EventGAN: Leveraging Large Scale Image Datasets for Event Cameras

Alex Zihao Zhu, Ziyun Wang, Kaung Khant, Kostas Daniilidis

Abstract—Event cameras provide a number of benefits over traditional cameras, such as the ability to track incredibly fast motions, high dynamic range, and low power consumption. However, their application into computer vision problems, many of which are primarily dominated by deep learning solutions, has been limited by the lack of labeled training data for events. In this work, we propose a method which leverages the existing labeled data for images by simulating events from a pair of temporal image frames, using a convolutional neural network. We train this network on pairs of images and events, using an adversarial discriminator loss and a pair of cycle consistency losses. The cycle consistency losses utilize a pair of pre-trained self-supervised networks which perform optical flow estimation and image reconstruction from events, and constrain our network to generate events which result in accurate outputs from both of these networks. Trained fully end to end, our network learns a generative model for events from images without the need for accurate modeling of the motion in the scene, exhibited by modeling based methods, while also implicitly modeling event noise. Using this simulator, we train a pair of downstream networks on object detection and 2D human pose estimation from events, using simulated data from large scale image datasets, and demonstrate the networks' abilities to generalize to datasets with real events. The code and dataset in this paper are available here: https://github.com/alexzzhu/EventGAN.

2021 IEEE International Conference on Computational Photography (ICCP) | 978-1-6654-1952-9/21/$31.00 ©2021 IEEE | DOI: 10.1109/ICCP51581.2021.9466265

1 Introduction

Deep learning has led a revolution for many computer vision tasks which had been considered incredibly challenging. The ability to leverage immense amounts of data to train neural networks has resulted in significant improvements in performance for many tasks. As a vision modality, event cameras have a lot to gain from deep learning. By combining the neural networks with the advantages of event cameras, we stand to be able to extend the operating volume of speeds and lighting conditions significantly beyond that which is achievable by traditional cameras.

However, these networks for events are limited by the amount of labeled training data available, due to the camera's relative infancy and the cost of acquiring accurate ground truth labels. While some works have been able to bypass this issue with self-supervised approaches [1], [2], [3], some problems, such as detection and classification, cannot currently be solved without a large corpus of labeled training data. In this work, we focus on an alternative to costly data labelling, by leveraging the large set of labeled image datasets via image to event simulation.

The highest fidelity event camera simulators today [4], [5], [6] all operate with a similar framework, by simulating optical flow in the image either through 3D camera motion, or a parametrized warping (e.g. affine) of the image, in order to precisely track the generation of events as each point in the image moves to a new pixel. However, these scenarios either require simulation of the full 3D scene, or severely constrain the motion in the image. In addition, modeling event noise, both in terms of erroneous events and noise in the event measurements, is a challenging open problem.

• *A. Zhu (alexzhu@seas.upenn.edu),*
Z. Wang (ziyunw@seas.upenn.edu),
K. Khant (khantk@seas.upenn.edu) and
K. Daniilidis (kostas@seas.upenn.edu) are with the Department of Computer and Information Science, University of Pennsylvania, Philadelphia, PA, 19104.

In this work, we present EventGAN, a novel method for image to event simulation, where we apply a convolutional neural network as the function between images and events. By learning this function with data, our method does not require any explicit knowledge of the scene or the relationship between images and events, but is instead able to regress a realistic set of events given only images as input. In addition, our network is able to learn the noise distribution over the events, which are currently not modeled by the competing methods. Finally, our proposed method has a fast, constant time simulation which is easily parallelizable on GPUs and integrable into any modern neural network architecture, as opposed to the prior work which requires 3D simulations of the scene.

Our network is trained on a set of image and event pairs, which are directly output by event cameras such as the DAVIS [7]. At training time, we apply an adversarial loss to align the generated events with the real events. In addition, we pre-train a pair of CNNs to perform optical flow estimation and image reconstruction from real events, and constrain our generator to produce events which allow these pre-trained networks to generate accurate outputs. In other words, we constrain the generated events to retain the motion and appearance information present in the real data.

Using this event simulation network, we train a set of downstream networks to perform object detection on cars and 2D human pose estimation, given images and labels from large scale image datasets such as KITTI [8], MPII [9] and Human3.6M [10]. We then evaluate performance on these downstream tasks on real event datasets, MVSEC [11] for car detection, and DHP19 [12] for human pose, demonstrating the generalization ability of these networks despite having mostly seen simulated data at training time.

Our main contributions can be summarized as:

- A novel pipeline for supervised training of deep neural networks for events, by simulating events from existing large scale image datasets and training on the simulated events and image labels.
- A novel network, EventGAN, for event simulation from a pair of images, trained using an adversarial loss and cycle consistency losses which constrain the generator network to generate events from which pre-trained networks are able to extract accurate optical flow and image reconstructions.
- A test dataset for car detection, with manually labeled bounding boxes for cars from the MVSEC [11] dataset.
- Experiments demonstrating the generalizability of the networks trained on simulated data to real event data, by training object detection and human pose networks on simulated data, and evaluating on real data.

2 Related Work

2.1 Event Simulation

Prior works on event simulation have focused on differencing log intensity frames, in order to simulate the condition required to trigger an event:

$$\|\log(I_{t+1}(\mathbf{x})) - \log(I_t(\mathbf{x}))\| \geq \theta \tag{1}$$

Earlier works by Bi et al. [13] and Kaiser et al. [14] simulating events by directly applying this equation to the log intensity difference between each pair of successive images. These methods were limited by the temporal resolution of these images, and as such could only handle relatively slow moving scenes. To improve fidelity, Rebecq et al. [4], Mueggler et al. [5] and Li et al. [6] perform full 3D simulations of a scene. This allows them to simulate images at arbitrary temporal resolution, while also having access to the optical flow within the scene, allowing for accurate event trajectories. However, these methods are limited to fully simulated scenes, or images where the motion is known (or where a simplified motion model such as an affine transform is applied). Performing 3D simulations is also a relatively expensive procedure, requiring complex rendering engines. In addition, these methods do not properly model the noise properties of the sensor. Rebecq et al. [4] apply Gaussian noise to the trigger threshold, θ, as an approximation, but no true model of the event noise distribution exists to our knowledge. Later, this approach is further improved in Vid2E [15] by applying a Super SlowMo [16] step before simulating events in order to preserve finer temporal information.

Our work, in contrast, runs in constant time using a CNN which is easily parallelizable and optimized for modern GPUs. The network learns both the motion information in the scene, as well as the noise distribution of the events.

2.2 Sim2Real/Domain Adapation

Learning from simulations and other modalities has been a rapidly growing topic, with deep learning approaches for many robotics problems in particular requiring much more training data than is practical to collect on a physical platform. However, this remains a challenging open problem, as conventional simulators often cannot perfectly model the data distribution in the real world, resulting in many methods attempting to bridge this gap [17], [18]. One popular approach to this problem in the image space is the use of Generative Adversarial Networks (GANs) [19], which consists of a generator trained to model the data distribution of the training set, while a discriminator is trained to differentiate between outputs from the fake and real data. With particular relevance to this work, conditional GANs [20], [21] are able to model relationships between data distributions, while CycleGANs apply additional cycle consistency losses [22].

A successful application of cross modality transfer is in the field of image to lidar transform. A number of recent works [23], [24], [25] have approached the problem of simulating lidar measurements from images, which allow networks to better reason about 3D scenes more efficiently.

With a similar motivation to our work, Iacono et al. [26] and Zanardi et al. [27] address the issue of transferring learning from images to events by running a network trained on images on the grayscale images produced by some event cameras such as the DAVIS [7], and using these outputs as ground truth to train a similar network for events. However, these methods treat the frame based outputs as ground truth, and so will learn biases and mistakes made by the frame based network (e.g. the best mAP of the grayscale network in Zanardi et al. [27] is 0.59, resulting in a mAP for the event based network of 0.26).

As an alternative approach, our work follows the philosophy of using GANs for image to event simulation. We then use the simulated events to train directly on the ground truth labels for the corresponding images, which should be at least as accurate if not better than outputs from a frame based network trained on these labels.

3 Method

The generative portion of our pipeline consists of a U-Net [28] encoder-decoder network, as used in Zhu et al. [29] and Rebecq et al. [2]. The generator takes as input a pair of grayscale images, concatenated along the channel dimension, and outputs a volumetric representation of the events, described in Section 3.1. To constrain this output, we apply an adversarial loss, described in Section 3.2, as well as a pair of cycle consistency losses, described in Section 3.3. The full pipeline for our method can be found in Figure 1.

3.1 Event Representation

The most compact way to represent a set of events is as a set of 4-tuples, consisting of the x, y position, timestamp, t, and polarity, p. However, regressing points in general is a difficult task, and faces challenges such as varying numbers of events and permutation invariance.

In this work, we bypass this issue by instead regressing an intermediate representation of the events as proposed by Zhu et al. [29]. In this representation, the events are scattered into a fixed size 3D spatiotemporal volume, where

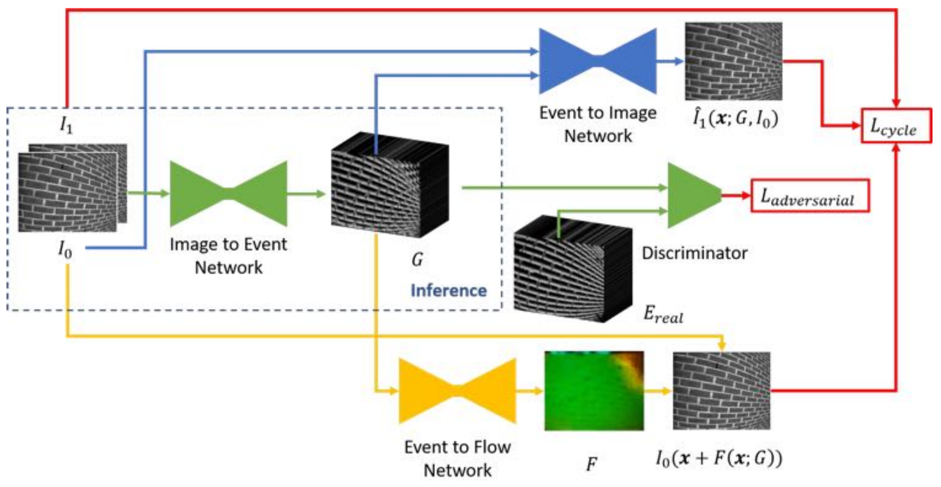


Fig. 1: Overview of the EventGAN pipeline. A pair of grayscale images are passed into the generator, which predicts a corresponding event volume. This output is constrained by an adversarial loss, as well as a pair of cycle consistency losses which constrain the generated volume to encode image and flow information.

each event, (x, y, t, p) is inserted into the volume, which has $B = 9$ temporal channels, with a linear kernel:

$$t_i^* = (B-1)(t_i - t_1)/(t_N - t_1) \tag{2}$$

$$V(x, y, t) = \sum_i \max(0, 1 - |t - t_i^*|) \tag{3}$$

This retains the distribution of the events in x-y-t space, and has shown success in a number of tasks [2], [29], [30].

However, we deviate from the prior work in that we generate separate volumes for each polarity, and concatenate them along the time dimension. This results in a volume which is strictly non-negative, allowing for a ReLU as the final activation of the network, such that the sparsity in the volume is easily preserved.

In addition, we normalize this volume similar to Rebecq et al. [2], with an additional clipping step, as follows:

$$\hat{V}(x, y, t) = \frac{\min(V(x, y, t), \eta_{98})}{\eta_{98}} \tag{4}$$

where η_{98} is the 98th percentile value in the set of non-zero values of V. This equates to a clipping operation, followed by a normalization such that the volume lies in $[0, 1]$. The clipping is designed to reduce the effect of hot pixels, which have an erroneously low contrast thresholds and thus generate a disproportionately many events, skewing the range.

3.2 Adversarial Loss

Perhaps the most direct way to supervise this network is to apply a direct numerical error, such as a L1 or L2 loss, between the predicted and real events. However, given a pair of images, the number of plausible event distributions between the images is extremely large (two images can not constrain the exact motion in between them). Such a direct loss would likely cause the network to overfit to the trajectories observed in the training set and fail to generalize.

Instead, we apply an adversarial loss [19]. This loss simply constrains the generated events to follow the same distribution as the real ones, and avoids directly constraining the network to memorizing the trajectories seen at training time. For each event-image pair, (x, y), we regress a generated event volume using our network, G, and then pass the generated events and real events through a discriminator network, D, which predicts the probability that its input is from real data. Our discriminator is a 4 layer PatchGAN classifier [21]. We alternatingly train the generator and discriminator, with the discriminator trained 2 steps for every 1 of the generator, using the hinge adversarial loss [31], [32]:

$$\begin{aligned}\mathcal{L}_D = &-\mathbb{E}_{(x,y)\sim p_{\text{data}}}[\min(0, -1 + D(x, y))] \\ &-\mathbb{E}_{y\sim p_{\text{data}}}[\min(0, -1 - D(G(y), y))] \end{aligned} \tag{5}$$

$$\mathcal{L}_G = -\mathbb{E}_{y\sim p_{\text{data}}} D(G(y), y) \tag{6}$$

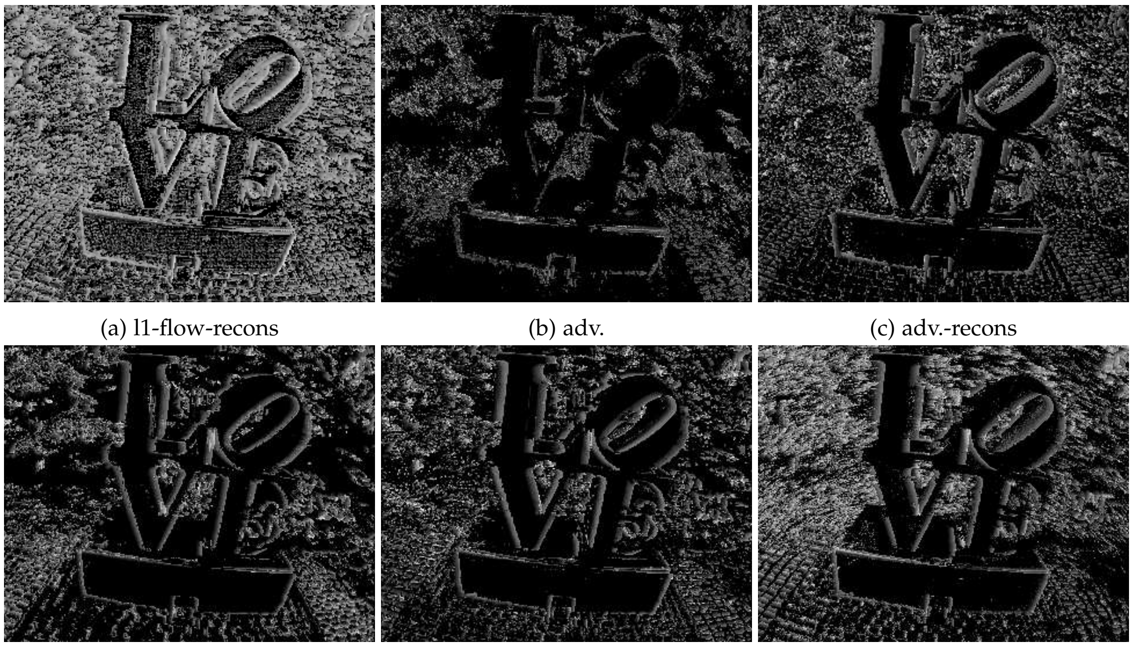
3.3 Cycle Consistency Losses

However, GANs are typically difficult to train, especially with a high dimensional output space such as an event volume. In addition, there are no guarantees on the simulated events retaining the salient information in the images, such as accurate motion and intensity information.

To this end, we apply an additional pair of losses which constrain the generated events to encode this motion and intensity information. In particular, we pre-train a pair of networks for optical flow estimation and image reconstruction from real events, using the pipeline in EV-FlowNet [1].

The flow network takes as input the event volume, and outputs a per pixel optical flow. Supervision is applied by warping the previous image to the time of the next image using the predicted flow, and applying an L1 loss between the warped and original image, as well as a local smoothness constraint.

The image reconstruction network takes as input the previous image and the event volume, and outputs the predicted next image, and is directly supervised by a L1 loss between the reconstructed and original image. The previous image is provided as input as we found that the image reconstruction network tended to overfit to the training set without it. Prior work by Rebecq et al. [2]



(a) l1-flow-recons (b) adv. (c) adv.-recons

(d) adv.-flow (e) adv-flow-rec (f) real

Fig. 2: Outputs from models trained with subsets of our proposed loss, all models trained with the same hyperparameters. Events are visualized as average timestamp images, i.e. the average timestamp at each pixel. Any voxel with non zero value will generate a color in the average timestamp image, allowing us to see the sparsity of the volume. **(a)**: L1 reconstruction loss in place of the adversarial loss, causing artifacts in the events, and no sparsity achieved, as observed in the interior of the 'LOVE' symbol in the time image. **(b)**: Adversarial loss only. Model struggles to converge, and requires significant hyperparameter tuning in order to achieve good results. **(c)**: Adversarial loss and reconstruction loss. Model is now stable, but the events do not have motion information. The image should have a gradient in the motion direction. **(d)**: Adversarial loss and flow loss. Motion direction can now be seen in the time image, but events are not generated in many areas. **(e)**: Adversarial loss, flow and reconstruction losses. Motion trails can now be clearly seen in the time image (see letters). **(f)**: Real events. Note that our method typically underestimates the amount of motion in the scene.

has circumvented this by training in a recurrent fashion, but doing so would require multiple passes through the recurrent network, which is undesirably expensive when the goal is to train the generator network. In addition, we summarize the event volume by summing along the time dimension. This is to maintain the invariance to permutation across time of the events. For example, two events occurring at the start of the window vs. two events at the end of the window should generate the same output image. The input, then, to the reconstruction network, is a 2-channel image consisting of the previous image and the summed event volume.

In summary, the cycle consistency losses are:

$$\mathcal{L}_F = \sum_{\mathbf{x}} \|I_0(\mathbf{x} - F(\mathbf{x}; G)) - I_1(\mathbf{x})\|_1 + \lambda_1 \left(\left\| \frac{dF}{dx}(\mathbf{x}; G) \right\|_1 + \left\| \frac{dF}{dy}(\mathbf{x}; G) \right\|_1 \right) \tag{7}$$

$$\mathcal{L}_R = \sum_{\mathbf{x}} \|\hat{I}_1(\mathbf{x}; G, I_0) - I_1(\mathbf{x})\|_1 \tag{8}$$

$$\mathcal{L}_{\text{cycle}} = \mathcal{L}_F + \mathcal{L}_G \tag{9}$$

When training the generator network, we pass the output from the generator as input to each of the pre-trained networks, and apply the same losses used to train each. However, in this case, we freeze the weights of each pre-trained network, such that the generator must tune its output to generate the best input for each pre-trained network. Both cycle consistency networks share the same architecture as the generator network, with the losses applied each time the generator is updated in the adversarial framework. The final losses at each step are:

$$\text{Generator step:} \qquad \mathcal{L}_{GS} = \mathcal{L}_G + \mathcal{L}_{\text{cycle}} \tag{10}$$

$$\text{Discriminator step:} \qquad \mathcal{L}_{DS} = \mathcal{L}_D \tag{11}$$

These losses provide useful gradients early in training, when the adversarial loss is typically unstable, and embed motion and appearance information in the predicted event volumes. Figure 2 shows the effect of each loss on the output of the generator.

In summary, the adversarial loss enforces sparsity in the event volume and similarity between the fake and real event distributions. The flow loss enforces motion information to be present within the volume, while the reconstruction loss enforces regularity in the number of events generated by the same point. This is particularly evident when one visualizes the image of the average timestamp at each pixel, where extremely low (but non-zero) values may be hidden in the count image, and where motion trails are clearly visible.

We also implement the tips prescribed by Gulrajani et al. [33] and Brock et al. [34]. In particular, we apply spectral normalization [35] in the encoder of the generator, and batch normalization [36] for the entire generator, while the discriminator has neither types of normalization. We also add noise to the labels seen by the discriminator by randomly flipping the labels from real to fake 10% of the time, as recommended by Chintala et al. [37].

4 Experiments

We train our network using the RAdam [39] optimizer for 100 epochs on events and images from the indoor_flying and outdoor_day sequences in the MVSEC dataset [11], as well as a newly collected dataset consisting of recordings from a DAVIS-346b camera [7], consisting of short (<60s) sequences with a number of different scenes and motions, in order to capture a large range of event distributions. As the objective of this work is to produce an event simulator which operates well on existing image datasets, we did not train on scenes which are challenging for images (e.g. night time driving). In total, the training set consists around 30 mins of data. During training, we perform weighted sampling from this dataset, with a 80%/20% split between the new data and MVSEC. Each input to the network consists of a pair of images, randomly picked between 1 and 6 frames apart, and the events between them.

Quantitative evaluations of generative models is difficult, as measuring how well the predicted events fit the true event distribution requires knowledge of the true event distribution. In general, the mapping from images to events is one-to-many. For images, networks trained a large corpus of image data are used to model these distributions, and metrics such as the Inception Score [40] or the Fréchet Inception Distance [41] are applied using these networks. However, this results in a second chicken and egg problem, as no such corpus of event data currently exists.

Instead, we primarily evaluate our method directly on a set of downstream tasks, and demonstrate that our simulated events are able to train networks for complex tasks which generalize to data with real events. In Sections 4.1 and 4.2 we describe our experiments for 2D human pose estimation and object detection, respectively.

Finally, in Section 5.3, we provide experiments where we measure the difference between our predicted events and the observed ones. While this is an imperfect measure, we show that we outperform competing work in Vid2E [15].

4.1 2D Human Pose Estimation

We train a 2D human pose detector for events based on the publicly available code from Xiao et al. [42], which uses an encoder-decoder style network to regress a heatmap for each desired joint. We use a ResNet-50 [43] encoder, pretrained on ImageNet [44]. For event inputs, we modify the number of input channels in the first layer, and randomly initialize the weights of this layer. The network is then trained on a 80%/20% split between the MPII [9] and Human3.6M [10] datasets. For each ground truth pose, the pair of images either 1 or 2 frames before and after the target frame are selected at random, and passed into the generator network to generate a simulated event volume.

We evaluate our method on the DHP19 [12] dataset, which consists of 3D joint positions of a human subject, recorded with motion capture, with events from four cameras surrounding the subject. Using the camera calibrations, we project these 3D joint positions into 2D image positions for each camera. Following the experiment schedule by Calabrese et al. [12], we use as a test set data from subjects 13-17 and cameras 2-3. As our method does not include any temporal consistency, we remove sequences with hand motions only, where most of the body is static and does not generate any events. This results in 16 motions across 5 subjects and 2 cameras. Following Calabrese et al. [12], we divide each sequence into chunks of 7500 events per camera, and evaluate on the average pose within each window. One issue with this direct evaluation is that the marker positions for DHP19 vary significantly from those in MPII and H36M. In order to overcome this offset between the joint positions, we freeze all but the final linear layer of our network, and fine tune this layer on the DHP19 training set (subjects 1-12, cameras 2-3). This is equivalent to training a linear model on the activations from the second to last layer, as is common in the self-supervised learning literature [46].

4.2 Object Detection

We train a detection network using the YOLOv3 pipeline [38]. We initialize the network from a pretrained YOLOv3 network with spatial pyramid pooling, with the first input layer randomly initialized. The network is trained on simulated events from the KITTI Object Detection dataset [8], with the target frame and either the frame one or two frames prior.

4.3 The Event Car Detection Dataset

For evaluation, we generated a novel dataset for car bounding box annotations for event data. Our dataset consists of 250 labeled images from the MVSEC [11] outdoor driving dataset, with corresponding timestamps. For each image, raters label bounding boxes for all cars within the scene, while also separating the cars into easy (large, no occlusion), hard (medium, or partial occlusion) or don't care (mostly occluded or too small) categories. In total, there are 451 easy instances, 506 hard instances and 959 don't care instances. This dataset will be publicly available.

4.4 Competing Methods

We additionally simulate the MPII, H36M and KITTI datasets using ESIM [4], by simulating a random affine transform of each image in the dataset, similar to the method used by Rebecq et al. [2]. Using this simulated data, we train the same networks described in Sections 4.1 and 4.2. For both experiments, we also train networks on real data as a baseline. For object detection, we train a network on the grayscale frames from KITTI, and evaluate on the grayscale frames from MVSEC and DDD17. For human pose estimation, we train a network on the events in the training set (subjects 1-12) of DHP19.

5 Results

5.1 2D Human Pose Estimation

We evaluate our method on the mean per joint position error (MPJPE) [12], $\frac{1}{N}\sum_i^N \|x_i - \hat{x}_i\|_2$, as well as PCKh@50

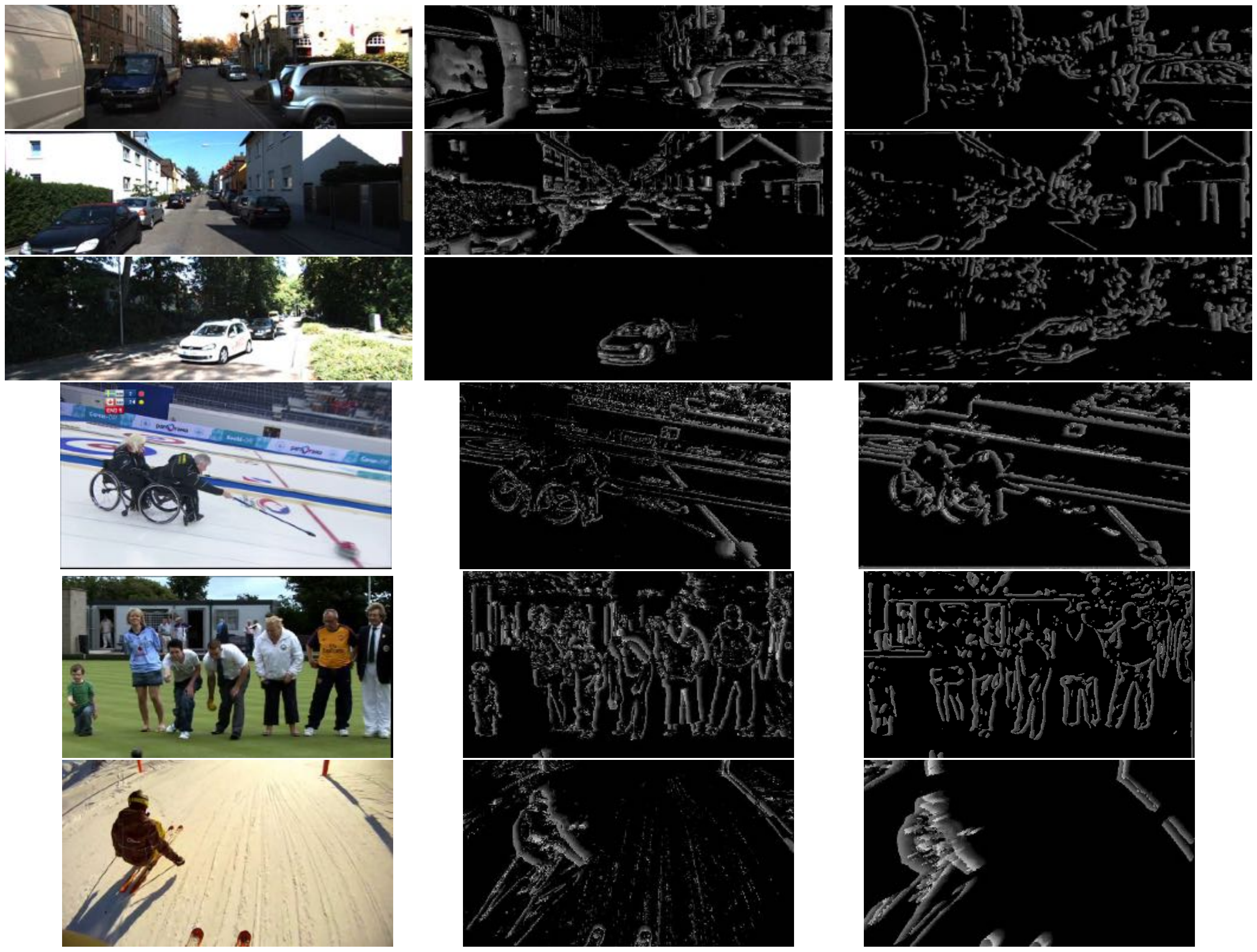


Fig. 3: Sample outputs generated by EventGAN, compared to ESIM [2], visualized as images of the average timestamp at each pixel. Top images are from the KITTI dataset [8], bottom are from MPII [9]. Compared to ESIM, our method is able to more accurately capture the motion in the scene, and capture fine grain information.

Training Data	Precision	Easy recall	Hard recall	Comb recall	AP	F-1
EventGAN	0.42	**0.57**	**0.34**	**0.45**	**0.30**	0.44
ESIM	0.23	0.08	0.02	0.05	0.02	0.09
Frame	**0.57**	0.48	0.27	0.37	0.29	**0.45**

TABLE 1: Object detection results on the Event Car Detection dataset. Metrics adopted from the PASCAL VOC challenge [45]. The EventGAN and ESIM models are trained on simulated events from the KITTI dataset, while the Frame model is trained on the real image frames from the KITTI dataset.

	Pretrained only		1 Epoch			30 Epochs			140 epochs
	EventGAN	ESIM	EventGAN	ESIM	Real	EventGAN	ESIM	Real	Real
MPJPE ↓	**14.55**	19.57	**6.76**	7.58	8.94	**6.44**	6.54	6.75	**6.39**
PCKh@50 ↑	**45.47**	40.53	**87.70**	85.89	80.55	**90.19**	89.93	87.53	89.86

TABLE 2: Human pose estimation results in MPJPE (pix.) (lower is better) and PCKh@50 (higher is better). All EventGAN and ESIM models are first pretrained on simulated events from the MPII and H36M datasets, and then the final linear layer is fine tuned on the DHP19 training set for the specified number of epochs. The Real models are trained directly (whole model) on the DHP19 training set for the specified number of epochs.

(percentage of correct keypoints) [9], which measures the percentage of joint predictions with error less than 50% of

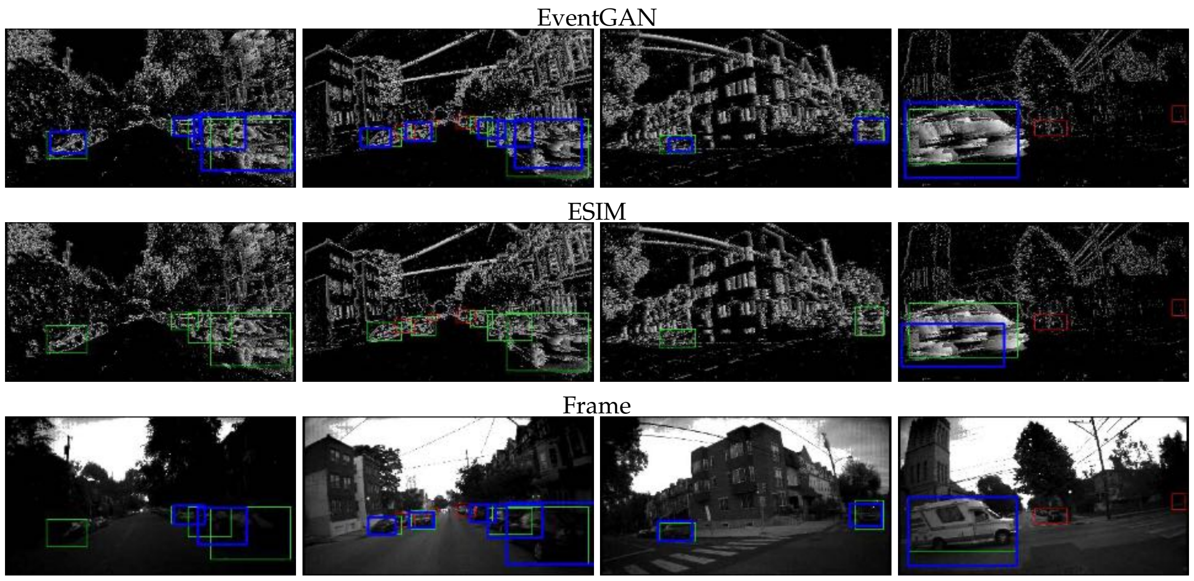


Fig. 4: Selected qualitative results of our car detection pipeline using the YOLOv3 network [38]. Detections are in blue, GT labels in green, and don't care regions in red. For explanation of the methods, please see Table 1.

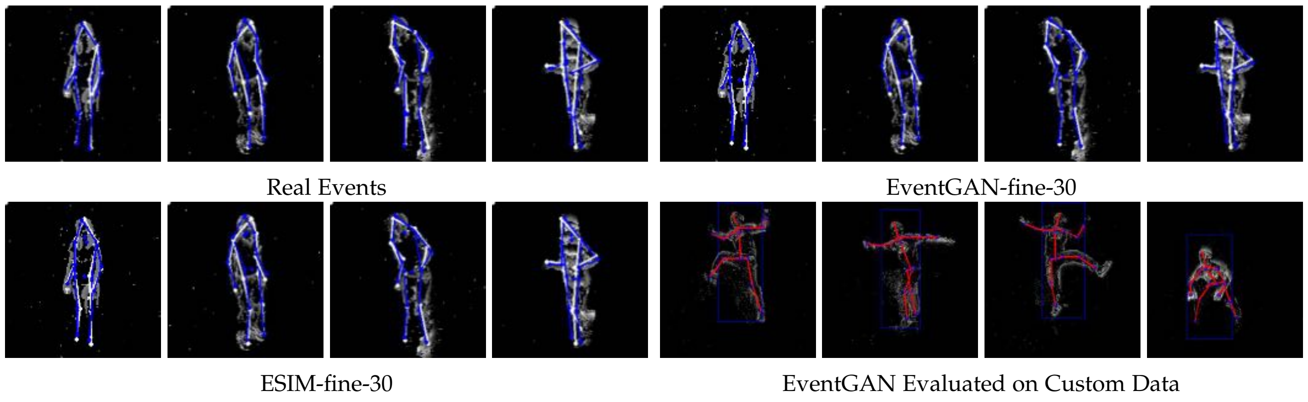


Fig. 5: Qualitative results of our human pose estimation on real event data. The first three sets are evaluated on samples from the DHP19 dataset [12], where ground truth is in white and predictions are in blue. Our model is able to achieve accuracy on par with a model directly trained on the real data after 30 epoch of fine tuning only the last linear layer. The last set shows our YOLOv3 detection pipeline combined with our human pose estimator. The detection network is trained on MPII to detect the human in the scene (blue box), which is fed into the human pose estimator to estimate the 2D joint positions (MPII format). Best viewed in color.

the head size. We define head size as $0.6\times$ the distance between the head and the midpoint between the shoulders.

In Table 2, we compare a network trained on simulated events from EventGAN, ESIM, and a network trained directly on the DHP19 training set. We also report results from fine tuning the final linear layer of the network on the DHP19 training set for both EventGAN and ESIM. Qualitative results from both DHP19 and out of sample data can be found in Figure 5. From these results, we can see that the data generated by EventGAN is able to train a network to learn representations that are very close to the true data. After only one epoch of fine tuning, and only of the final layer, we are able to achieve significantly higher accuracy than training on the real data, and come close to the accuracy of a network trained for 140 epochs on real data. However, the gap between ESIM and our method is also relatively small. This is largely due to the low difficulty of the dataset, as even training on real events converges to a relatively good solution after only one epoch of training. This was observed even when testing with much smaller networks, although they converge to a lower accuracy. The dataset is also much cleaner, and as such is closer to the ESIM outputs.

5.2 Object Detection

We evaluate our method according to the precision-recall statistics defined by the PASCAL VOC challenge [45]. Predictions with confidence < 0.2 are removed, and non-maximum suppression is applied for boxes with IoU > 0.2. In total, we report precision, recall on the easy and hard classes, as well as the AP and F-1 scores for each training input in Table 1. We provide qualitative results in Figure 4.

From these results, we observed that our method is able to achieve reasonably strong results, and comes close to matching the performance of the network with frame inputs, which was trained on real data. The difference in performance implies a small sim-to-real gap, but may also simply be due to a stronger signal in the images for certain frames (although this may also be true the other way round). On the other hand, the sim-to-real gap is significant when training on ESIM. As the true event distribution differs largely from the simulated data, the network is only able to perform accurate detections when the input has relatively low noise (e.g. Figure 4 right), resulting in very low recall.

5.3 Direct Quantitative Evaluation

In addition to the evaluation based on the downstream tasks, we provide quantitative and qualitative evaluation of our method against competing methods on the night sequence of MVSEC [11]. We compare our method against the approach described in Vid2E [15], where a video interpolation [16] step is performed before feeding the images to the ESIM [4] simulator. Note that EventGAN could also take advantage of the interpolated image frames, which is why we directly compare with the ESIM generator in the main experiments. **Quantitative Evaluation** We directly calculate the L2 distance and classification score between the generated volumes and the ground truth volumes. For a volume E and ground truth $\hat{E}$, the L2 distance is calculated by:

$$||E - \hat{E}||_2 = \sqrt{\sum_{x,y,t,p} (E(x,y,t,p) - \hat{E}(x,y,t,p))^2}$$

For classification score, we first scale the predicted event volumes so that the max value matches that of the ground truth, to minimize the effect of the contrast threshold. Then, we classify an event bin as correct if its value is within some threshold of the ground truth. We use two of such thresholds: 0.5 and 1, and report the proportion of correct classifications for each volume. The values in an event volume can be seen as the number of events, and a threshold of 1 means the error in prediction is within 1 event, although this is dominated by no event voxels. To get the best performance from Vid2E [15], we run a parameter sweep of contrast threshold from 0.1 to 1.7 and refractory period from 10^{-5} to 10^{-1}. For the sake of space, we report the result for the best refractory period for each contrast threshold based on L2 distance.

As shown in **Table 3**, EventGAN outperforms different configurations of the competing methods in all of these metrics. Note that the best contrast threshold found in this search was outside the default range (0 to 1) specified in the V2E paper, which shows it is a difficult task to find the contrast threshold that best emulates the event generation for a real event-based camera. In contrast, our network learns this information from the dataset and avoids the expensive search procedure. Normalizing the event volumes alleviates the problem to some extent, but it fails to account for the change of event distribution because thresholding the log-intensity is inherently a non-linear function. We observe this in **Table 3**, where different contrast thresholds produce vastly different errors. On the other hand, EventGAN provides an effective method for generating events without expensive video interpolation or hyperparameter search.

Method	L2	Acc(0.1)	Acc(0.5)	Acc(1)
Vid2E, c=0.1	101.489	0.867	0.945	0.977
Vid2E, c=0.3	100.727	0.889	0.946	0.978
Vid2E, c=0.5	100.281	0.906	0.943	0.979
Vid2E, c=0.7	99.917	0.916	0.945	0.979
Vid2E, c=0.9	99.071	0.924	0.949	0.979
Vid2E, c=1.1	99.282	0.931	0.952	0.979
Vid2E, c=1.3	98.480	0.940	0.958	0.981
Vid2E, c=1.5	99.414	0.942	0.960	0.981
Vid2E, c=1.7	98.092	0.948	0.964	0.984
EventGAN	**91.786**	**0.957**	**0.972**	**0.992**

TABLE 3: Comparison between EventGAN and [15] on samples of MVSEC [11] night driving sequence 1. For L2, We normalize the generated volumes by setting the max value of each volume to 1. We evalute the accuracy with three different thresholds.

5.4 Qualitative Evaluation

To provide concrete examples of the quality of generated events, we show a pair of zoom-in views of the event volumes generated with EventGAN versus those generated with the competing methods in Figure 6. It can be seen from these images that EventGan is able to produce the noise properties that resemble those from the actual camera. Competing methods produce reasonable but overly clean events. Qualitatively, we find the events from ESIM to be closer to the events generated with direct L1/L2 supervision in our experiments. By including a pair of adversarial losses, EventGAN takes advantage GAN's ability to model complex image distributions and produces realistics events without making simplifying assumptions about the noise distribution. In addition, EventGan retains more fine-grained details of the captured scene without the need to tuning the contrast threshold.

6 Implementation Details

6.1 Training

Our generator network, and flow and reconstruction networks, share the same U-Net style architecture used by Zhu et al. [29] and Rebecq et al. [2]. In particular, the network has four encoder layers and four decoder layers, with two residual blocks [43] in between. The input is a discretized event volume with nine bins along the temporal dimension, resulting in a 18 channel input when accounting for polarity. The activations of the first layer have 32 channels, and double for each encoder layer, reaching a max of 512, and then halving for each decoder layer. We train using the RAdam [39] optimizer for 100 epochs.

We also implement the tips prescribed by Gulrajani et al. [33] and Brock et al. [34]. In particular, we apply spectral normalization [35] in the encoder of the generator, and batch normalization [36] for the entire generator, while the

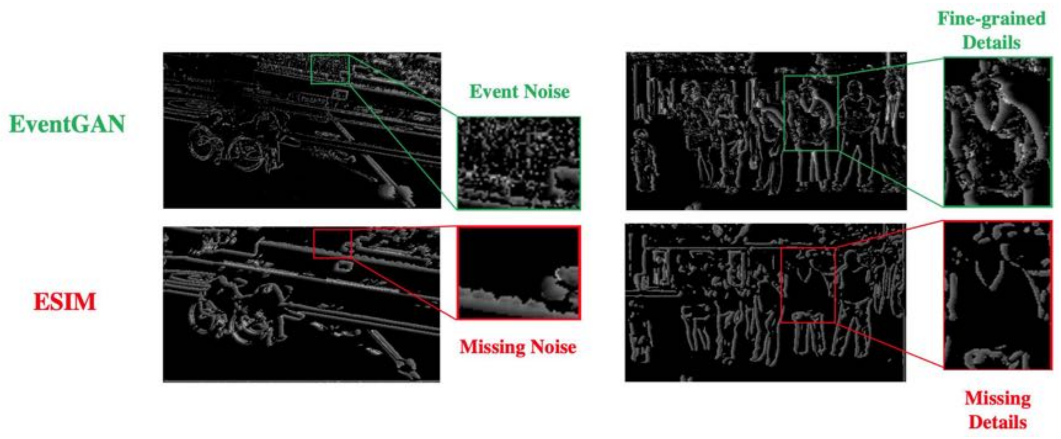


Fig. 6: Qualitative comparison between events generated from EventGAN and from ESIM [4]. Top two figures: events from EventGAN. Bottom two figures: events from ESIM [4].

discriminator has either types of normalization. We also add noise to the labels seen by the discriminator by randomly flipping the labels from real to fake 10% of the time, as recommended by Chintala et al. [37].

6.2 The Event Car Detection Dataset

For this dataset, we also explored the use of the DDD17 dataset [47], while contains 12 hours of driving data recorded from a DAVIS 346b [7] camera. However, we found that, due to the positioning of the camera behind the windshield, the majority of cars in each sequence were extremely small, and the positioning of the cameras and the scenes viewed differed significantly from the training data in KITTI. As a result, the trained YOLO network failed to produce reasonable detections. Small objects are a challenging problem for detectors in general, and, as the objective of the evaluation was to compare between the training data and input modality, we have omitted this dataset from the experiments.

6.3 ESIM

We generate simulated events using ESIM by following the procedure described by Rebecq et al. [2]. In particular, we use the Multi Objects 2D renderer [1], which allows us to simulate motion from a single image by applying a random affine transformation to the image. In particular, we apply a random translation, scaling and rotation to the image, between -0.1 and 0.1 radians for rotation, and between -10% and 10% of the image size for translation and scaling. We then keep the first 50000 simulated events, which are used to generate a discretized event volume which is used for training. As this process is relatively slow (6s per image), simulation is performed as a pre-processing step. Where possible, we applied the default parameters recommended by the package. We also experimented with the "Simulating events from a video" option for ESIM [2], but found that the frame rate for the datasets used in this work (KITTI, MPII and H36M) was too low for this method.

1. https://github.com/uzh-rpg/rpg_esim/wiki/Multi-Objects-2D-renderer
2. https://github.com/uzh-rpg/rpg_esim/wiki/Simulating-events-from-a-video

7 Discussion

In this section, we discuss a few important questions that we consider during the development of the approach. These questions motivate us to follow the current design of EventGan.

Choice of Loss Functions: to train EventGan, we choose to combine the a pair of adversarial losses with a set of auxiliary losses to ensure the consistency of the generated events with the input images (flow, image reconstruction, etc.). We chose not to use a direct L1/L2 loss, as this would force the network to learn a direct 1-to-1 mapping between input images and output events. This would cause the network to struggle if presented with identical images with different sets out events (from different intermediary motions between the images). On the other hand, an adversarial loss allows the network to learn a distribution over the image to event mapping, and so the generator only needs to output a feasible set of events, rather than the exact set that was captured in the data.

On the other hand, the reconstruction loss allows the network to learn output events that could generate the subsequent image, but does not have any constraint over realistic motions. For example, imagine a dot which moves 5 pixels to the right. A feasible set of events for the reconstruction loss would be negative events over where the dot originated, and positive events at the new location, where the dot 'teleports' in the image. The flow loss forces there to be a smooth trajectory of events (as seen at training time for the flow network), in order for accurate flow estimation. Conversely, the flow loss does not apply a constraint over the number of events, which the reconstruction loss provides.

Choice of Time Slices: a natural question in dealing with event volumes is the choice of temporal bins in the event volume and how it affects the quality of event generation. We agree that using a limited number of temporal bins (nine in our experiments) might suffer in high-speed scenarios between two consecutive frames. However, our network is designed to be agnostic to the length of time over which the event volume spanned, due to the normalization of the timestamps in the volume and our random sampling of pairs of temporal frames over different time durations. As such, the volume can be generated over very small time intervals (e.g. 10ms), where each slice captures a very small time interval. This has been demonstrated to work for very fast motions in prior works such as [2], [29].

8 Conclusions

In this work, we have proposed a novel method for training supervised neural networks for events using image data by way of image to event simulation. Given events and images from an event camera, our deep learning pipeline is able to accurately simulate events from a pair of grayscale images from existing image datasets. These events can be used to train downstream networks for complex tasks such as object detection and 2D human pose estimation, and generalize to real events.

The largest limitation of this work is the need for a pair of frames (video), thus prohibiting the use of larger image datasets such as ImageNet [44] and COCO [48]. While it is possible to train a GAN to predict events from a single image, this would become a complex future prediction task, as the GAN must hallucinate the motion within the image. Other promising future directions include exploring other event representations, more complicated adversarial architectures, and exploring more complex downstream tasks.

Acknowledgments

Thanks to Tobi Delbruck and the team at iniLabs forproviding and supporting the DAVIS-346b cameras. Thanks to George Pavlakos for supporting our human pose models, and Ken Chaney for the training infrastructure. We also gratefully appreciate support through the following grants: NSF IIS 1703319 and the DARPA-SRC C-BRIC.

References

[1] A. Z. Zhu and L. Yuan, "EV-FlowNet: Self-supervised optical flow estimation for event-based cameras," in *Robotics: Science and Systems*, 2018.

[2] H. Rebecq, R. Ranftl, V. Koltun, and D. Scaramuzza, "Events-to-video: Bringing modern computer vision to event cameras," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 3857–3866.

[3] C. Ye, A. Mitrokhin, C. Fermüller, J. A. Yorke, and Y. Aloimonos, "Unsupervised learning of dense optical flow, depth and egomotion from sparse event data," *arXiv preprint arXiv:1809.08625*, vol. 25, 2019.

[4] H. Rebecq, D. Gehrig, and D. Scaramuzza, "ESIM: An open event camera simulator," in *Conference on Robot Learning*, 2018, pp. 969–982.

[5] 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," *The International Journal of Robotics Research*, vol. 36, no. 2, pp. 142–149, 2017.

[6] W. Li, S. Saeedi, J. McCormac, R. Clark, D. Tzoumanikas, Q. Ye, Y. Huang, R. Tang, and S. Leutenegger, "Interiornet: Mega-scale multi-sensor photo-realistic indoor scenes dataset," in *29th British Machine Vision Conference 2018*, 2018.

[7] C. Brandli, R. Berner, M. Yang, S.-C. Liu, and T. Delbruck, "A 240× 180 130 db 3 μs latency global shutter spatiotemporal vision sensor," *IEEE Journal of Solid-State Circuits*, vol. 49, no. 10, pp. 2333–2341, 2014.

[8] A. Geiger, P. Lenz, and R. Urtasun, "Are we ready for autonomous driving? The KITTI vision benchmark suite," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

[9] M. Andriluka, L. Pishchulin, P. Gehler, and B. Schiele, "2D Human pose estimation: New benchmark and state of the art analysis," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

[10] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu, "Human3.6M: Large scale datasets and predictive methods for 3D human sensing in natural environments," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 7, pp. 1325–1339, jul 2014.

[11] A. Z. Zhu, D. Thakur, T. Özaslan, B. Pfrommer, V. Kumar, and K. Daniilidis, "The multivehicle stereo event camera dataset: An event camera dataset for 3D perception," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 2032–2039, 2018.

[12] E. Calabrese, G. Taverni, C. Awai Easthope, S. Skriabine, F. Corradi, L. Longinotti, K. Eng, and T. Delbruck, "DHP19: Dynamic vision sensor 3D human pose dataset," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 0–0.

[13] Y. Bi and Y. Andreopoulos, "PIX2NVS: Parameterized conversion of pixel-domain video frames to neuromorphic vision streams," in *2017 IEEE International Conference on Image Processing (ICIP)*. IEEE, 2017, pp. 1990–1994.

[14] J. Kaiser, J. C. V. Tieck, C. Hubschneider, P. Wolf, M. Weber, M. Hoff, A. Friedrich, K. Wojtasik, A. Roennau, R. Kohlhaas *et al.*, "Towards a framework for end-to-end control of a simulated vehicle with spiking neural networks," in *2016 IEEE International Conference on Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR)*. IEEE, 2016, pp. 127–134.

[15] D. Gehrig, M. Gehrig, J. Hidalgo-Carrió, and D. Scaramuzza, "Video to events: Recycling video datasets for event cameras," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 3586–3595.

[16] H. Jiang, D. Sun, V. Jampani, M.-H. Yang, E. Learned-Miller, and J. Kautz, "Super slomo: High quality estimation of multiple intermediate frames for video interpolation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2018, pp. 9000–9008.

[17] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-real transfer of robotic control with dynamics randomization," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2018, pp. 1–8.

[18] S. James, P. Wohlhart, M. Kalakrishnan, D. Kalashnikov, A. Irpan, J. Ibarz, S. Levine, R. Hadsell, and K. Bousmalis, "Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 12 627–12 637.

[19] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[20] M. Mirza and S. Osindero, "Conditional generative adversarial nets," *arXiv preprint arXiv:1411.1784*, 2014.

[21] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1125–1134.

[22] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2223–2232.

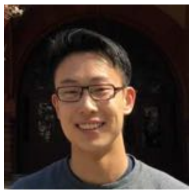
[23] Y. You, Y. Wang, W.-L. Chao, D. Garg, G. Pleiss, B. Hariharan, M. Campbell, and K. Q. Weinberger, "Pseudo-LiDAR++: Accurate depth for 3d object detection in autonomous driving," *arXiv preprint arXiv:1906.06310*, 2019.

[24] Y. Wang, W.-L. Chao, D. Garg, B. Hariharan, M. Campbell, and K. Q. Weinberger, "Pseudo-LiDAR from visual depth estimation: Bridging the gap in 3d object detection for autonomous driving," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 8445–8453.

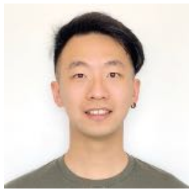
[25] X. Weng and K. Kitani, "Monocular 3D object detection with pseudo-LiDAR point cloud," *arXiv preprint arXiv:1903.09847*, 2019.

[26] M. Iacono, S. Weber, A. Glover, and C. Bartolozzi, "Towards event-driven object detection with off-the-shelf deep learning," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2018, pp. 1–9.

[27] A. Zanardi, A. Aumiller, J. Zilly, A. Censi, and E. Frazzoli, "Cross-modal learning filters for rgb-neuromorphic wormhole learning," *Robotics: Science and System XV*, p. P45, 2019.
[28] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical image computing and computer-assisted intervention*. Springer, 2015, pp. 234–241.
[29] A. Z. Zhu, L. Yuan, K. Chaney, and K. Daniilidis, "Unsupervised event-based learning of optical flow, depth, and egomotion," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2019, pp. 989–997.
[30] K. Chaney, A. Zihao Zhu, and K. Daniilidis, "Learning event-based height from plane and parallax," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2019, pp. 0–0.
[31] J. H. Lim and J. C. Ye, "Geometric gan," *arXiv preprint arXiv:1705.02894*, 2017.
[32] D. Tran, R. Ranganath, and D. Blei, "Hierarchical implicit models and likelihood-free variational inference," in *Advances in Neural Information Processing Systems*, 2017, pp. 5523–5533.
[33] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," in *Advances in neural information processing systems*, 2017, pp. 5767–5777.
[34] A. Brock, J. Donahue, and K. Simonyan, "Large scale GAN training for high fidelity natural image synthesis," in *International Conference on Learning Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=B1xsqj09Fm
[35] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=B1QRgziT-
[36] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning*, 2015, pp. 448–456.
[37] S. Chintala, E. Denton, M. Arjovsky, and M. Mathieu, "How to train a gan," in *NIPS 2016 Workshop on Adversarial Training*, 2016.
[38] J. Redmon and A. Farhadi, "YOLOv3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.
[39] L. Liu, H. Jiang, P. He, W. Chen, X. Liu, J. Gao, and J. Han, "On the variance of the adaptive learning rate and beyond," *arXiv preprint arXiv:1908.03265*, 2019.
[40] T. Salimans, I. Goodfellow, W. Zaremba, V. Cheung, A. Radford, and X. Chen, "Improved techniques for training gans," in *Advances in neural information processing systems*, 2016, pp. 2234–2242.
[41] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "GANs trained by a two time-scale update rule converge to a local nash equilibrium," in *Advances in Neural Information Processing Systems*, 2017, pp. 6626–6637.
[42] B. Xiao, H. Wu, and Y. Wei, "Simple baselines for human pose estimation and tracking," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 466–481.
[43] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.
[44] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.
[45] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (VOC) challenge," *International journal of computer vision*, vol. 88, no. 2, pp. 303–338, 2010.
[46] P. Goyal, D. Mahajan, A. Gupta, and I. Misra, "Scaling and benchmarking self-supervised visual representation learning," *arXiv preprint arXiv:1905.01235*, 2019.
[47] J. Binas, D. Neil, S.-C. Liu, and T. Delbruck, "Ddd17: End-to-end davis driving dataset," in *International Conference on Machine Learning, Workshop on Machine Learning for Autonomous Vehicles (MLAV 2017)*, 2017.
[48] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," in *European conference on computer vision*. Springer, 2014, pp. 740–755.



Alex Zihao Zhu received his BSc in Electrical and Computer Engineering and Computer Science, and his MSc and PhD in Computer and Information Science at the University of Pennsylvania. His research interests are in computer vision and robotics, with focuses on autonomous driving, geometry, self-supervised learning and event cameras.



Ziyun Wang received his BSc in Computer Science from Rice University, and his MSc in Robotics from the University of Pennsylvania, where he is currently pursuing a PhD in computer and information science. He is interested in computer vision and machine learning problems in robotics, focusing on event-based vision sensors.

Kaung Khant is pursuing a BSc in Computer Science and Economics at the University of Pennsylvania, and is expected to graduate in 2021.

Kostas Daniilidis Kostas Daniilidis is the Ruth Yalom Stone Professor of Computer and Information Science at the University of Pennsylvania where he has been faculty since 1998. He is an IEEE Fellow. He was the director of the interdisciplinary GRASP laboratory from 2008 to 2013, Associate Dean for Graduate Education from 2012-2016, and Director of Online Learning since 2016. He obtained his PhD in Computer Science from the University of Karlsruhe, 1992. His main interest today is in deep learning of 3D representations, data association, event-based cameras, semantic localization and mapping, and vision based manipulation.