitative results. Following [45], we reconstruct the four color channels independently at quarter resolution (Fig. 10a), upsample with bicubic interpolation, and recombine them into a low-quality color image (Fig. 10b). We then combine the latter with a full-resolution grayscale image obtained by running our network on all the events (ignoring the CFA). To do this, we project the (upsampled) color image in the LAB colorspace, and replace the luminance channel with the high-quality grayscale reconstruction (Fig. 10d). This exploits the human visual system's lower acuity to color differences than to luminance, a widely known phenomenon commonly used to compress videos (chroma subsampling [46]).

Fig. 11 shows color reconstruction results from our method, compared to HF and MR. Unlike MR and our method, HF preserves the Bayer pattern, thus color images can simply be obtained by applying a demosaicing algorithm to raw image reconstructions. For MR, we applied the same technique as our method to obtain color reconstructions. Qualitatively, the results are consistent with those obtained for grayscale reconstructions (Fig. 5): our method



(a) Color events (b) DAVIS frame (c) HF (d) MR (e) Our reconstruction

Fig. 11. Color reconstruction results from color events (a) using datasets from [45], comparing a conventional frame (b) with multiple reconstructions from events only: HF (c), MR (d), and ours (d). For visualization, the color events (a) are split into each color channel. Positive (ON) events are colored by the corresponding filter color, and negative (OFF) events are black.

produces cleaner reconstructions than HF (less noise and "bleeding edges" artifacts), and richer reconstructions than MR which tends to smooth out details. The last two rows of Fig. 11 show two scenarios with challenging lighting conditions: an HDR scene and a low-light scene.

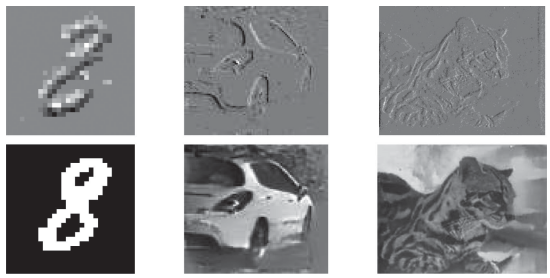
5.4 Downstream Applications

We now investigate the possibility of using our reconstructions as an intermediate representation that facilitates direct application of conventional computer vision algorithms to event data.

Representations for Event Data. Because the output of an event camera is an asynchronous stream of events (a representation that is fundamentally different from images), existing computer vision techniques cannot be directly applied to events. To address this problem, many algorithms have been specifically tailored to leverage event data for a wide range of applications (a good survey is provided by [47]). Despite strong differences between these methods, we argue that they follow the same paradigm, relying on two ingredients: (i) a mechanism to build an internal representation of past event data, and (ii) an inference mechanism to decode new events given the current internal representation. For example, [13], [14] tackle the task of object classification from event data as follows. As internal representation, they use a *time surface*: essentially an image recording the timestamp of the last event fired at each pixel (with some temporal decay to increase the influence of recent events), which is updated with every event. They train a supervised classifier (linear SVM) to predict an object class from this surface. Finally, object prediction (the inference mechanism) consists in evaluating the classifier, which can either be done with every new event (if fast enough), or at regular intervals [14].

The image reconstructions from our method can also be viewed as a representation for event data. Similarly to other representations, our network uses, at any given time, all past events to produce an image (in other words, to update the representation). Unlike other representations, however, our image reconstructions live in the space of natural images. As such, they are *transferrable*: any computer vision algorithm operating on regular images can be used as the inference mechanism, enabling the application of pre-existing vision algorithms to event data. We acknowledge, however, that reconstructing an image with our network is slow compared to other methods that use simpler and more efficient representations that may be tailored to a task (e.g., time surfaces for optic flow estimation [9]). Nevertheless, for the remainder of this section, we will show the effectiveness of our reconstructions as event representations on two different downstream tasks: object classification from events and camera pose estimation with events and inertial measurements. In both cases, we achieve state-of-the-art accuracy.

Object Classification. Pattern recognition from event data is an active research topic. While one line of work focuses on spiking neural architectures (SNNs) to recognize patterns from a stream of events with minimal latency (HFIRST [48]), conventional machine learning techniques combined with novel event representations such as time surfaces (HOTS [13]) have shown the most promising results so far. Recently, HATS [14] addressed the problem of object classification from a stream of events. They proposed several modifications to HOTS, and achieved major improvements in classification accuracy, outperforming all prior approaches by a large margin.



(a) N-MNIST (b) N-CARS (c) N-Caltech101

Fig. 12. Samples from each dataset used in the evaluation of our object classification approach based on events (Section 5.4). Top: Preview of the event sequence. Bottom: Our image reconstruction.

We propose an alternative approach to object classification based on a stream of events. Instead of using a handcrafted event representation, we directly apply a classification network (trained on image data) to images reconstructed from events. We compare our approach against several recent methods: HOTS, and the state-of-the-art HATS, using the datasets and metric (classification accuracy) used in the HATS paper. The N-MNIST (Neuromorphic-MNIST) and N-Caltech101 datasets [49] are event-based versions of the MNIST [50] and Caltech101 [51] datasets. To convert the images to event sequences, an event camera was placed on a motor, and automatically moved while pointing at images from MNIST (respectively Caltech101) that were projected onto a white wall. The N-CARS dataset [14] (recorded with an ATIS event camera [52]) proposes a binary classification task: deciding whether a car is visible or not using a 100ms sequence of events. Fig. 12 shows a sample event sequence from each of the three datasets.

Our approach follows the same methodology for each dataset. First, for each event sequence in the training set, we use our network to reconstruct an image from the events (Fig. 12, bottom row). We then train an off-the-shelf CNN for object classification using the reconstructed images from the training set. For N-MNIST, we use a simple CNN (details in supplementary material, available online) and train it from scratch. For N-Caltech101 and N-CARS, we use ResNet-18 [29], initialized with weights pretrained on ImageNet [34], and fine-tune the network for the dataset at hand. Once trained, we evaluate each network on the test set (images reconstructed from the events in the test set) and report the classification accuracy. Furthermore, we perform a transfer learning experiment for the N-MNIST and N-Caltech101 datasets (for which corresponding images are available for every event sequence): we train the CNN on the conventional image datasets, and evaluate the network directly on images reconstructed from events without fine-tuning.

For the baselines, we directly report the accuracy provided in [14]. To make the comparison with HATS as fair as possible, we also provide results of classifying HATS features with a ResNet-18 network (instead of the linear SVM that was used originally). The results are presented in

TABLE 3
Classification Accuracy Compared to Recent Approaches, Including HATS [14], the State-of-the-Art

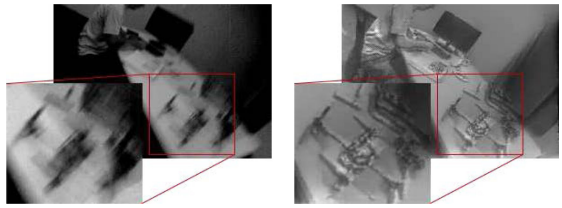
	N-MNIST	N-CARS	N-Caltech101
HOTS	0.808	0.624	0.210
HATS/linear SVM	**0.991**	0.902	0.642
HATS/ResNet-18	n.a.	0.904	0.700
Ours (transfer learning)	0.807	n.a.	0.821
Ours (fine-tuned)	0.983	**0.910**	**0.866**

Table 3, where the datasets are presented in increasing order of difficulty from left to right. Despite the simplicity of our approach, it outperforms all baselines. The gap between our method and the state-of-the-art increases with the difficulty of the datasets. While we perform slightly worse than HATS on N-MNIST (98.3 percent versus 99.1 percent), this can be attributed to the synthetic nature of N-MNIST, for which our approach does not bring substantial advantages compared to a hand-crafted feature representation such as HATS. Note that, in contrast to HATS, we did not perform any hyperparameter tuning. On N-CARS (binary classification task with natural event data), our method slightly outperforms the baseline (91 percent versus 90.4 percent for HATS).

On N-Caltech101 (the most challenging dataset, requiring classification of natural event data into 101 object classes), our method outperforms HATS by a large margin (86.6 percent versus 70.0 percent). This significant gap can be explained by the fact that our approach leverages decades of computer vision research and datasets. Lifting the event stream into the image domain with our events-to-video approach allows us to use a mature CNN architecture that was pretrained on existing labeled datasets. We can thus use powerful hierarchical features learned on a large body of image data – something that is not possible with event data, for which labeled datasets are scarce. Strikingly, our approach, in a pure transfer learning setting (i.e. feeding images reconstructed from events to a network trained on real image data) performs better than all other methods, while not using the event sequences from the training set. To the best of our knowledge, this is the first time that direct transfer learning between image data and event data has been achieved.

While the proposed approach reaches state-of-the-art accuracy, alternative approaches such as HATS are computationally more efficient, mostly because updating the internal representation (time surface) requires less operations than generating a new image with our neural network. Nonetheless, our approach is real-time capable. On N-Caltech101, end-to-end classification takes less than 10 ms (sequence reconstruction: ≤ 8 ms, object classification: ≤ 2 ms) on an NVIDIA RTX 2080 Ti GPU. More details can be found in Section 6.

Visual-Inertial Odometry. The task of visual-inertial odometry (VIO) is to recover the 6-degrees-of-freedom (6-DOF) pose of a camera from a set of visual measurements (images or events) and inertial measurements from an inertial measurement unit (IMU) that is rigidly attached to the camera. Because of its importance in augmented/virtual reality and mobile robotics, VIO has been extensively studied in the last decade and is relatively mature today [53], [54], [55], [56], [57]. Yet systems based on conventional cameras fail in challenging conditions such as high-speed motions and high-dynamic-range environments. This has recently motivated the development of VIO systems with event data (EVIO) [58], [59], [60].



(a) DAVIS frame (b) Our reconstruction

Fig. 13. Comparison of DAVIS frames and reconstructed frames on a high-speed portion of the 'dynamic_6dof' sequence. Our reconstructions from events do not suffer from motion blur, which leads to increased pose estimation accuracy (Table 4).

The state-of-the-art EVIO system, UltimateSLAM [60], operates by independently tracking visual features from pseudo-images reconstructed from events using motion compensation [59] (i.e. the internal representation) plus optional images from a conventional camera, and fusing the tracks with inertial measurements using an existing optimization backend [54].

Here, we go one step further and directly apply an off-the-shelf VIO system (specifically, VINS-Mono [57], which is state-of-the-art [61]) to videos reconstructed from events using either our approach, MR, or HF, and evaluate against UltimateSLAM. As is standard [58], [59], [60], we use sequences from the Event Camera Dataset [38], which contain events, frames, and IMU measurements from a DAVIS240C [25] sensor. Each sequence is 60 seconds long, and contains data from a hand-held event camera undergoing a variety of motions in several environments. All sequences feature extremely fast motions (angular velocity up to $880^\circ/s$ and linear velocity up to 3.5 m/s), which leads to severe motion blur on the frames (Fig. 13). We compare our approach against the two operating modes of UltimateSLAM: UltimateSLAM (E+I) which uses only events and IMU, and UltimateSLAM (E+F+I) that uses the events, the IMU, and additional frames. We run a publicly available VIO evaluation toolbox [62] on raw trajectories provided by the authors of UltimateSLAM, which ensures that the trajectories estimated by all methods are evaluated in the exact same manner. For completeness, we also report results from running VINS-Mono directly on the frames from the DAVIS sensor.

Table 4 presents the mean translation error of each method, for all datasets (additional results are presented in the supplement). First, we note that our method performs better than UltimateSLAM (E+I) on all sequences, with the exception of the 'shapes_6dof' sequence. This sequence features a few synthetic shapes with very few features (≤ 10), which cause VINS-Mono to not properly initialize, leading to high error (note that this is a problem with VINS-Mono and not our image reconstructions). Overall, the median error of our method is 0.15m, which is almost half the error of UltimateSLAM (E+I) (0.27m) which uses the exact same data. Indeed, while UltimateSLAM (E+I) uses coarse

TABLE 4
Mean Translation Error (in Meters) on the Sequences from [38]

Inputs	Ours E+I	U.SLAM E+I	U.SLAM E+F+I	HF E+I	MR E+I	VINS-Mono F+I
shapes_translation	0.18	0.32	**0.17**	failed	2.00	0.93
poster_translation	**0.05**	0.09	0.06	0.49	0.15	failed
boxes_translation	**0.15**	0.81	0.26	0.70	0.45	0.22
dynamic_translation	**0.08**	0.23	0.09	0.58	0.17	0.13
shapes_6dof	1.09	0.09	**0.06**	failed	3.00	1.99
poster_6dof	**0.12**	0.20	0.22	0.45	0.17	1.99
boxes_6dof	0.62	0.41	**0.34**	1.71	1.17	0.94
dynamic_6dof	0.15	0.27	**0.11**	failed	0.55	0.76
hdr_boxes	0.34	0.44	0.37	0.64	0.66	**0.32**
Mean	0.31	0.32	**0.19**	0.76	0.92	0.91
Median	**0.15**	0.27	0.17	0.61	0.55	0.84

Our method outperforms all other methods that use events and IMU, including UltimateSLAM(E+I). Surprisingly, it even performs on par with UltimateSLAM (E+F+I), while not using additional frames. Methods for which the mean translation error exceeds 5m *are marked as "failed".*

pseudo-images created from a single, small window of events, our network is able to reconstruct images with finer details and higher temporal consistency – both of which lead to better feature tracks and thus better pose estimates. Even more strikingly, our approach performs on par with UltimateSLAM (E+F+I), while the latter requires additional frames which we do not need. The median error of both methods is comparable (0.15m for ours versus 0.17 m for UltimateSLAM (E+F+I)).

Finally, we point out that running the same VIO (VINS-Mono) on competing image reconstructions (MR and HF) yields significantly larger tracking errors (e.g. median error three times larger for MR), which further highlights the superiority of our image reconstructions for downstream vision applications. We acknowledge that our approach is not as fast as UltimateSLAM. Since the main difference between both approaches is how they build the internal event representation, a rough estimate of the performance gap can be obtained by comparing the time it takes for each method to synthesize a new image. UltimateSLAM takes about 1 ms on a CPU whereas our method takes $\leq 4\text{ ms}$ on a high-end GPU. Nevertheless, our events-to-video network allows harnessing the outstanding properties of events for VIO and exceeds the accuracy of state-of-the-art EVIO algorithms that were designed specifically for event data.

6 Analysis

In this section, we provide a thorough analysis of our network. First, we measure the computational efficiency of our approach and compare it against several baselines. Second, we perform some ablation studies to justify the choice of some components. Finally, we analyze some of the interesting properties of our network and contrast these with prior work.

TABLE 5
Mean Inference Time Per Event Tensor (Measured on an NVIDIA RTX 2080 Ti GPU), and Maximum Frame Rate Achievable in Real Time, When Using Windows of Events of Fixed Duration

Resolution	Inference time (ms)	Max. frame rate (Hz)
128×128	2.71	369
240×180	5.4	185
346×260	10.5	95
640×480	30.9	32
1280×720	90.9	11

TABLE 6
Time it Takes to Process $N = 10,000$ Events for HF, MR, and Our Method (One 240×180 Event Tensor*)

	Frame synthesis time (ms) with $N = 10,000$ events
HF	0.7 / 1.5**
MR	0.9
Ours*	5.4

*HF works best when some filtering (e.g. a bilateral filter) is applied**, which increases the per-frame computation time.*

6.1 Computational Efficiency

Here we analyze the performance (i.e. computational efficiency) of the network. We also justify the specific hyperparameters of our architecture as the result of a simple search over these parameters.

We compare the performance of our method against HF [5] and MR [4]. Due to fundamental differences in the way each of these methods processes event data, it is difficult to provide a direct and fair performance comparison. HF and MR process the event stream in an event-by-event fashion, providing (in theory) a new image reconstruction with every incoming event. However, the raw image reconstructions from HF need to be filtered (for example, using a bilateral filter) to obtain results with reasonable quality. Our method, instead, operates on windows of events, which can be chosen to have a fixed number of events, or a fixed duration. In the former case, the reconstruction rate grows with the event rate. In the latter case, the reconstruction rate is constant (unlike HF and MR), and depends only on the sensor resolution.[3] In Table 5, we report the mean inference time per event tensor (for several common resolutions), and the corresponding maximum reconstruction rate achievable in real time, when operating on fixed-duration event windows. In addition, Table 6 reports the time it takes to process $N = 10,000$ events for MR and HF, compared against our inference time (i.e. on a tensor built from N events – this number was chosen because it yields good-quality images for all three methods). We ran our method and MR on an NVIDIA GeForce RTX 2080 Ti GPU, and HF on an Intel Core i9-9900K @ 3.60 GHz CPU.

Discussion. Our method is not as fast as HF or MR, which can process event data roughly 5 times faster. While high performance is not the main focus of the present work, our network can nonetheless easily run in real time (Table 5), providing state-of-the-art reconstructions in terms of video quality. We believe there is ample room for performance improvements. First, the performance of our approach may improve significantly when implemented on hardware specifically optimized to perform fast and efficient inference in neural networks. Second, exploiting the sparsity of the

3. Building an event tensor takes linear time with respect to the number of events. However, this time is negligible compared to the inference time.

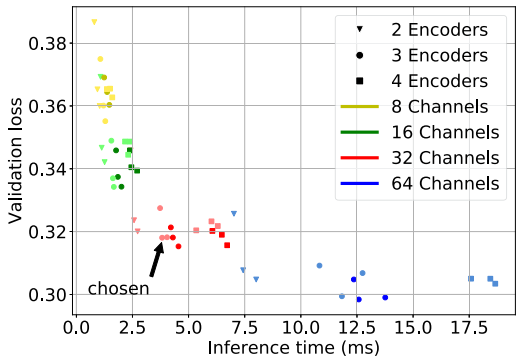


Fig. 14. Identifying a good trade-off between accuracy and efficiency. Each point in this graph corresponds to a different architecture variant. The color intensity indicates the type of skip connection: light colors correspond to element-wise sum and normal colors to concatenation. The distribution forms an "elbow" that suggests a good trade-off between inference time and reconstruction quality.

event tensors (most values of which are zeros) could additionally improve the computational efficiency by a large margin. One promising direction in that regard would be to use sparse convolutions [63] or hardware accelerators designed to efficiently process sparse inputs [64]. Finally, we believe one of the most alluring characteristics of our method is its ability to summarize a large number of events into one high-quality image. Since the network is robust to the number of events in each window (Section 6), it can be used with large windows of events when online operation is required (for example, for generating a live video preview), and run again offline with smaller windows to generate a high framerate video a posterori (as shown for example in Section 5.1). Alternatively, our network could also be used to generate high-quality images at low framerates (i.e. using large batches of events), which could be fused with event data using a complementary filter [5] to synthesize high-quality videos at very high framerate more efficiently.

Searching for a Lightweight Architecture. In this section, we show that our choice of network architecture parameters provides a good trade-off between quality and performance. We performed a simple architecture search over important hyperparameters:

- number of encoders N_E in the range $\{2, 3, 4\}$,
- number of residual blocks N_R (in $\{0, 1, 2\}$),
- type of skip connection $\oplus$ (concatenation or sum),
- number of feature channels N_b ($\{8, 16, 32, 64\}$).

We then trained a simplified version of the network (with the recurrent connection disabled) for each parameter combination (72 combinations in total) on a small subset of the full dataset to convergence (10 epochs), monitoring the validation loss as well as the mean inference time per event tensor (evaluated on the full network). The results are presented in Fig. 14. Notably, the distribution of the architectures form an "elbow", which indicates that there exists a good trade-off between model complexity (i.e. inference time) and reconstruction quality (validation loss). Based on these results, we choose $N_E = 3$, $N_R = 2$, $N_b = 32$, and element-wise sum for the skip connection.

TABLE 7
Ablation Study: Effect of the Temporal Loss

	MSE	SSIM	LPIPS	Temporal Error
w/o temporal loss	0.07	0.59	0.43	2.08
w/ temporal loss	**0.05**	**0.62**	**0.41**	**1.43**

The temporal loss improves the temporal consistency as well as the image quality.

TABLE 8
Ablation Study: Effect of the Recurrent Connection

	MSE	SSIM	LPIPS	Temporal Error
w/o recurrent	0.08	0.54	0.47	4.0
w/ recurrent	**0.05**	**0.62**	**0.41**	**1.43**

The quality and temporal consistency improve when using the recurrent connection, validating that our network is able to successfully propagate information through time.

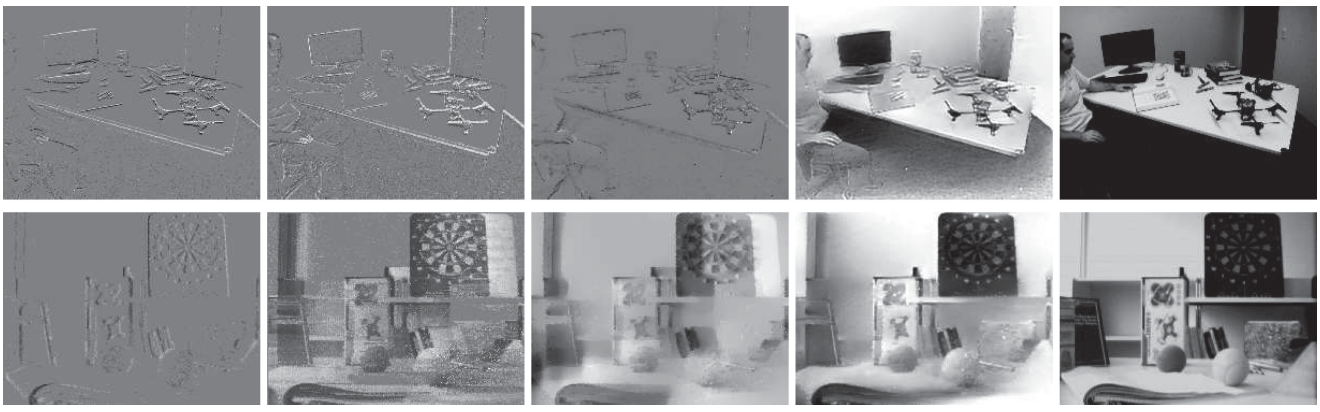
6.2 Ablation Studies

We now present some ablation studies to highlight the impact of some of the key features of our architecture.

Temporal Loss. To measure the impact of the temporal loss (Section 3.5, Eq. (2)), we trained the network without it (using $\lambda_{TC} = 0$). Table 7 compares the resulting network with the full network, in terms of temporal consistency and image reconstruction quality. We use the same sequences as in our quantitative evaluation (Section 4), and report the mean values over all datasets, for each metric. Unsurprisingly, the network that was trained with the temporal loss achieves better temporal consistency (31 percent decrease of the temporal error on average). More interestingly, the full network also performs better in terms of image quality overall (average improvement of about 5 percent), suggesting that the temporal loss also acts as a regularizer, driving the optimizer to converge to a better local optimum.

Recurrent Connection. Table 8 compares the image quality and temporal consistency when the recurrent connection of our network is removed. It highlights the role that the recurrent connection plays in achieving good video reconstruction quality. The recurrent connection increases temporal video consistency by a large margin (65 percent decrease in temporal error), removing the high-frequency blinking present in the reconstructions produced without the recurrent connection (see supplementary video). The recurrent connection also increases the image quality (15 percent), suggesting that the network can effectively leverage its short-term memory to reconstruct accurate images.

ConvLSTM versus Vanilla RNN. We now compare our network (based on stacked ConvLSTMs [27]) to the preliminary version of this work [30] based on a vanilla RNN, focusing on the generalization ability of both networks to the duration of the event windows used (or, equivalently, the number of events in each window). We train our network and [30] with the same training data, and evaluate both networks at two different inference rates, feeding non-overlapping event windows of a fixed duration τ. In the first experiment, we use $\tau = 50\text{ms}$, which is close to the average duration of each event tensor in the training data. In the second experiment, we use $\tau = 5\text{ms}$ to assess the generalization ability of both networks to a window size that is significantly different from the training data. Note that the second



(a) Events (b) HF (c) MR (d) Ours (e) Ground truth

Fig. 15. Analysis of the initialization phase (reconstruction from few events). This figure shows image reconstructions from each method, 0.5 seconds after the sensor was started. HF [5] and MR [4], which are based on event integration, cannot recover the intensity correctly, resulting in "edge" images (first and second row) or severe "ghosting" effects (third row, where the trace of the dartboard is clearly visible). In contrast, our network successfully reconstructs most of the scene accurately, even with a low number of events.

experiment is harder since the networks see a much smaller number of events at each time step (i.e. incomplete information), and thus must rely more strongly on their internal memory to recall the missing data necessary to produce good quality reconstructions. The results are reported in Table 9. When the window length is close to the training conditions ($\tau = 50$ ms), the networks perform similarly. However, with $\tau = 5$ ms, the image quality drops drastically for the vanilla RNN [30], while degrading only slightly with our network (6 percent decrease in SSIM). Our network is thus more robust to varying window sizes, maintaining intensity forward in time in a more stable fashion.

6.3 Edge Cases

We now present two interesting edge cases that shed light on some characteristics of our approach. We first analyze the initialization phase, when few events have been observed. We then perform a simple experiment to estimate the effective size of our network's memory, to better understand its behavior in regions with low event rates.

Initialization. By analyzing the initialization phase (i.e. when few events have been previously triggered) we gain interesting insight into how our network operates. We see significantly different behaviour when compared to prior approaches that are based on direct event integration. Fig. 15 compares image reconstructions from our approach, HF, and MR during the initialization phase. We specifically examine the interval from 0s to 0.5s from the beginning of capture. HF and MR, which rely on event integration, can only recover the intensity up to the initial (unknown) image $\mathcal{I}_0$ (i.e. they can only recover $\hat{\mathcal{I}} \approx \mathcal{I} - \mathcal{I}_0$), which results in an "edge" image which does not capture the appearance of the scene correctly. In contrast, our method successfully leverages deep priors to reconstruct the scene despite the low number of events.

TABLE 9
Reconstruction Quality With $\tau = 50$ ms versus $\tau = 5$ ms Windows

	MSE	SSIM	LPIPS
RNN ($\tau = 50$ ms)	0.05	0.61	**0.40**
RNN ($\tau = 5$ ms)	0.14	0.27	0.71
Ours ($\tau = 50$ ms)	**0.05**	**0.62**	0.41
Ours ($\tau = 5$ ms)	0.06	0.58	0.44

The RNN [30] does not generalize to $\tau = 5$ ms, yielding poor quality reconstructions (high MSE and LPIPS, low SSIM). In contrast, the reconstruction quality of our network degrades only slightly ($\leq$6 percent decrease in SSIM), thanks to its more stable recurrent connection.

Network Memory. For how long can our network remember intensity information? To answer this question – in other words, to measure the effective size of the temporal receptive field of our network – we perform a simple experiment. We take a given sequence of events and artificially stop the events at a fixed time t (this is achieved in practice by zeroing out all the event tensors with timestamps $\geq t$). We feed the resulting empty event tensors to our network and present the evolution of the reconstructions as a function of the number of iterations in Fig. 16. In the complete absence of events, the current reconstruction should be left untouched, i.e. the network should simply copy all the pixel values forward. As shown in Fig. 16, this is in fact what the network has learned to do in the first few iterations, although this was never hardcoded in the network design; rather, the recurrent network discovered this pattern from the training data. However, as can be observed in Fig. 16, the network gradually decays the image intensity when forced to perform inference with tensors containing no events. Interestingly, the image decay is not isotropic: the regions with high contrast (e.g. the dart board in the first row) tend to be preserved for a higher number of iterations. While this experiment may seem artificial – such a situation cannot happen in practice since no event tensor is generated when no events fire – it sheds light on the behavior of the network in regions where very few events are fired. We believe the length of the memory is strongly tied to the distribution of optical flows in the training data. Indeed, our synthetic training datasets (aimed at general-purpose reconstruction) contain few "pauses" (i.e. regions with low event rate), thus our network does not need to retain information for long time periods. This suggests that our network could be further improved for specific applications by generating training data with a similar distribution of optical flow than the target application. In autonomous driving for example, the network may learn to retain intensity information for a long

(a) Initial time (b) After 1 iteration (c) After 3 iterations (d) After 10 iterations (e) After 20 iterations

Fig. 16. In this experiment, the events are artificially stopped at some time t, i.e. the network is fed empty event tensors in all subsequent iterations. The correct thing to do would be to simply copy the first image to all subsequent predictions. The network has instead learned to gradually decay the image.

time in the center of the image (focus of expansion, low event rate), and for a shorter amount of time on the sides of the image (higher event rate).

7 Conclusion

We presented a novel events-to-video reconstruction framework based on a recurrent convolutional network trained on simulated event data. In addition to outperforming state-of-the-art reconstruction methods on real event data by a large margin ($> 20\%$ improvement), we showed the applicability of our method to synthesize high framerate, high dynamic range, and color video reconstructions from event data only. Finally, we demonstrated the effectiveness of our reconstructions as as intermediate representation that bridges event cameras and mainstream computer vision.

Acknowledgments

The authors would like to thank Cedric Scheerlinck, Matthias Faessler and his family for valuable discussions and for help recording data, as well as iniVation and Samsung for providing the event cameras that were used for this research. This work was supported by the Swiss National Center of Competence Research Robotics (NCCR), the SNSF-ERC Starting Grant and Qualcomm (through the Qualcomm Innovation Fellowship 2018).

References

[1] P. Lichtsteiner, C. Posch, and T. Delbruck, "A 128 $\times$ 128 120 dB 15 μs latency asynchronous temporal contrast vision sensor," *IEEE J. Solid-State Circuits*, vol. 43, no. 2, pp. 566–576, Feb. 2008.

[2] P. Bardow, A. J. Davison, and S. Leutenegger, "Simultaneous optical flow and intensity estimation from an event camera," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 884–892.

[3] S. Barua, Y. Miyatani, and A. Veeraraghavan, "Direct face detection and video reconstruction from event cameras," in *Proc. IEEE Winter Conf. Appl. Comput. Vis.*, 2016, pp. 1–9.

[4] G. Munda, C. Reinbacher, and T. Pock, "Real-time intensity-image reconstruction for event cameras using manifold regularisation," *Int. J. Comput. Vis.*, vol. 126, no. 12, pp. 1381–1393, Jul. 2018.

[5] C. Scheerlinck, N. Barnes, and R. Mahony, "Continuous-time intensity estimation using event cameras," in *Proc. Asian Conf. Comput. Vis.*, 2018, pp. 308–324.

[6] G. Taverni *et al.*, "Front and back illuminated dynamic and active pixel vision sensors comparison," *IEEE Trans. Circuits Syst. II*, vol. 65, no. 5, pp. 677–681, May 2018.

[7] J. Conradt, M. Cook, R. Berner, P. Lichtsteiner, R. J. Douglas, and T. Delbruck, "A pencil balancing robot using a pair of AER dynamic vision sensors," in *Proc. IEEE Int. Symp. Circuits Syst.*, 2009, pp. 781–784.

[8] M. Cook, L. Gugelmann, F. Jug, C. Krautz, and A. Steger, "Interacting maps for fast visual interpretation," in *Proc. Int. Joint Conf. Neural Netw.*, 2011, pp. 770–776.

[9] R. Benosman, C. Clercq, X. Lagorce, S.-H. Ieng, and C. Bartolozzi, "Event-based visual flow," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 2, pp. 407–417, Feb. 2014.

[10] E. Mueggler, B. Huber, and D. Scaramuzza, "Event-based, 6-DOF pose tracking for high-speed maneuvers," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 2014, pp. 2761–2768.

[11] H. Kim, S. Leutenegger, and A. J. Davison, "Real-time 3D reconstruction and 6-DoF tracking with an event camera," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 349–364.

[12] G. Gallego, J. E. A. Lund, E. Mueggler, H. Rebecq, T. Delbruck, and D. Scaramuzza, "Event-based, 6-DOF camera tracking from photometric depth maps," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 10, pp. 2402–2412, Oct. 2018.

[13] X. Lagorce, G. Orchard, F. Gallupi, B. E. Shi, and R. Benosman, "HOTS: A hierarchy of event-based time-surfaces for pattern recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 7, pp. 1346–1359, Jul. 2017.

[14] A. Sironi, M. Brambilla, N. Bourdis, X. Lagorce, and R. Benosman, "HATS: Histograms of averaged time surfaces for robust event-based object classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 1731–1740.

[15] A. Z. Zhu, L. Yuan, K. Chaney, and K. Daniilidis, "EV-FlowNet: Self-supervised optical flow estimation for event-based cameras," in *Proc. Conf. Robotics: Sci. Syst.*, 2018.

[16] Y. Zhou, G. Gallego, H. Rebecq, L. Kneip, H. Li, and D. Scaramuzza, "Semi-dense 3D reconstruction with a stereo event camera," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 242–258.

[17] H. Kim, A. Handa, R. Benosman, S.-H. Ieng, and A. J. Davison, "Simultaneous mosaicing and tracking with an event camera," in *Proc. British Mach. Vis. Conf.*, 2014.

[18] D. Gehrig, H. Rebecq, G. Gallego, and D. Scaramuzza, "Asynchronous, photometric feature tracking using events and frames," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 766–781.

[19] M. Aharon, M. Elad, and A. M. Bruckstein, "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," *IEEE Trans. Signal Process.*, vol. 54, no. 11, pp. 4311–4322, Nov. 2006.

[20] C. Brandli, L. Muller, and T. Delbruck, "Real-time, high-speed video decompression using a frame- and event-based DAVIS sensor," in *Proc. IEEE Int. Symp. Circuits Syst.*, 2014, pp. 686–689.

[21] J. Xu, J. Zou, and Z. Gao, "Comment on temperature and parasitic photocurrent effects in dynamic vision sensors," *IEEE Trans. Electron Devices*, vol. 65, no. 7, pp. 3081–3082, Jul. 2018.

[22] A. Z. Zhu, L. Yuan, K. Chaney, and K. Daniilidis, "Unsupervised event-based optical flow using motion compensation," in *Proc. Eur. Conf. Comput. Vis. Workshops*, 2018, pp. 711–714.

[23] H. Rebecq, D. Gehrig, and D. Scaramuzza, "ESIM: An open event camera simulator," in *Proc. Conf. Robot. Learn.*, 2018, pp. 969–982.

[24] T. Lin *et al.*, "Microsoft COCO: Common objects in context," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 740–755.
[25] C. Brandli, R. Berner, M. Yang, S.-C. Liu, and T. Delbruck, "A 240 × 180 130 dB 3 us latency global shutter spatiotemporal vision sensor," *IEEE J. Solid-State Circuits*, vol. 49, no. 10, pp. 2333–2341, 2014.
[26] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Proc. Int. Conf. Medical Image Comput. Comput.-Assisted Intervention*, 2015, pp. 234–241.
[27] X. Shi, Z. Chen, H. Wang, D. Yeung, W. Wong, and W. Woo, "Convolutional LSTM network: A machine learning approach for precipitation nowcasting," in *Proc. Conf. Neural Inf. Process. Syst.*, 2015, pp. 802–810.
[28] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learn.*, 2015, pp. 448–456.
[29] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.
[30] H. Rebecq, R. Ranftl, V. Koltun, and D. Scaramuzza, "Events-to-video: Bringing modern computer vision to event cameras," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2019.
[31] J. Johnson, A. Alahi, and F. Li, "Perceptual losses for real-time style transfer and super-resolution," in *Proc. Eur. Conf. Comput. Vis.*, 2016, pp. 694–711.
[32] R. Zhang, P. Isola, A. A. Efros, E. Shechtman, and O. Wang, "The unreasonable effectiveness of deep features as a perceptual metric," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 586–595.
[33] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Representations*, 2015.
[34] O. Russakovsky *et al.*, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, Apr. 2015.
[35] W. Lai, J. Huang, O. Wang, E. Shechtman, E. Yumer, and M. Yang, "Learning blind video temporal consistency," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 179–195.
[36] A. Paszke *et al.*, "Automatic differentiation in PyTorch," in *Proc. 31st Conf. Neural Inf. Process. Syst. Workshops*, 2017.
[37] D. P. Kingma and J. L. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Representations*, 2015.
[38] E. Mueggler, H. Rebecq, G. Gallego, T. Delbruck, and D. Scaramuzza, "The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and SLAM," *Int. J. Robot. Res.*, vol. 36, pp. 142–149, 2017.
[39] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: From error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, Apr. 2004.
[40] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, "FlowNet 2.0: Evolution of optical flow estimation with deep networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 1647–1655.
[41] B. Son *et al.*, "A 640x480 dynamic vision sensor with a 9 um pixel and 300Meps address-event representation," in *Proc. IEEE Int. Solid-State Circuits Conf.*, 2017, pp. 66–67.
[42] "ReduceFlicker," 2016. [Online]. Available: http://avisynth.nl/index.php/ReduceFlicker
[43] A. Z. Zhu, D. Thakur, T. Ozaslan, B. Pfrommer, V. Kumar, and K. Daniilidis, "The multivehicle stereo event camera dataset: An event camera dataset for 3D perception," *IEEE Robot. Autom. Lett.*, vol. 3, no. 3, pp. 2032–2039, Jul. 2018.
[44] D. P. Moeys *et al.*, "Color temporal contrast sensitivity in dynamic vision sensors," in *Proc. IEEE Int. Symp. Circuits Syst.*, 2017, pp. 1–4.
[45] C. Scheerlinck, H. Rebecq, T. Stoffregen, N. Barnes, R. Mahony, and D. Scaramuzza, "CED: Color event camera dataset," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, 2019.
[46] C. Poynton, "Chroma subsampling notation," 2002. [Online]. Available: http://vektor.theorem.ca/graphics/ycbcr/Chroma_subsampling_notation.pdf
[47] G. Gallego *et al.*, "Event-based vision: A survey," vol. abs/1904.08405, 2019. [Online]. Available: http://arxiv.org/abs/1904.08405
[48] G. Orchard, C. Meyer, R. Etienne-Cummings, C. Posch, N. Thakor, and R. Benosman, "HFirst: A temporal approach to object recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 10, pp. 2028–2040, Oct. 2015.
[49] G. Orchard, A. Jayawant, G. K. Cohen, and N. Thakor, "Converting static image datasets to spiking neuromorphic datasets using saccades," *Front. Neurosci.*, vol. 9, 2015, Art. no. 437.
[50] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
[51] L. Fei-Fei, R. Fergus, and P. Perona, "One-shot learning of object categories," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 4, pp. 594–611, Apr. 2006.
[52] C. Posch, D. Matolin, and R. Wohlgenannt, "A QVGA 143 dB dynamic range frame-free PWM image sensor with lossless pixel-level video compression and time-domain CDS," *IEEE J. Solid-State Circuits*, vol. 46, no. 1, pp. 259–275, Jan. 2011.
[53] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint Kalman filter for vision-aided inertial navigation," in *Proc. IEEE Int. Conf. Robot. Autom.*, Apr. 2007, pp. 3565–3572.
[54] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visual-inertial SLAM using nonlinear optimization," *Int. J. Robot. Res.*, vol. 34, no. 3, pp. 314–334, 2015, doi: 10.1177/0278364914554813.
[55] M. Bloesch, S. Omari, M. Hutter, and R. Siegwart, "Robust visual inertial odometry using a direct EKF-based approach," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 2015, pp. 298–304.
[56] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-manifold preintegration for real-time visual-inertial odometry," *IEEE Trans. Robot.*, vol. 33, no. 1, pp. 1–21, Feb. 2017.
[57] T. Qin, P. Li, and S. Shen, "VINS-Mono: A robust and versatile monocular visual-inertial state estimator," *IEEE Trans. Robot.*, vol. 34, no. 4, pp. 1004–1020, Aug. 2018.
[58] A. Z. Zhu, N. Atanasov, and K. Daniilidis, "Event-based visual inertial odometry," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 5816–5824.
[59] H. Rebecq, T. Horstschaefer, and D. Scaramuzza, "Real-time visual-inertial odometry for event cameras using keyframe-based nonlinear optimization," in *Proc. British Mach. Vis. Conf.*, 2017, pp. 16.1–16.12.
[60] A. Rosinol Vidal, H. Rebecq, T. Horstschaefer, and D. Scaramuzza, "Ultimate SLAM? combining events, images, and IMU for robust visual SLAM in HDR and high speed scenarios," *IEEE Robot. Autom. Lett.*, vol. 3, no. 2, pp. 994–1001, Apr. 2018.
[61] J. Delmerico and D. Scaramuzza, "A benchmark comparison of monocular visual-inertial odometry algorithms for flying robots," in *Proc. IEEE Int. Conf. Robot. Autom.*, 2018, pp. 2502–2509.
[62] Z. Zhang and D. Scaramuzza, "A tutorial on quantitative trajectory evaluation for visual(-inertial) odometry," in *Proc. IEEE/RSJ Int. Conf. Intell. Robot. Syst.*, 2018, pp. 7244–7251.
[63] B. Graham, M. Engelcke, and L. van der Maaten, "3D semantic segmentation with submanifold sparse convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 9224–9232.
[64] A. Aimar *et al.*, "NullHop: A flexible convolutional neural network accelerator based on sparse representations of feature maps," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 30, no. 3, pp. 644–656, Mar. 2018.

Henri Rebecq received the MSc Eng. degree from Télécom ParisTech, and the MSc degree from Ecole Normale Supérieure de Cachan, France. He is currently working toward the PhD degree in the Robotics and Perception Group, University of Zurich and ETH Zurich, where he is working on on event-based vision. His research interests include 3D reconstruction, SLAM, and computational photography with event cameras. He is a recipient of the Best Industry Paper Award at the British Machine Vision Conference (2016), Misha Mahowald Prize for Neuromorphic Engineering (2017), Qualcomm Innovation Fellowship (2018), and IEEE RA-L 2018 Best Paper (Honorable Mention).

René Ranftl received the MSc and PhD degrees from the Graz University of Technology, Austria, in 2010 and 2015, respectively. He is currently a senior research scientist with the Intel Intelligent Systems Lab in Munich, Germany. His research interests include topics in computer vision, machine learning, and robotics. He is the recipient of several awards including a Best Systems Paper Award at the Conference on Robot Learning 2018 and the Best Paper Award at the International Conference on Scale Space and Variational Methods 2015.

Vladlen Koltun is currently a senior principal researcher and the director of the Intelligent Systems Lab at Intel. His lab conducts high-impact basic research on intelligent systems, with emphasis on computer vision, robotics, and machine learning. He has mentored more than 50 PhD students, postdocs, research scientists, and PhD student interns.

Davide Scaramuzza received the PhD in robotics and computer vision from ETH Zurich and a postdoctoral from the University of Pennsylvania. He is currently a associate professor of robotics and perception at the Department of Informatics (University of Zurich), and Department of Neuroinformatics (University of Zurich and ETH Zurich), where he does research at the intersection of robotics, computer vision, and neuroscience. From 2009 to 2012, he led the European project sFly, which introduced the PX4 autopilot and pioneered visual-SLAM–based autonomous navigation of micro drones. For his research contributions, he was awarded the Misha Mahowald Neuromorphic Engineering Award, IEEE Robotics and Automation Society Early Career Award, European Research Council (ERC) Consolidator Grant, a Google Research Award, European Young Research Award, and several conference paper awards. He coauthored the book Introduction to Autonomous Mobile Robots (published by MIT Press) and more than 100 papers on robotics and perception. In 2019, he was elevated to the grade of IEEE Senior Member.