EvDNeRF: Reconstructing Event Data with Dynamic Neural Radiance Fields

Anish Bhattacharya
University of Pennsylvania
Philadelphia, PA 19104
anish1@seas.upenn.edu

Ratnesh Madaan
Microsoft
Redmond, WA 98052
ratneshmadaan@gmail.com

Fernando Cladera
University of Pennsylvania
Philadelphia, PA 19104
fclad@seas.upenn.edu

Sai Vemprala
Scaled Foundations
Redmond, WA 98052
saihvemprala@gmail.com

Rogerio Bonatti
Microsoft
Redmond, WA 98052
rbonatti@microsoft.com

Kostas Daniilidis
University of Pennsylvania
Philadelphia, PA 19104
kostas@cis.upenn.edu

Ashish Kapoor
Scaled Foundations
Redmond, WA 98052
ashish@scaledfoundations.ai

Vijay Kumar
University of Pennsylvania
Philadelphia, PA 19104
kumar@seas.upenn.edu

Nikolai Matni
University of Pennsylvania
Philadelphia, PA 19104
nmatni@seas.upenn.edu

Jayesh K. Gupta
Poly Corporation
Bellevue, WA 98004
jayesh@withpoly.com

Abstract

We present EvDNeRF, a pipeline for generating event data and training an event-based dynamic NeRF, for the purpose of faithfully reconstructing eventstreams on scenes with rigid and non-rigid deformations that may be too fast to capture with a standard camera. Event cameras register asynchronous per-pixel brightness changes at MHz rates with high dynamic range, making them ideal for observing fast motion with almost no motion blur. Neural radiance fields (NeRFs) offer visual-quality geometric-based learnable rendering, but prior work with events has only considered reconstruction of static scenes. Our EvDNeRF can predict eventstreams of dynamic scenes from a static or moving viewpoint between any desired timestamps, thereby allowing it to be used as an event-based simulator for a given scene. We show that by training on varied batch sizes of events, we can improve test-time predictions of events at fine time resolutions, outperforming baselines that pair standard dynamic NeRFs with event generators. We release our simulated and real datasets, as well as code for multi-view event-based data generation and the training and evaluation of EvDNeRF models [1].

[1] https://github.com/anish-bhattacharya/EvDNeRF

1. Introduction

Simulation of 3D scenes based on discrete observations of 2D images is a challenging problem in computer graphics and robotics. Occlusions, shadows, imperfect state estimation, and perspective distortion make this a difficult task for classical graphics modeling methods. Recently, neural radiance fields (NeRF) [25] have emerged as a promising solution to these challenges. This method can take in a set of images from known camera viewpoints, and generate views of the reconstructed scene from novel viewpoints with high visual fidelity. This work has been extended to dynamic scenes with complex textures and structures [6, 23, 34, 44], which can be useful for planning and simulation. However, these methods work under the assumption of little to no motion blur and favorable lighting conditions, and fail in many dynamic scenes that break such constraints, as is often the case when generating 3D reconstructions of scenes in the wild. For example, it would be extremely difficult to create usable training images from a camera carried by a jogging human or a flying quadrotor, or of scenes with high-speed motion such as a projectile. This has motivated recent work training NeRF models from event cameras [16, 21, 39].

Event-based cameras, or neuromorphic cameras, are small and lightweight vision sensors containing pixels

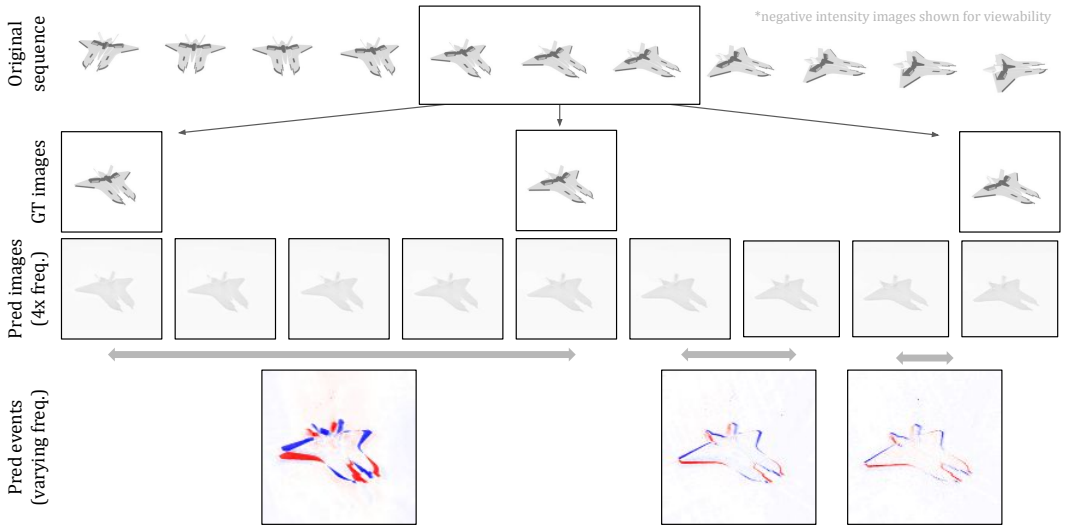


Figure 1. Evaluation of EvDNeRF on Jet-Down dataset during unseen camera motion around object. Ground truth images are sparse, but we can query the network for intensity images at a higher rate. We also vary the temporal resolution of predicted events, showing the original, 4x, and 8x resolution in the final row.

which fire independently at MHz frequencies. Each pixel detects changes in brightness, triggering a positive or negative "event" at that pixel location depending on a tunable event threshold. This asynchronous datastream of events (interchangeably events, event data, eventstream) is in stark contrast to the synchronous frame-based data from a 30-60Hz CMOS camera. While challenging to use for classical computer vision tasks, event cameras have now been used in the computer vision and robotics community to reconstruct images and video [28, 30, 31, 37, 38, 41, 42], perform state estimation [3, 5, 15, 20, 22, 26, 36, 47, 49], control robots [5, 7, 8, 19, 40, 45], and more [9]. These works use a variety of methods, from filtering to learning-based. Event cameras also have a high dynamic range, reaching 120dB, more than double that of a standard camera (event frequencies drop to 10-100kHz in low light). The detection of brightness changes naturally lends itself to motion detection and modeling without the challenge of parsing out static backgrounds.

Generating simulated events from a dynamic scene has been done by processing RGB video with either learned methods or an accurate model of an event camera sensor [4, 10, 11]. However, these methods cannot generalize to novel viewpoints or temporal/spatial resolutions. Models trained on NeRF backbones, however, can provide a strong geometric prior to make such generalizations possible. Initial work training NeRFs from events [16, 21, 39] considers static scenes, with single trajectory camera motion creating the eventstream. In contrast to these works, we train dynamic NeRFs on event data of moving scenes. We present results on both simulated and real data, with single objects exhibiting highly dynamic trajectories as well as non-rigid deformations, and scenes with multiple objects and occlusions. To the best of our knowledge, we are the first to train dynamic NeRF models on event data, which advances the state-of-the-art in generative events simulators, thereby enabling future event-based applications. **Our contributions are as follows:**

- EvDNeRF, the first dynamic NeRF trained from event data; we show generalization to event predictions at unseen, fine time resolutions and camera motion.
- A data generation procedure of multi-view eventstreams from rigid and non-rigid dynamic scenes in simulation and real-world.
- Demonstration and analysis of transfer learning an EvDNeRF to an unseen object.
- Open-source code and datasets to reproduce our results or train new models.

2. Related works

Modeling dynamic scenes. Modeling moving scenes from vision data alone is a challenging open task in computer vision. Since the problem is constrained only by 2D images collected over time, the task becomes more constrained as the amount of diverse data increases. The ability to train models in a self-supervised fashion with images was enabled by neural implicit rendering, starting with [25]. Extensions to dynamic scenes with rigid and non-rigid deformations largely have fallen into two categories, one training modular deformation or flow networks [6, 34, 44] and another directly training higher-dimensional architectures [23, 29, 32, 33] with more complex optimization landscapes.

Learning static NeRFs from event data. There has recently been some work on training traditional, static NeRF models from event camera data [16, 21, 39]. Rudnev, *et al*. (2023) does so with color event cameras to reconstruct objects in color and depth. Since the scene is static, the event camera is rotated around the object to generate the motion necessary for events. A combination of *positive* sampling of non-zero event pixels and *negative* sampling of background pixels makes for efficient NeRF training. Hwang, *et al*. (2023) uses a deadzone event reconstruction loss that has zero gradient between target event threshold discretizations. The positive and negative thresholds themselves are jointly learned, while regularizing distance from *a priori* known approximate values. They train on evenly-sized time windows of events and add small amounts of random noise during training to generate a model that is robust to event camera noise, which can occur in low-light conditions. Klenk, *et al*. (2023) similarly trains a NeRF from eventstreams, but employs additional loss terms (a) regularizing predictions in areas of extended-durations of no events, and (b) comparing predicted intensity images to ground truth blurred RGB frames of the scene to reconstruct colored images. These works attempt to generate high visual quality images of a static scene by training NeRFs on eventstreams and using an affine transformation on the resulting image predictions (or directly optimizing on blurred image frames) to correct the image brightness. In contrast, our work focuses on making accurate predictions of eventstreams themselves, on dynamic scenes with rigid and non-rigid deformations, with test-time generalization to queries of novel event batch size and viewing angle or camera motion.

Simulating event-based data. A small number of events simulators have been released that approach the problem of modeling this unique data in different ways. Some of these attempt to model the unique characteristics of the neuromorphic camera and its parameters to a high accuracy [4, 18]. This may help reduce the sim-to-real gap when transferring algorithms developed in simulation to real-world. Another work [27] utilizes a surface of active events [1] to calculate events from the high-fidelity Blender renderer. ESIM [35] (part of the Vid2E pipeline [11]) benefits from utilizing a deep-learning-based video upsampler [17] to generate continuous streams of events from sparse images. However, without proper upsampling this method struggles to simulate events at arbitrary intermediate timesteps, and still may contain patchy artifacts. Unlike our work, none of these simulators leverage geometric understanding of the scene, and therefore cannot simulate events from viewpoints outside the provided data.

3. Methods

We aim to build a dynamic neural radiance field from an eventstream, for predicting the expected events of a dynamic, bounded scene. To serve as an events simulator, we would like this model to be able to predict events on fine temporal resolutions and novel viewpoints outside the training data, from a variety of datasets with occlusions, shadows, scene dynamics, and multiple objects. The methods presented below leverage the asynchronous, discrete, very fast nature of event cameras to accomplish these goals better than traditional image-based simulators.

3.1. Problem formulation

A single event is represented as $e_k = (t_k, x_k, y_k, p_k)$, denoting a brightness change registered by an event camera at time t_k, pixel location (x_k, y_k) in the camera frame, with polarity $p_k \in \{-1, +1\}$. The polarity of an event indicates a positive or negative change in logarithmic illumination, quantized by positive and negative thresholds $C^{\pm}$. The change in brightness between two timesteps can be calculated from intensity images B_t.

$$\Delta L_{(t_k,x_k,y_k)} = L(t_k, x_k, y_k) - L(t_{k-1}, x_k, y_k) \tag{1}$$

$$e_p = \begin{cases} -1, & \text{if } \Delta L \leq C^- \\ +1, & \text{if } \Delta L \geq C^+ \end{cases} \tag{2}$$

$$\text{where} \quad L = \log(B). \tag{3}$$

Our pipeline approaches $\Delta\hat{L}_{t_k}$ estimation via Equation 1, by generating intensity image predictions $\hat{B}_t$ at two times t_k and t_{k-1} and using Equation 3. We can then determine the number $\hat{N}_{evs}$ of predicted events at a pixel location during that time window:

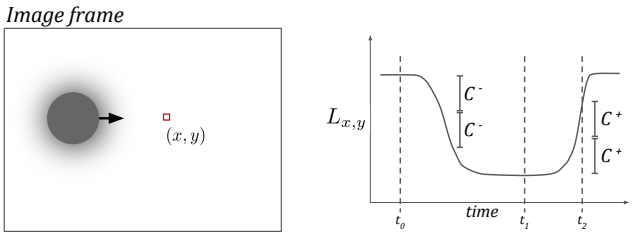
$$\hat{N}_{evs,(x_k,y_k)} = \begin{cases} \left\lfloor \frac{\Delta\hat{L}_{(x_k,y_k)}}{C^+} \right\rfloor, & \text{if } \Delta L \geq 0 \\ \left\lfloor \frac{\Delta\hat{L}_{(x_k,y_k)}}{|C^-|} \right\rfloor, & \text{if } \Delta L < 0. \end{cases} \tag{4}$$

Given a supervisory eventstream E_N and Equation 2, we can calculate each pixel's true logarithm brightness changes for a given time window $t_{k-1} < \tau < t_k$ batch of events (interchangeably event batch, sliced events, and so on) by a per-pixel summation:

$$\Delta L_{(t_k,x_k,y_k)} = \sum_{p\in\{-1,+1\}} C^p E_{\{t_{k-1}<\tau<t_k\},x_k,y_k,p} \tag{5}$$

Therefore, we can supervise EvDNeRF training between our predicted $\Delta\hat{L}_{t_k}$ and the true ΔL_{t_k} by Equation 5 over all image pixels.

An important note is that the calculated $\hat{N}_{evs}$ is a minimum count; as the time window $t_k - t_{k-1}$ increases, the predicted events form a discretized, aliased representation of the underlying smooth brightness changes occurring in the scene. For demonstration, Figure 2 shows how measurements between timesteps (t_0, t_1) would result in two negative events at the chosen pixel, but measurements between (t_0, t_2) would result in no events. Given this observation, we purposely vary the time window of batched events during the training of EvDNeRF, and notably find



(a) Illustration of an object with a smooth edge passing rightward over a pixel.

(b) Logarithm brightness change at pixel (x, y).

Figure 2. Demonstration of how time window discretizations can influence the count of events between timestep pairs. The time window (t_0, t_1) produces two negative events, whereas (t_0, t_2) produces no events.

that this improves our generation of finely-sliced events at test time. Note that while event cameras may not produce additional events at an event pixel during its refractory period, the scale of our time windows are generally an order of magnitude larger, so we disregard this parameter.

3.2. Event data generation

For both the simulation and real-world datasets we emulate a motion capture-like scenario of multiple event cameras capturing time synchronized datastreams from equally-spaced viewpoints. In simulation, we use an open-source simulator and renderer for data generation paired with an event generation package; in real-world we use one event camera and re-capture a repeatable scene motion at various viewing angles.

Simulation. We use the open source data generation package Kubric [14] to generate synthetic, photorealistic videos of dynamic objects using the renderer Blender. These scenes are bounded within a [4m, 4m, 4m] box with no background textures. The objects are pulled from the ShapeNet online dataset of textured meshes [2] and each motion trajectory (*e.g.* Down, Spiral, Land) is defined by setting the absolute pose of the object at discrete, sampled timesteps. 18 camera viewing angles (20° separation) around the scene capture time-synchronized images without blur or noise. These images are passed into the Vid2E package [11] which upsamples each sequence via Super-SloMo [17] and produces a continuous eventstream from each viewpoint using ESIM [35]. Figure 3a outlines this process. For quantitative validation, we generate eventstreams from 18 viewpoints offset from training viewpoints by 10°.

Real world. Since we require a high number of viewpoints of a dynamic scene but do not have access to a large array of timesynced event cameras, we create a highly repeatable scene motion. An example is seen in Figure 3b, where a knife rotates on a servo from an initial state to the final position seen in the image. The tripod-mounted servo is placed at 18 orientations relative to the event camera and the servo is triggered via an Arduino microcontroller. Eventstream timestamps are plotted in a histogram, where a sudden peak in events indicates the start of a scene motion; this serves as the synchronization timestamp between all viewpoints. Once all viewpoints are time-synchronized, the eventstreams (each a N×4 array) can be batched by arbitrary time windows for training EvDNeRF.

3.3. Training dynamic neural radiance fields from events

Dynamic NeRF model. We use DNeRF [34] as the backbone for EvDNeRF. This architecture incorporates a deformation network $\Psi_t(\mathbf{x}, t)$ that predicts the displacement of any 3D point in space $\mathbf{x}$ at time t relative to its position in some canonical scene, typically taken to be the scene reconstruction at $t = 0$. Predicted displacements of each query point are used to feed deformation-compensated points into the canonical NeRF model $\Psi_x(\mathbf{x}, (\theta, \phi))$ which predicts color and density at these points, dependent on viewing angle (θ, ϕ), to be fed into a backwards differentiable rendering function. We assume our dynamic scene is contained within a 3D origin-centered box of size $[b, b, b]$, and that we only deal with intensity (*i.e.* grayscale) images since most event cameras are not color (thereby making $\mathbf{c}$ a scalar, though we denote it as a vector for generality).

$$\Psi_t(\mathbf{x}, t) = \Delta\mathbf{x} \text{ , where } \mathbf{x} \in \mathbb{R}^3 \text{ and } \frac{-b}{2} < \mathbf{x}_i < \frac{b}{2}.$$

$$\Psi_x\left(\mathbf{x}, (\theta, \phi)\right) = (\mathbf{c}, \sigma)$$

The NeRF network Ψ_x [25] learns volumetric density by simulating the geometry of a renderer and casting rays from multiple annotated viewpoints. Both Ψ_t and Ψ_x, composed of fully-connected feedforward networks (*i.e.* multilayer perceptrons, or MLPs), are able to learn continuous deformation and density models in 3D space, respectively, by being paired with differentiable rendering [24] and positional encoding of query points prior to input into the neural network [43].

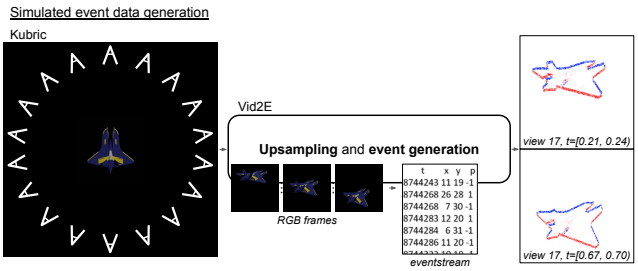
The rendering function accumulates S colors and densities at points $\mathbf{x}_S$ along a ray $\mathbf{r}$ to obtain a pixel's color $C(\mathbf{r})$:

$$C(\mathbf{r}) = \sum_{i}^{S} T_i \alpha_i \mathbf{c}_i$$

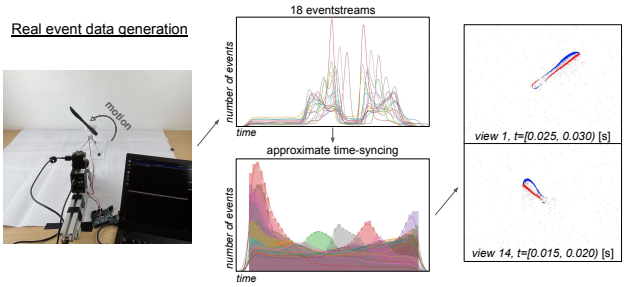
$$\text{where } T_i = \exp\left(-\sum_{j=1}^{i-1} \sigma_j \|\mathbf{x_{j+1}} - \mathbf{x_j}\|\right),$$

$$\alpha_i = 1 - \exp(-\sigma_i \|\mathbf{x_{i+1}} - \mathbf{x_i}\|).$$

The ray $\mathbf{r}$ is a unit vector with origin at camera origin and direction through a pixel center. The transmittance for a point along the ray, T_i, is determined by accumulated densities from the camera origin until that point. The alpha value α_i is calculated at each point only relative to the distance to a



(a) Generating simulated eventstreams from rendered images. We take blurred grayscale images and eventstreams as the data available to models for training.



(b) Real eventstreams from an event camera from multiple viewpoints. We take the eventstreams as the data available to models for training.

Figure 3. Event data generation for simulated and real sequences. Note that in simulation, timestamps are unitless and range from [0, 1.0], whereas in real-world, timestamps are in seconds and range from [0, 0.5][s].

neighboring sample point.

Using this rendering function and the DNeRF backbone, as we described in Section 3.1, we predict intensity images at two time instances and calculate the continuous-valued log brightness change $\Delta\hat{L}_{t_k}$ (Equation 1), which can then be compared to the ground truth discrete-valued ΔL_{t_k} (Equation 5). During training, we sample 50% of rays positively from event pixels and 50% randomly.

Varied batching of events. We vary the batch size of events during training to enable EvDNeRF to make fine temporal predictions at test time, as motivated in Section 3.1. We start training with batches of events equal in time window to the framerate of the image stream (*e.g.* 32 frames for simulated scenes), and halve the batch size multiple times during training, thereby repeatedly doubling the temporal resolution of our supervisory signal.

Loss function. An event reconstruction loss term $\mathcal{L}_{ev}$ is formulated as a piecewise loss function calculated at each pixel location to accommodate the discrete nature of threshold-based events.

$$\mathcal{L}_{ev,\,xy} = \begin{cases} 0, \text{ if } \Delta L_{xy} \leq \Delta\hat{L}_{xy} < \Delta L_{xy} + C^{\pm} \\ \left\| \Delta\hat{L}_{xy} - \left(\Delta L_{xy} + \frac{C^{\pm}}{2} \right) \right\|_2^2, \text{ otherwise.} \end{cases}$$

The positive or negative threshold $C^{\pm}$ is determined by the polarity of the ground truth value ΔL_{xy}. Within the $C^{\pm}$-width bin of any particular discretized, ground truth, log-brightness-change ΔL_{xy}, the loss of prediction $\Delta\hat{L}_{xy}$ is valued 0. However, outside this bin, the loss is valued as a squared distance to the center of that bin.

4. Results

We present events reconstruction images from EvDNeRF across various test-time queries. Each events reconstruction is a pseudoframe representing the collection of events predicted between two query timestamps, coalesced into an image where the intensity of pixel color represents the number of events at a pixel location (red indicates positive events, blue indicates negative events). Equations 1 and 4 describe the event counting process from network predictions. We show various events reconstructions for training samples, very fine untrained time slices, unseen viewpoints, and nonzero camera motion, with comparison to ground truth samples when available. While the problem of reconstructing events from a geometric model of a dynamic scene does not have other works with which to make direct comparison, we form baselines to be competitive with our model across certain metrics. We found EvDNeRF to be more unstable to train than DNeRF, likely due to the task of training from the finite-difference-like event data, and that it does not easily reconstruct visual-quality images or depth maps. Therefore, while the most simple baseline model is formed from training DNeRF on the available motion-blurred images (*DNeRF*), we also attempt to utilize DNeRF to possibly "smoothen" out the results of EvDNeRF for improved events reconstruction; to this end, *EvDNeRF+DNeRF* uses image outputs from EvDNeRF at 100k iterations of training for another 100k iterations of DNeRF training. We also use E2VID to train a DNeRF model from images reconstructed from the eventstream (*E2VID+DNeRF*). These methods are illustrated in Figure 4, with examples of training input types, training midpoint data types, and reconstructed validation outputs. Finally, as we aim to reconstruct the eventstream itself, we also tested using VID2E on the output of *E2VID+DNeRF* to reconstruct high-quality eventstreams; however, this returned high levels of background events, and therefore is not pictured here (details in the supplementary).

4.1. Events reconstruction

Simulated Jet scenes. We evaluate our method compared to baselines on simulated scenes with the Jet object and three trajectories; quantitative metrics are shown in Table 1, where EvDNeRF outperforms all baselines on events reconstruction PSNR (peak signal-to-noise ratio) and LPIPS (learned perceptual image patch similarity) [46], and either outperforms or is competitive for SSIM (structural similarity) and MAE (mean absolute error) measures. Note that these metrics are calculated directly on the $C^{\pm}$-quantized

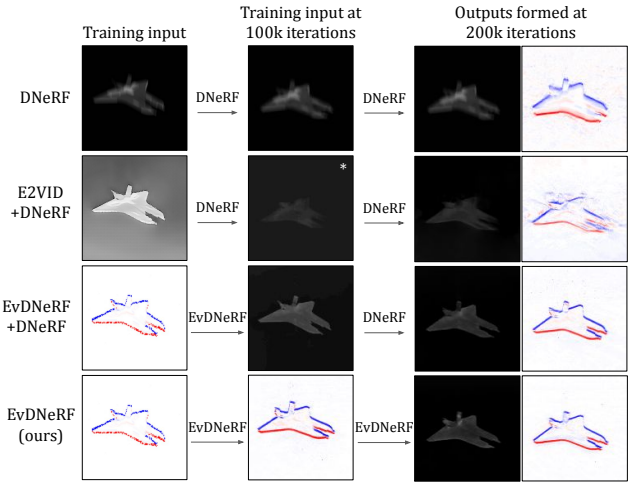


Figure 4. Baselines and EvDNeRF training explained, by visualizing sample data at the first training step, training midpoint, and end of training (output). $\xrightarrow{\text{DNeRF}}$ and $\xrightarrow{\text{EvDNeRF}}$ indicate 100k training steps of either method; EvDNeRF trains directly from event batches while DNeRF trains from images (baseline method *DNeRF* trains from blurred images).
(* contrast improved for viewability.)

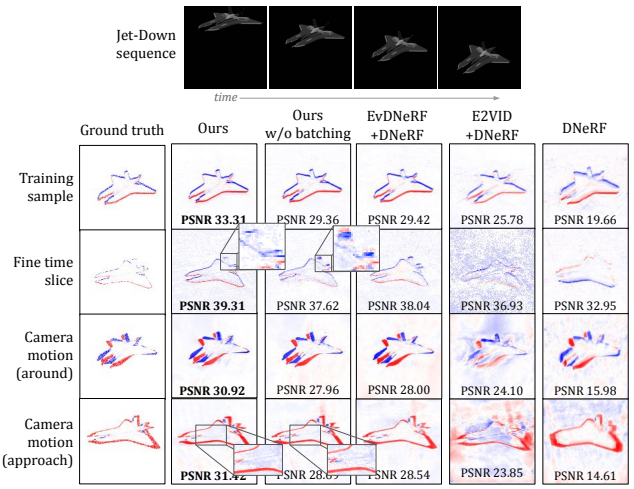


Figure 5. Comparisons of predicted event images between our method, an ablation without our varied events batching, and baselines, for various test-time queries including unseen temporal resolutions and camera motions. Our method constructs events with the best temporal resolution, as seen in the images as sharp object "edges" and fine details. Our reconstructions also do not contain as much spurious background predictions as DNeRF-based methods. Note that since we compare event frames with the un-normalized $\Delta\hat{L}$ values, the PSNR values should only be considered a relative metric, rather than an absolute metric as is commonly used for uint8-valued images.

values of $\Delta\hat{L}$ when possible; for example, the data range for PSNR is calculated as the range of ground truth ΔL_{t_k} calculated over all pixels from Equation 5. Figure 6 demonstrates that EvDNeRF quickly approaches very high PSNR (34) for event reconstruction during training, while baseline methods cannot surpass a ceiling of 27.

Variations of unseen time windows and camera motions, as well as a training sample for comparison, can be seen in Figure 5. In addition to the baselines mentioned above, we include an ablation of our method without varied batching of events during training. Across all samples, EvDNeRF with varied event batching (*Ours*) has the highest PSNR on event images. The first notable improvements EvDNeRF makes over DNeRF methods is apparent in the training sample reconstruction (first row), where EvDNeRF both (a) has a sharper temporal distribution of events (represented by a sharper object), and (b) has fewer background event predictions due to the positive sampling methodology of events rather than randomly sampling rays in the image. When testing on a very fine time window (second row), the ground truth sample appears patchy along the edges of the jet; this is an artifact of the VID2E procedure for generating events. EvDNeRF, however, provides a smooth and consistent eventstream in the same areas, further demonstrating an edge of our method over existing simulators. Direct comparison to the non-batching ablation also shows that our method better reconstructs fine details of the jet fins (shown in cutouts).

Though training data only contains stationary camera viewpoints, the bottom two rows query events during camera motion. In these examples, we again see that *E2VID+DNeRF* and *DNeRF*, which both only use DNeRF to train, have more severe, spurious background predictions. While our method boasts the highest PSNR values, visually some of the methods' reconstructions appear similar. However, the bottom cutouts (with slightly improved contrast) show that our method reconstructs fine events on the interior of the jet–the two faint lines running along the length of the jet. These are less clear in the other methods. More variety of camera viewpoints are shown in the supplementary.

Note that since *EvDNeRF+DNeRF* outperforms *Ours w/o batching* in terms of PSNR, it might suggest we should apply batching to *EvDNeRF+DNeRF* for fair comparison to EvDNeRF. However, temporal batching is not applicable to the highly discretized images, which arrive at a fixed frequency (32Hz); batching can only be done with events since the datastream can be considered near-continuous (MHz+).

Non-rigid scene deformations. We present events reconstruction with EvDNeRF on two simulated scenes with non-rigid deformations, or scenes where viewpoint change cannot emulate scene dynamics (Figure 7). Performance is comparable to the rigid deformation case of the Jet datasets. The Multi dataset features lots of occlusions and shadows; occlusions can challenge the deformation network Ψ_t of the DNeRF backbone during training, but we found it to work well. The Lego tractor scene has a large stationary cabin,

Simulated data event image reconstruction																
	Jet-Down				Jet-Spiral				Jet-Land				Multi			
	PSNR↑	SSIM↑	LPIPS↓	MAE↓	PSNR↑	SSIM↑	LPIPS↓	MAE↓	PSNR↑	SSIM↑	LPIPS↓	MAE↓	PSNR↑	SSIM↑	LPIPS↓	MAE↓
DNeRF	26.11	0.800	0.325	0.568	25.83	0.786	0.527	0.620	30.13	0.868	0.329	0.329	25.57	0.499	0.662	1.012
E2VID+DNeRF	26.97	0.820	0.237	0.448	26.82	0.848	0.354	0.446	30.82	**0.871**	0.223	**0.258**	26.83	0.821	0.377	0.427
EvDNeRF+DNeRF	32.54	0.877	0.140	0.364	31.08	**0.878**	0.153	**0.384**	33.08	0.851	0.176	0.338	27.45	0.841	0.154	0.374
EvDNeRF (ours)	**33.39**	**0.891**	**0.117**	**0.336**	**32.57**	0.848	**0.145**	0.392	**34.56**	0.841	**0.149**	0.332	**27.50**	**0.843**	**0.079**	**0.366**

Table 1. Quantitative metrics comparing EvDNeRF to baselines for a simulated scene of the Jet object on three different trajectories, as well as the Multi multiple object dataset. EvDNeRF outperforms the baselines for most datasets across various metrics.

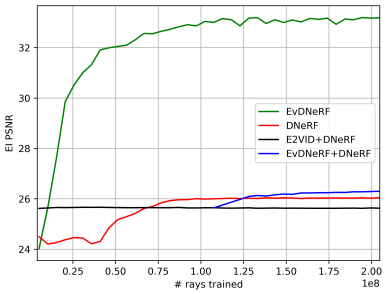


Figure 6. PSNR of event image (EI) reconstructions vs. number of rays trained, showing that *EvDNeRF* widely outperforms baselines on the Jet-Down dataset. *EvDNeRF+DNeRF* tracks *EvDNeRF* exactly until the halfway point, where switching to optimizing intensity images causes a drop in EI PSNR.

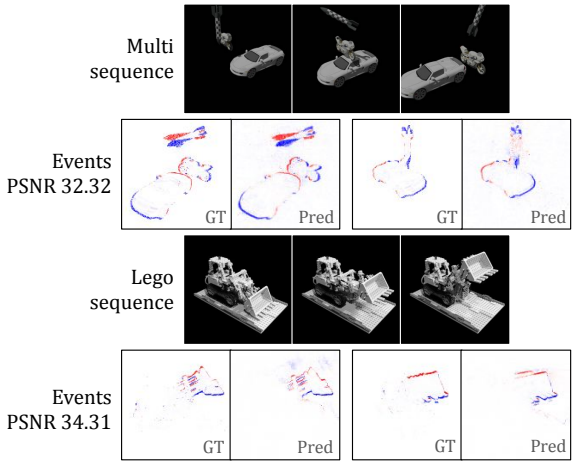
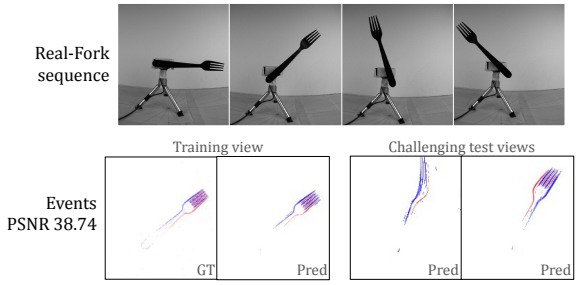


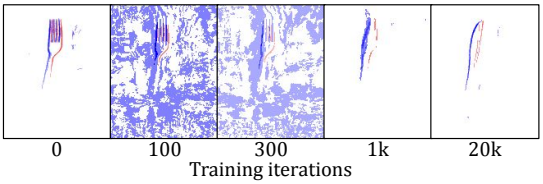
Figure 7. Reconstructions of events by EvDNeRF on simulated scenes with non-rigid deformations, at test viewpoints. Predictions (*Pred*) are very similar to ground truth (*GT*) and return high PSNR.

with only the bucket being raised mechanically by the lego components; this is why only the bucket is seen in the events reconstruction, and image and depth reconstructions of the scene also primarily show the bucket (see supplementary).

Real world. For demonstration on real-world data, we use a fork rotating on a servo as described in Section 3.2; this simple object was chosen due to its broad structure and fine features. The Real-Fork dataset eventstream contains background and residual events that are characteristic of event camera "noise". We believe this causes many low-valued,



(a) Images of scene motion and events reconstructions.



(b) Visualizing the training transfer from a fork to a knife.

Figure 8. Various results on Real-Fork. (a) shows a single training sample for comparison to ground truth and PSNR, as well as two challenging test samples. See the supplementary for full trajectory reconstructions. (b) shows the training progression of transferring an EvDNeRF model to a new object.

dispersed background events, or "floating" events, in the reconstructions. These are very easily filtered out with a low threshold, which we do for the event images we present in this paper in Figure 8 (non-filtered images can be found in the supplementary). Reconstructions in Figure 8a are very good even on challenging test viewpoints such as the nearly head-on view of the fork approaching the camera (first *Test views* image).

4.2. Transferring to a new object

NeRF models are generally considered non-transferable to novel objects or scenes since they are self-supervised to learn the geometry of the scene on which it was trained. However, we explore the possibility of transferring an EvDNeRF trained on real-world data of one object to another with the same trajectory. The objects here are of roughly the same size, simplifying the transfer learning process. Figure 8b demonstrates transfer progress by showing a validation viewpoint at a few snapshots in training. Note that while training on the fork from scratch took 200k iterations, the transfer to the knife took less than 20k iterations. Since training EvDNeRF from events is more unstable than train-

ing a dynamic NeRF from images, such a transfer might be very useful to reduce train time on new, but similar, scenes.

4.3. Limitations and extensions

We find that number of training views has a big impact on events reconstruction to constrain both the spatial and temporal dimensions of dynamic scenes. Figure 9 shows the events reconstruction PSNR at an intermediate point in training as number of training views increases. We show this for the simulated Jet-Down dataset and the Real-Fork dataset. As seen in the Jet-Down reconstructions on the right of the figure, captured from the * points on the green curve, 9 training views correctly positions the object in the center of the frame, whereas fewer views does not constrain the position of the object in space. For Real-Fork, the object has reduced spread across its trajectory as number of views increases, indicating improved inference of the temporal dynamics of the scene. These results show that many views are needed for training an accurate EvDNeRF, as enabled by our simulated and real data generation pipelines.

While this work focuses on developing an eventstream simulator that can generalize to novel views and temporal resolutions using a NeRF backbone, the intensity image and depth map reconstruction from EvDNeRF can be poor. Intensity images can be affine-shifted in log brightness to match that of ground truth images, but the visual quality of these images is not as good as those seen from event-trained static-NeRF works (see Section 2). This may be because the DNeRF backbone attempts to separate the temporal and spatial dimensions via two MLPs and only uses reconstruction loss terms, a difficult task made more challenging by training directly from events. It is possible that utilizing additional consistency loss terms to supervise flow [6, 23] or combining with flow estimation from events [12, 13, 48] might improve our results. We may also use a different backbone that jointly learns the time-spatial relationship within one architecture [23, 33]. Further work will be necessary to determine whether these improve intensity image and depth reconstruction when training directly from event data.

5. Conclusion

We present EvDNeRF, the first work addressing eventstream reconstruction across novel temporal resolutions and viewpoints of a dynamic scene trained solely from events, by leveraging the strong multi-view consistency properties of neural radiance fields. We thoroughly assess the performance of our method, compared to baselines designed to address possible limitations of EvDNeRF, across various scenes including rigid and non-rigid deformations, in simulation and in real-world. One limitation is the instability of training from events, likely due to the fact that events capture differences in scene appearance rather

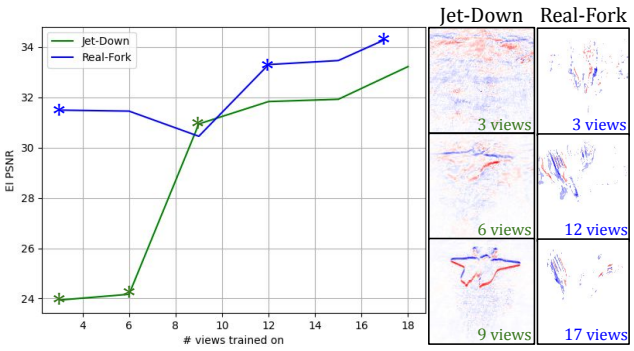


Figure 9. Events reconstruction PSNR as a function of the number of views trained on, for a simulated and real dataset after partial training of EvDNeRF. Validation reconstructions are extracted at * points and shown on the right. For Jet-Down, 9 views are necessary to correctly place the object's events reconstructions in space. For Real-Fork, more views appears to better constrain reconstructions in the temporal dimension.

than the scene itself; however, our experiment on transferring EvDNeRF models to new scenes suggests that training on new, similar scenes can be done quickly, reducing training time and likelihood of divergence.

Since EvDNeRF can be used as an events simulator of a given scene, where novel event batches unseen in the given dataset can be queried, this work might enable future robotics and computer vision tasks or research where in-situ event data may not be easily acquired. Future work leveraging NeRFs for simulating eventstreams may apply more recent advances in dynamic NeRFs to events, such as unbounded scenes and sparse-view training; jointly utilizing blurred image data with eventstreams may improve image and depth reconstruction; other work may learn the controllable dynamics or object properties in a dynamic scene.

Acknowledgements

We would like to thank all the members of the Autonomous Systems Research group at Microsoft Research for their support and discussions; Anthony Bisulco for guidance on collecting real eventstream data; Bernd Pfrommer for his excellent work on drivers and software supporting event camera research; Jiahui Lei for NeRF-related suggestions and discussions; Kenneth Chaney for his support on using event cameras and related software. This work was supported by National Science Foundation, grant no. DGE-2236662.

References

[1] Ryad Benosman, Charles Clercq, Xavier Lagorce, Sio-Hoi Ieng, and Chiara Bartolozzi. Event-based visual flow. *IEEE transactions on neural networks and learning systems*, 25(2):407–417, 2013. 3

[2] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015. 4

[3] Peiyu Chen, Weipeng Guan, and Peng Lu. ESVIO: Event-based stereo visual inertial odometry. *IEEE Robotics and Automation Letters*, 2023. 2

[4] Tobi Delbruck, Yuhuang Hu, and Zhe He. V2e: From video frames to realistic dvs event camera streams. *arXiv e-prints*, pages arXiv–2006, 2020. 2, 3

[5] Rika Sugimoto Dimitrova, Mathias Gehrig, Dario Brescianini, and Davide Scaramuzza. Towards low-latency high-bandwidth control of quadrotors using event cameras. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4294–4300. IEEE, 2020. 2

[6] Yilun Du, Yinan Zhang, Hong-Xing Yu, Joshua B Tenenbaum, and Jiajun Wu. Neural radiance flow for 4d view synthesis and video processing. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 14304–14314. IEEE Computer Society, 2021. 1, 2, 8

[7] A Gómez Eguíluz, Juan Pablo Rodríguez-Gómez, R Tapia, Francisco Javier Maldonado, José Ángel Acosta, JR Martínez-de Dios, and Aníbal Ollero. Why fly blind? event-based visual guidance for ornithopter robot flight. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1958–1965. IEEE, 2021. 2

[8] Davide Falanga, Kevin Kleber, and Davide Scaramuzza. Dynamic obstacle avoidance for quadrotors with event cameras. *Science Robotics*, 5(40):eaaz9712, 2020. 2

[9] Guillermo Gallego, Tobi Delbrück, Garrick Orchard, Chiara Bartolozzi, Brian Taba, Andrea Censi, Stefan Leutenegger, Andrew J Davison, Jörg Conradt, Kostas Daniilidis, et al. Event-based vision: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 44(1):154–180, 2020. 2

[10] Garibaldi Pineda García, Patrick Camilleri, Qian Liu, and Steve Furber. pydvs: An extensible, real-time dynamic vision sensor emulator using off-the-shelf hardware. In *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–7. IEEE, 2016. 2

[11] Daniel Gehrig, Mathias Gehrig, Javier Hidalgo-Carrió, and Davide Scaramuzza. Video to events: Recycling video datasets for event cameras. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3586–3595, 2020. 2, 3, 4

[12] Mathias Gehrig, Mario Millhäusler, Daniel Gehrig, and Davide Scaramuzza. E-raft: Dense optical flow from event cameras. In *2021 International Conference on 3D Vision (3DV)*, pages 197–206. IEEE, 2021. 8

[13] Mathias Gehrig, Manasi Muglikar, and Davide Scaramuzza. Dense continuous-time optical flow from events and frames. *arXiv preprint arXiv:2203.13674*, 2022. 8

[14] Klaus Greff, Francois Belletti, Lucas Beyer, Carl Doersch, Yilun Du, Daniel Duckworth, David J Fleet, Dan Gnanapragasam, Florian Golemo, Charles Herrmann, Thomas Kipf, Abhijit Kundu, Dmitry Lagun, Issam Laradji, Hsueh-Ti (Derek) Liu, Henning Meyer, Yishu Miao, Derek Nowrouzezahrai, Cengiz Oztireli, Etienne Pot, Noha Radwan, Daniel Rebain, Sara Sabour, Mehdi S. M. Sajjadi, Matan Sela, Vincent Sitzmann, Austin Stone, Deqing Sun, Suhani Vora, Ziyu Wang, Tianhao Wu, Kwang Moo Yi, Fangcheng Zhong, and Andrea Tagliasacchi. Kubric: a scalable dataset generator. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 4

[15] Antea Hadviger, Igor Cvišić, Ivan Marković, Sacha Vražić, and Ivan Petrović. Feature-based event stereo visual odometry. In *2021 European Conference on Mobile Robots (ECMR)*, pages 1–6. IEEE, 2021. 2

[16] Inwoo Hwang, Junho Kim, and Young Min Kim. Ev-nerf: Event based neural radiance field. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 837–847, 2023. 1, 2, 3

[17] Huaizu Jiang, Deqing Sun, Varun Jampani, Ming-Hsuan Yang, Erik Learned-Miller, and Jan 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*, pages 9000–9008, 2018. 3, 4

[18] Damien Joubert, Alexandre Marcireau, Nic Ralph, Andrew Jolley, André van Schaik, and Gregory Cohen. Event camera simulator improvements via characterized parameters. *Frontiers in Neuroscience*, 15:702765, 2021. 3

[19] Jacques Kaiser, J Camilo Vasquez Tieck, Christian Hubschneider, Peter Wolf, Michael Weber, Michael Hoff, Alexander Friedrich, Konrad Wojtasik, Arne Roennau, Ralf 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)*, pages 127–134. IEEE, 2016. 2

[20] Hanme Kim, Stefan Leutenegger, and Andrew J Davison. Real-time 3d reconstruction and 6-dof tracking with an event camera. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part VI 14*, pages 349–364. Springer, 2016. 2

[21] Simon Klenk, Lukas Koestler, Davide Scaramuzza, and Daniel Cremers. E-nerf: Neural radiance fields from a moving event camera. *IEEE Robotics and Automation Letters*, 8(3):1587–1594, 2023. 1, 2, 3

[22] Beat Kueng, Elias Mueggler, Guillermo Gallego, and Davide Scaramuzza. Low-latency visual odometry using event-based feature tracks. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 16–23. IEEE, 2016. 2

[23] Zhengqi Li, Simon Niklaus, Noah Snavely, and Oliver Wang. Neural scene flow fields for space-time view synthesis of dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6498–6508, 2021. 1, 2, 8

[24] Nelson Max. Optical models for direct volume rendering. *IEEE Transactions on Visualization and Computer Graphics*, 1(2):99–108, 1995. 4

[25] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf:

Representing scenes as neural radiance fields for view synthesis. In *European Conference on Computer Vision (ECCV)*, pages 405–421. Springer, 2020. 1, 2, 4

[26] Elias Mueggler, Guillermo Gallego, Henri Rebecq, and Davide Scaramuzza. Continuous-time visual-inertial odometry for event cameras. *IEEE Transactions on Robotics*, 34(6):1425–1440, 2018. 2

[27] Elias Mueggler, Henri Rebecq, Guillermo Gallego, Tobi Delbruck, and Davide Scaramuzza. The event-camera dataset and simulator: Event-based data for pose estimation, visual odometry, and slam. *The International Journal of Robotics Research*, 36(2):142–149, 2017. 3

[28] Gottfried Munda, Christian Reinbacher, and Thomas Pock. Real-time intensity-image reconstruction for event cameras using manifold regularisation. *International Journal of Computer Vision*, 126:1381–1393, 2018. 2

[29] Julian Ost, Fahim Mannan, Nils Thuerey, Julian Knodt, and Felix Heide. Neural scene graphs for dynamic scenes. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2856–2865, 2021. 2

[30] Liyuan Pan, Richard Hartley, Cedric Scheerlinck, Miaomiao Liu, Xin Yu, and Yuchao Dai. High frame rate video reconstruction based on an event camera. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(5):2519–2533, 2020. 2

[31] Liyuan Pan, Cedric Scheerlinck, Xin Yu, Richard Hartley, Miaomiao Liu, and Yuchao Dai. Bringing a blurry frame alive at high frame-rate with an event camera. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6820–6829, 2019. 2

[32] Keunhong Park, Utkarsh Sinha, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Steven M Seitz, and Ricardo Martin-Brualla. Nerfies: Deformable neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5865–5874, 2021. 2

[33] Keunhong Park, Utkarsh Sinha, Peter Hedman, Jonathan T Barron, Sofien Bouaziz, Dan B Goldman, Ricardo Martin-Brualla, and Steven M Seitz. Hypernerf: A higher-dimensional representation for topologically varying neural radiance fields. *arXiv preprint arXiv:2106.13228*, 2021. 2, 8

[34] Albert Pumarola, Enric Corona, Gerard Pons-Moll, and Francesc Moreno-Noguer. D-NeRF: Neural radiance fields for dynamic scenes. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10318–10327, 2021. 1, 2, 4

[35] Henri Rebecq, Daniel Gehrig, and Davide Scaramuzza. Esim: an open event camera simulator. In *Conference on robot learning*, pages 969–982. PMLR, 2018. 3, 4

[36] Henri Rebecq, Timo Horstschäfer, Guillermo Gallego, and Davide Scaramuzza. Evo: A geometric approach to event-based 6-dof parallel tracking and mapping in real time. *IEEE Robotics and Automation Letters*, 2(2):593–600, 2016. 2

[37] Henri Rebecq, René Ranftl, Vladlen Koltun, and Davide Scaramuzza. Events-to-video: Bringing modern computer vision to event cameras. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3857–3866, 2019. 2

[38] Henri Rebecq, René Ranftl, Vladlen Koltun, and Davide Scaramuzza. High speed and high dynamic range video with an event camera. *IEEE transactions on pattern analysis and machine intelligence*, 43(6):1964–1980, 2019. 2

[39] Viktor Rudnev, Mohamed Elgharib, Christian Theobalt, and Vladislav Golyanik. Eventnerf: Neural radiance fields from a single colour event camera. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4992–5002, 2023. 1, 2, 3

[40] Nitin J Sanket, Chethan M Parameshwara, Chahat Deep Singh, Ashwin V Kuruttukulam, Cornelia Fermüller, Davide Scaramuzza, and Yiannis Aloimonos. Evdodgenet: Deep dynamic obstacle dodging with event cameras. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10651–10657. IEEE, 2020. 2

[41] Cedric Scheerlinck, Nick Barnes, and Robert Mahony. Continuous-time intensity estimation using event cameras. In *Asian Conference on Computer Vision*, pages 308–324. Springer, 2018. 2

[42] Cedric Scheerlinck, Henri Rebecq, Daniel Gehrig, Nick Barnes, Robert Mahony, and Davide Scaramuzza. Fast image reconstruction with an event camera. In *IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, pages 156–163, 2020. 2

[43] Matthew Tancik, Pratul Srinivasan, Ben Mildenhall, Sara Fridovich-Keil, Nithin Raghavan, Utkarsh Singhal, Ravi Ramamoorthi, Jonathan Barron, and Ren Ng. Fourier features let networks learn high frequency functions in low dimensional domains. *Advances in Neural Information Processing Systems*, 33:7537–7547, 2020. 4

[44] Edgar Tretschk, Ayush Tewari, Vladislav Golyanik, Michael Zollhöfer, Christoph Lassner, and Christian Theobalt. Non-rigid neural radiance fields: Reconstruction and novel view synthesis of a dynamic scene from monocular video. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12959–12970, 2021. 1, 2

[45] Ziyun Wang, Fernando Cladera Ojeda, Anthony Bisulco, Daewon Lee, Camillo J Taylor, Kostas Daniilidis, M Ani Hsieh, Daniel D Lee, and Volkan Isler. Ev-catcher: High-speed object catching using low-latency event-based neural networks. *IEEE Robotics and Automation Letters*, 7(4):8737–8744, 2022. 2

[46] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 586–595, 2018. 5

[47] Yi Zhou, Guillermo Gallego, and Shaojie Shen. Event-based stereo visual odometry. *IEEE Transactions on Robotics*, 37(5):1433–1450, 2021. 2

[48] Alex Zihao Zhu, Liangzhe Yuan, Kenneth Chaney, and Kostas Daniilidis. Ev-flownet: Self-supervised optical flow estimation for event-based cameras. *arXiv preprint arXiv:1802.06898*, 2018. 8

[49] Yi-Fan Zuo, Jiaqi Yang, Jiaben Chen, Xia Wang, Yifu Wang, and Laurent Kneip. Devo: Depth-event camera visual odometry in challenging conditions. In *2022 International Conference on Robotics and Automation (ICRA)*, pages 2179–2185. IEEE, 2022. 2