E-NeRF: Neural Radiance Fields from a Moving Event Camera

Simon Klenk[1], Lukas Koestler[1], Davide Scaramuzza[2], Daniel Cremers[1]

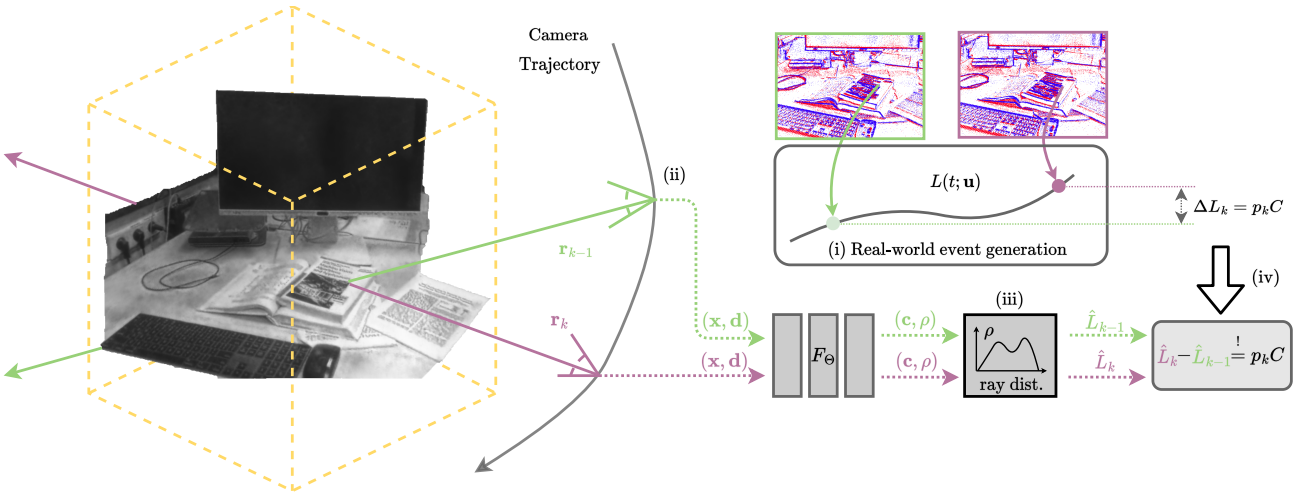


Fig. 1: Overview of the proposed E-NeRF. (i) For a fixed pixel $\mathbf{u} \in \mathbb{R}^2$ the real-world logarithmic brightness $L(t;\mathbf{u})$ varies over time. If $L(t;\mathbf{u})$ crosses a threshold C wrt. to the memory value a new event is triggered (cf. Equation 1). (ii) The proposed E-NeRF reconstructs a radiance field from a moving event camera. We evaluate the NeRF-MLP F_Θ at the two rays $\mathbf{r}_{k-1}$ and $\mathbf{r}_k$ corresponding to the two last events e_{k-1} and e_k for pixel $\mathbf{u}$ to obtain color and density $(\mathbf{c}, \rho)$. (iii) The color and density for many points along each ray are accumulated into the final estimated pixel brightness $\hat{I}_{k-1}$ and $\hat{I}_k$ using volume rendering. We convert the brightness with the linlog mapping in Equation 2 to $\hat{L}_{k-1}$ and $\hat{L}_k$. (iv) The event constraint induces the loss function on the two rendered logarithmic brightness values, see Equation 3 (and Equation 4 for unknown threshold C).

***Abstract*—Estimating neural radiance fields (NeRFs) from "ideal" images has been extensively studied in the computer vision community. Most approaches assume optimal illumination and slow camera motion. These assumptions are often violated in robotic applications, where images may contain motion blur, and the scene may not have suitable illumination. This can cause significant problems for downstream tasks such as navigation, inspection, or visualization of the scene. To alleviate these problems, we present E-NeRF, the first method which estimates a volumetric scene representation in the form of a NeRF from a fast-moving event camera. Our method can recover NeRFs during very fast motion and in high-dynamic-range conditions where frame-based approaches fail. We show that rendering high-quality frames is possible by only providing an event stream as input. Furthermore, by combining events and frames, we can estimate NeRFs of higher quality than state-of-the-art approaches under severe motion blur. We also show that combining events and frames can overcome failure cases of NeRF estimation in scenarios where only a few input views are available without requiring additional regularization.**

***Index Terms*—Mapping, Deep Learning Methods, Event Cameras**

Code and simulated datasets can be found under https://github.com/knelk/enerf.

I. Introduction

CAMERAS are frequently used in robotic perception for tasks such as localization, path planning, scene understanding, and inspection. Furthermore, cameras are ubiquitous in virtual and augmented reality systems, where they are used for ego localization, spatial reasoning, and visualizations. These systems require compact and high-quality visual representations of the underlying three-dimensional geometry.

Recently, there has been a trend in the computer vision community to study neural radiance fields (NeRFs) as a solution to scene representation and novel view synthesis. NeRFs represent the scene by a multilayer perceptron (MLP) combined with differentiable rendering [1]. They allow for novel view synthesis at unseen viewpoints. Up to this date,

Manuscript received August 19, 2022; revised December 6, 2022; Accepted: January 13, 2023.

This paper was recommended for publication by Editor Javier Civera upon evaluation of the Associate Editor and Reviewers' comments.

This work was supported by the European Research Council (ERC) under Grant Agreements 884679 (SIMULACRON) and 864042 (AGILEFLIGHT), the Munich Center for Machine Learning, the BMBF Project BBKI-Chips, the Swiss National Science Foundation (SNSF) through the National Centre of Competence in Research (NCCR) Robotics.

[1]The first, second, and fourth authors are with Computer Vision Group, Technical University of Munich, Germany. [2]The third author is with the Robotics and Perception Group, University of Zurich, Switzerland.

Digital Object Identifier (DOI): see top of this page.

NeRFs have mainly been studied on simulated data as well as on high-quality, real-world images gathered under ideal conditions [2], [3], which ensure good surrounding light and minimal noise, apart from a few exceptions such as [4], [6].

One common problem of real-world images is motion blur, which is caused by fast motion and high exposure times (usually required in low-light scenes). During the exposure time interval, each pixel of the moving camera integrates light from different points in the scene, resulting in a mixture of color values. Hence, motion blur is essentially a loss of information that can cause robotic navigation pipelines to fail [8], [7] and severely limit downstream processing, e.g., inspection tasks. Simply reducing the exposure time is often not feasible since this also reduces the amount of light received on the sensor and increases the amount of noise. Another problem of real-world images from conventional cameras is their limited dynamic range, which can cause bright regions of a scene to appear fully white and dark regions to appear fully black. This is a problem if the affected image areas contain important information, such as text.

The event camera [9] is a type of camera that addresses both these shortcomings of conventional cameras. Hence, it can replace and complement frame-based cameras in tasks where high dynamic range (HDR) and fast motion are common. Motivated by this fact, we propose to use an event camera for estimating NeRFs in challenging real-world capturing conditions. The main contributions of this paper are:

- This is the first attempt to tackle NeRF estimation from event cameras during very fast motion.
- Our method can combine the event stream with (color) images. This allows estimating NeRFs in challenging conditions where frame-based approaches fail or require additional regularization, e.g., under strong motion blur or if only a few frame-based views are available.
- We open-source our code and the simulated datasets.

II. Related Work

This work tackles the problem of reconstructing a 3D neural radiance field (NeRF) from a moving event camera. We thus present related work that reconstructs NeRFs in challenging scenarios and methods that perform 2D or 3D reconstruction from event data. We refer interested readers to the excellent resources on neural rendering [2], [3], event-based vision [9], and 3D reconstruction [5, Chap. 13].

a) 3D Scene Reconstruction: The reconstruction of 3D scenes from a moving RGB camera has traditionally been tackled by methods that produce sparse point clouds [31], [32], depth maps [30] or voxel grids [33]. Recently, NeRFs [1] have gained popularity due to their ability to synthesize high-quality novel views. Usually, the input images to NeRF contain little noise. One exception to this is Deblur-NeRF [4] which handles blurry input images by modeling the blur formation. Their method is limited to moderate motion blur, and they only model nonconsistent blur, i.e. they can only recover a NeRF for blur that is caused by irregular (non-straight, non-circular) motion trajectories. RawNeRF [6] can deal with images captured under low light conditions. Their method requires raw images and does not handle motion blur, which makes it applicable only to scenes captured with a temporarily static camera. Overall, none of these methods work for scenes captured by a moving camera under low light conditions, however, these conditions are common in robotics applications.

b) Event Cameras: Event cameras are vision sensors which transmit binary events per pixel, indicating an increase or decrease of the observed brightness. Converting this asynchronous data stream to images is well-studied in 2D [10], [11], [13], [12]. Rebeq et al. [10] train a recurrent U-Net on synthetic events and reconstruct intensities of high speed phenomna. However, since the event stream does not contain complete photometric information (it only contains derivative data and isomotion-dependent), several approaches combine frames with events. For example, Tulyakov et al. [14] train a fusion network for color frames and events, showing impressive results on the task of frame rate upsampling. Without requiring any training data, the approach by Pan et al. [11] combines motion-blurred frames with events using energy-based minimization. They model motion blur as an integral over the exposure time, and optimize for a global event threshold. Similarly, the work by Scheerlinck et al. [13] fuses frames with events using a classical filtering technique. Zhou et al. [17] compute a semi-dense 3D reconstruction from event data with known poses, as well as by estimating the poses simultaneously [15], [16]. These works produce sparse point clouds for navigation purposes, which in contrast to our work are not well-suited for (novel-view) visualizations.

c) Concurrent Work: Rudnev et al. [18] (EventNeRF) and Hwang et al. [19] (Ev-NeRF) investigate how NeRFs could be reconstructed from event data. These works show the need of the event vision community to generate 3D reconstructions from events. Rudnev et al. [18] aim to reconstruct visually pleasing NeRFs in color space by using a color event camera. Their setting is constrained to a static event camera with controlled illumination and fully controlled object motion (rotation on a turntable). They assume that events are only triggered by the foreground object, and require the background color to be known. In contrast, we focus on estimating NeRFs from a moving event camera with many background events and under general illumination. Hwang et al. [19] reconstruct NeRFs from a slowly moving event camera without considering motion blur. In contrast to both of these works, we focus our evaluation on failure cases of frame-based cameras. Additionally, we are the only work to show NeRF reconstructions from a combination of events and frames leveraging the advantages of both sensor modalities. We introduce a novel no-event loss function and we do not require the contrast threshold to be determined explicitly. Furthermore, E-NeRF does not rely on an integration window but employs directly neighboring events in the loss.

III. Method

A. Optimizing a Neural Radiance Field

Akin to NeRF [1], E-NeRF optimizes a neural radiance field parameterized by an MLP $F_\Theta : (\mathbf{x}, \mathbf{d}) \rightarrow (\mathbf{c}, \rho)$, which maps Cartesian input coordinates $\mathbf{x} \in \mathbb{R}^3$ and viewing direction

TABLE I: SOTA comparison on the spiral sequences. The camera is moving on two different spiral paths (Spiral 1 and Spiral 2) around the synthetic scene, while continuously facing inward to the center of interest. The background texture shows a colored road network (see Figure 2). The sparse sequences use fewer train views and all sequences show notable motion blur. The proposed E-NeRF shows the best results in the event-only setting as well as when combining events and blurred frames. It shows even better results than NeRF trained on perfectly sharp images, which is partially due to the sparseness of the training views. Deblur-NeRF is often worse than NeRF on the blurred frames because the blur is consistent (cf. Deblur-NeRF paper) and the views are sparse – this reverses for some of the shake sequences in Table II. The methods are evaluated on sharp test views using PSNR, SSIM [23], and LPIPS [24]. The best and second-best result are marked in **bold** and underlined, respectively. †The "NeRF w/o blur" baseline uses groundtruth sharp frames that would not be available in real-world applications and is thus never marked. We use a batch size of 30096 event pairs in all spiral sequences.

	Spiral 1 train/test views: 30/32			Spiral 2 train/test views: 20/24			Spiral 1 (sparse) train/test views: 17/32			Spiral 1 (more-sparse) train/test views: 9/32		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
torch-ngp [25] w/o blur†	25.74	0.74	0.06	24.36	0.63	0.06	20.67	0.38	0.11	19.85	0.26	0.16
torch-ngp [25] w/ blur	23.18	0.59	0.09	23.85	0.54	0.09	20.49	0.32	0.13	19.80	0.26	0.17
Deblur-NeRF [4]	21.23	0.46	0.23	18.10	0.18	0.72	17.32	0.18	0.72	16.88	0.16	0.70
E2VID [27] + torch-ngp [25]	21.86	0.47	0.14	21.83	0.30	0.13	19.77	0.26	0.18	19.75	0.26	0.17
E-NeRF event-only	26.91	0.82	0.04	25.71	0.79	0.04	26.91	0.82	0.04	26.91	0.82	0.04
E-NeRF events & blur	**28.23**	**0.85**	**0.04**	**27.92**	**0.82**	**0.03**	**27.34**	**0.85**	**0.04**	**27.27**	**0.83**	**0.04**

$\mathbf{d} \in \mathbb{S}^2$ to the predicted color $\mathbf{c} \in \mathbb{R}^3$ and volume density $\rho \in \mathbb{R}$, see Figure 1. The predicted color value $I(\mathbf{u})$ at pixel $\mathbf{u} \in \mathbb{R}^2$ is obtained by volumetric rendering [35] of N color predictions $\{\mathbf{c}_{k,j}\}_{j=1}^{j=N}$ along the ray $\mathbf{r}_k$, where $\mathbf{r}_k = \pi^{-1}(\mathbf{T}(t_k), \mathbf{u}_k, \mathbf{K}, \boldsymbol{\kappa})$, with interpolated camera poses $\mathbf{T}(t_k) \in SE(3)$, pre-calibrated intrinsics $\mathbf{K}$ and pre-calibrated distortion parameters $\boldsymbol{\kappa}$. We perform uniform sampling along each ray. The original NeRF minimizes a least squares error between the rendered predictions $\hat{I}(\mathbf{u})$ and ground truth colors $I(\mathbf{u})$ provided by the images, whereas E-NeRF utilizes the event generation model Equation 1 to optimize F_Θ, which is detailed in the next section. Because ordinary neural networks do not accurately represent high-frequency details, one key implementation detail is to use a proper encoding [21] of input coordinates $(\mathbf{x}, \mathbf{d})$. NeRF solves this by applying a Fourier-based feature mapping, whereas E-NeRF use a multi-resolution hash-encoding [20].

B. Event Stream Utilization

Input to our optimization problem is a continuous stream of events $e_k = (\mathbf{u}_k, t_k, p_k)$ which occur asynchronously at pixel $\mathbf{u}_k$ with micro-second timestamp t_k. The polarity $p_k \in \{+1, -1\}$ indicates an increase or decrease of the logarithmic brightness $L(\mathbf{u}_k, t_k)$ by the contrast threshold C, i.e. an event at time t_k is triggered if the following condition holds:

$$\Delta L = L(\mathbf{u}_k, t_k) - L(\mathbf{u}_k, t_{k-1}) = p_k\, C \tag{1}$$

where t_{k-1} is the time of the last event at pixel $\mathbf{u}_k$ and

$$L(\mathbf{u}, t) = \text{linlog}(I(\mathbf{u})) = \begin{cases} I(\mathbf{u}) \cdot ln(B)/B, & \text{if } I(\mathbf{u}) < B \\ ln(I(\mathbf{u})), & \text{else.} \end{cases} \tag{2}$$

The threshold B determines the linear region, where no logarithmic mapping is applied. Following v2e [22] we set $B = 20$, which models realistic event distributions. For each event e_k, we compute two associated rays $\mathbf{r}_k$ and $\mathbf{r}_{k-1}$ at time t_k and t_{k-1}, respectively, see Figure 1. We query the MLP for its color predictions $\{\mathbf{c}_{k,j}\}_{j=1}^{j=N}$ and $\{\mathbf{c}_{k-1,j}\}_{j=1}^{j=N}$ along the respective rays and perform volumetric rendering to obtain $\hat{I}(\mathbf{u}_k)$ and $\hat{I}(\mathbf{u}_{k-1})$. We then perform the linlog mapping in equation 2 to obtain the predicted logarithmic brightness difference $\Delta\hat{L}_k = \hat{L}(\mathbf{x}_k, t_k) - \hat{L}(\mathbf{x}_k, t_{k-1})$. Stacking all N_{evs} predictions into $\Delta\hat{\mathbf{L}} \in \mathbb{R}^{N_{\text{evs}}}$ and assuming a Gaussian distribution of residuals $\Delta\hat{L}_k - \Delta L_k$, we perform a least squares minimization of the event loss

$$\mathcal{L}_{\text{evs}}(\Theta) = \|\Delta\hat{\mathbf{L}}(\Theta) - \Delta\mathbf{L}(\Theta)\|_2^2, \tag{3}$$

where $\Delta L_k = p_k\, C$ is measured by the event camera. The contrast threshold C of a real event camera varies over the image plane and over time [22], which can make Equation 1 impractical to use in a real-world setup. Hence, inspired by [16], for real-world data we propose to use

$$\mathcal{L}_{\text{evs, norm}} = \| \frac{\Delta\hat{\mathbf{L}}(\Theta)}{\|\Delta\hat{\mathbf{L}}(\Theta)\|_2} - \frac{\Delta\mathbf{L}(\Theta)}{\|\Delta\mathbf{L}(\Theta)\|_2} \|_2^2. \tag{4}$$

C. Event Sampling

In each optimization step, we include a large number of event pairs from different pixels and at different times. An event pairs consists of an event and its successor event in time (at the same pixel). Our method is general and can also work on event pairs $\{e_k, e_{k+j}\}$ which are not direct neighbors but further apart in time ($j > 1$). In this case, we accumulate the polarities of all events occurring between the chosen events, i.e. we set the measured brightness difference in our loss function to $\Delta L_k = C \sum_{i=k+1}^{i=k+j} p_i$. Uniform brightness areas do not trigger events. We can include this fact in our model by also sampling from areas where no event has occurred for a time period of T_{noevs}. A non-existing event means that L did neither increase nor decrease by more than C, hence we formulate the no-event loss as

$$\mathcal{L}_{\text{noevs}} = \sum_k relu(|\hat{L}_k - \hat{L}_{k-1}| - C). \tag{5}$$

We precompute no-event locations by saving their pixel coordinates $\mathbf{u}_{\text{noev}}$ as well as the beginning $t_0(\mathbf{u}_{\text{noev}})$ and end $t_1(\mathbf{u}_{\text{noev}})$ of the time interval T_{noevs}. During training we first sample a random no-event pixel and then sample two random timestamps in the interval $[t_0(\mathbf{u}_{\text{noev}}), t_1(\mathbf{u}_{\text{noev}})]$. Overall, if the no-event loss is enabled, we sample two thirds of event rays and one third of no-event rays. The final loss is a weighted combination of $\mathcal{L}_{\text{evs}} + \lambda_{\text{noevs}}\, \mathcal{L}_{\text{noevs}}$.

D. Implementation Details

We base our code on torch-ngp [25] employing a hash-based encoding [20]. The training usually converges on one NVIDIA A40 in about 1 to 3 hours (including elaborate logging and preprocessing). Larger batch sizes can improve PSNR values slightly and converge faster. Given external poses from a motion capture system, for each sampled event we interpolate the rotational component of its pose $\mathbf{T}(t_k)$ by spherical linear interpolation (slerp), and the translational component by cubic interpolation.

IV. EXPERIMENTS

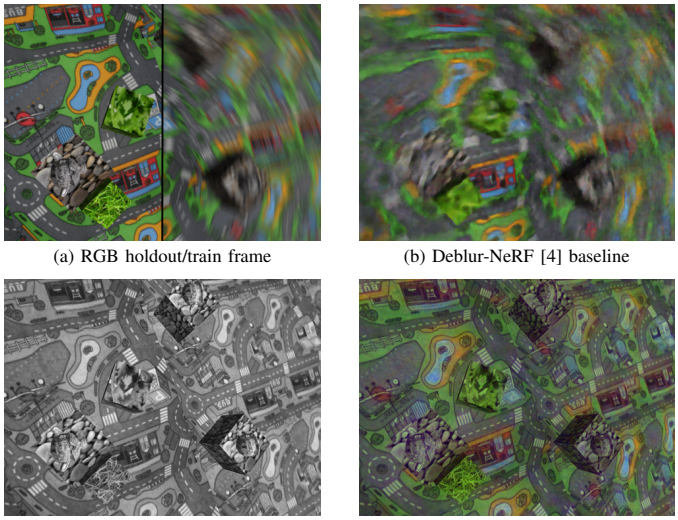
We evaluate the proposed E-NeRF on synthetic data based on ESIM [26] as well as on real-world, public benchmarks. We compare E-NeRF to several baselines: (i) frame-based torch-ngp [25] (ii) frame-based Deblur-NeRF [4] when considering scenarios with motion blur, and (iii) our novel baseline method consisting of frame reconstruction using E2VID [27] followed by torch-ngp [25][1]. While RawNeRF [6] would be a good additional baseline it cannot be used because there was no public code at the time of writing and because it requires raw images from a camera that is static during the exposure. Following the motivation of this work, we consider scenarios that are frequent in robotics and challenging for frame-based (and event-based) vision.

A. Synthetic Data

We simulate five sequences with different motion trajectories and background textures using the event camera simulator ESIM [26]. Quantitative image metrics are computed between rendered frames and holdout test frames without motion blur. Since the absolute brightness is non-observable from the event stream (see Equation 1), we perform an affine brightness transform on our predictions using the holdout test frames for all presented methods. The evaluation uses the peak signal to noise ratio (PSNR), the structural similarity (SSIM) [23], and the Learned Perceptual Image Patch Similarity (LPIPS) [24] using AlexNet.

a) Motion Blur: We simulate images with motion blur in ESIM by integrating irradiance maps over the exposure time, which yields realistic, motion-dependent blur [26, Sec. 7.4]. Table I and Table II show that our method produces better results than the baselines. Notably, the results are much better in comparison to combining E2VID with frame-based torch-ngp. While E2VID has the advantage of being trained on a large dataset, the proposed E-NeRF fuses information in 3D and can leverage spatial consistency without requiring any training data. Additionally, E-NeRF shows better results than frame-based methods, which cannot overcome the strong motion blur. While Deblur-NeRF shows good results for the mild blur in Table II, it is worse than torch-ngp for the strong blur in Table I. This result can be expected because the blur is consistent, i.e. not stemming from random motion in all directions, and the training views are not dense. This shows that carefully modelling the blur process can produce better results, but using blur-resistant event data is preferable. Finally, the results in Figure 2 show that E-NeRF is able to reconstruct sharp details, which can be beneficial for robotics applications. For example for licence plate detection it is important that the reconstruction is sharp and the characters are legible, while an overall photorealistic reconstruction is less important.

[1]E2VID yields strong artefacts during fast motion, degrading its performance significantly. For this reason, to be fair to the E2VID baseline, we only feed a subset of all events into E2VID (between 1/4th and 1/16th) during high event rates, and report the best result. This upsampling is common practice and required for good results, as noted in the official code of [34].



(a) RGB holdout/train frame (b) Deblur-NeRF [4] baseline

(c) E-NeRF (event-only) (d) E-NeRF (events & frames w/ blur).

Fig. 2: Qualitative evaluation of sequence Shake Carpet 1. (a) The input views contain strong motion blur. (b) The frame-only baseline Deblur-NeRF [4] can not recover sharp details. (c) In contrast, E-NeRF recovers sharp details from event data only. (d) When combining grayscale events and blurry color frames, our method is able to produce sharp and colored reconstructions. Due to only using grayscale events the colors are not photorealistic, but could still be useful for object recognition.

b) Combining Events and Frames: We combine events and blurry frames by a simple weighted combination

$$\mathcal{L} = \mathcal{L}_{\text{evs}} + \lambda_{\text{rgb}} \mathcal{L}_{\text{rgb}}, \tag{6}$$

where $\mathcal{L}_{\text{rgb}}$ is the original RGB rendering NeRF loss [1]. Using events yields sharp edges but uniform areas show fog-like artefacts due to the discrete nature of events. Additionally, the color for one shaded area might be slightly off because it is never directly measured but inferred from derivative-like data. Frame data can help to correct both problems. In Figure 2 we combine grayscale events and blurry color frames into a sharp and colored reconstruction. We do not require a color event camera but simply map the predicted color values of the event rays to grayscale. Note that most commercially available event cameras are grayscale. Moreover, combining events and frames opens up potential low-power reconstructions, where frames are triggered at low-frequency and the event camera captures necessary data in the blind time.

c) Ablation Study: We study our method by changing single components and report the average image metrics over all five simulated sequences. Changing the loss function from Equation 3 to the normalized loss in Equation 4 results in a minor drop of performance (see line (i) in Table III). This is expected, since the normalized loss is an approximation of the ideal physical model. However, the drop in performance is not

TABLE II: SOTA comparison on the shake sequences. The camera is performing a random, forward-facing motion with abrupt changes of direction in each sequence. The background textures in the Shake Moon sequences show a gray moon landscape, whereas the background in Shake Carpet 1 consists of a colored road network (see Figure 2). The proposed E-NeRF shows the best results in the event-only setting and outperforms NeRF trained on blurry frames. Deblur-NeRF is better than NeRF for mild blur, however, not for strong blur in sequence Shake Carpet 1 and not in Table I. This shows that modelling the blur process can be beneficial, but using blur-resistant events is preferable and hence E-NeRF shows better results. The methods are evaluated on sharp test views using PSNR, SSIM [23], and LPIPS [24]. The best and second-best result are marked in **bold** and underlined, respectively. †The "NeRF w/o blur" baseline uses groundtruth sharp frames that would not be available in real-world applications and is thus never marked. We use a batch size of 20096 event pairs in all shake sequences

.

	Shake Moon 1 train/test views: 39/38, mild blur			Shake Moon 2 train/test views: 33/32, medium blur			Shake Carpet 1 train/test views: 20/21, strong blur		
	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
torch-ngp [25] w/o blur†	33.12	0.93	0.02	28.41	0.84	0.03	29.42	0.87	0.03
torch-ngp [25] w/ blur	24.22	0.61	0.12	22.04	0.52	0.13	19.69	0.31	0.22
Deblur-NeRF [4]	25.97	0.68	0.20	24.78	0.58	0.36	16.44	0.18	0.69
E2VID [27] + torch-ngp [25]	22.39	0.80	0.05	22.31	0.78	0.06	22.81	0.80	0.05
E-NeRF event-only	31.10	0.94	0.02	30.32	0.93	0.02	26.65	**0.89**	0.03
E-NeRF events & blur	**31.35**	**0.95**	**0.01**	**30.60**	**0.93**	**0.02**	**27.90**	0.89	**0.03**

TABLE III: Ablation study of E-NeRF. We (i) replace the loss function Equation 3 by Equation 4, (ii) change the event sampling from an average event pair distance of 1ms to 30ms, (iii) add the no-event loss from Equation 5 to the loss function by sampling up to one fourth of no-events in each batch (with $T_{\text{noevs}} = 25$ms). While those modifications do not improve the average metrics, we find that using them on real data can result in better visual quality. The methods are evaluated using PSNR, SSIM [23], and LPIPS [24]. The best and second-best result are marked in **bold** and underlined, respectively.

		PSNR ↑	SSIM ↑	LPIPS ↓
	E-NeRF	**28.14**	**0.88**	**0.028**
(i)	normalized loss (Equation 4)	28.07	0.86	0.031
(ii)	event sampling 60ms window	28.13	0.86	0.033
(iii)	no-event loss $\lambda_{\text{noevs}} = 1$	27.76	0.87	0.030

very large, and we find the normalized loss function to work better on some of the real-world data (e.g. Figure 4).

Our default sampling strategy is to sample the closest event neighbors in time, which results in an average event pair distance of around 1ms. We investigate a different sampling with event pairs being farther apart in time. To do this, we first divide the event stream into windows of 60ms. For each sampled event, we pick the last event at that pixel in the 60ms window as neighbor, resulting in an average event pair distance of approximately 30ms. We show in line (ii) of table Table III that by changing the sampling strategy as described, we receive a minor drop in performance. We believe that this drop might be due to sampling some event rays repeatedly (at the end of the 60ms window). Since this strategy results in unstable training on Shake Moon 1 and Shake Moon 2, we only accumulate up to five events on those sequences.

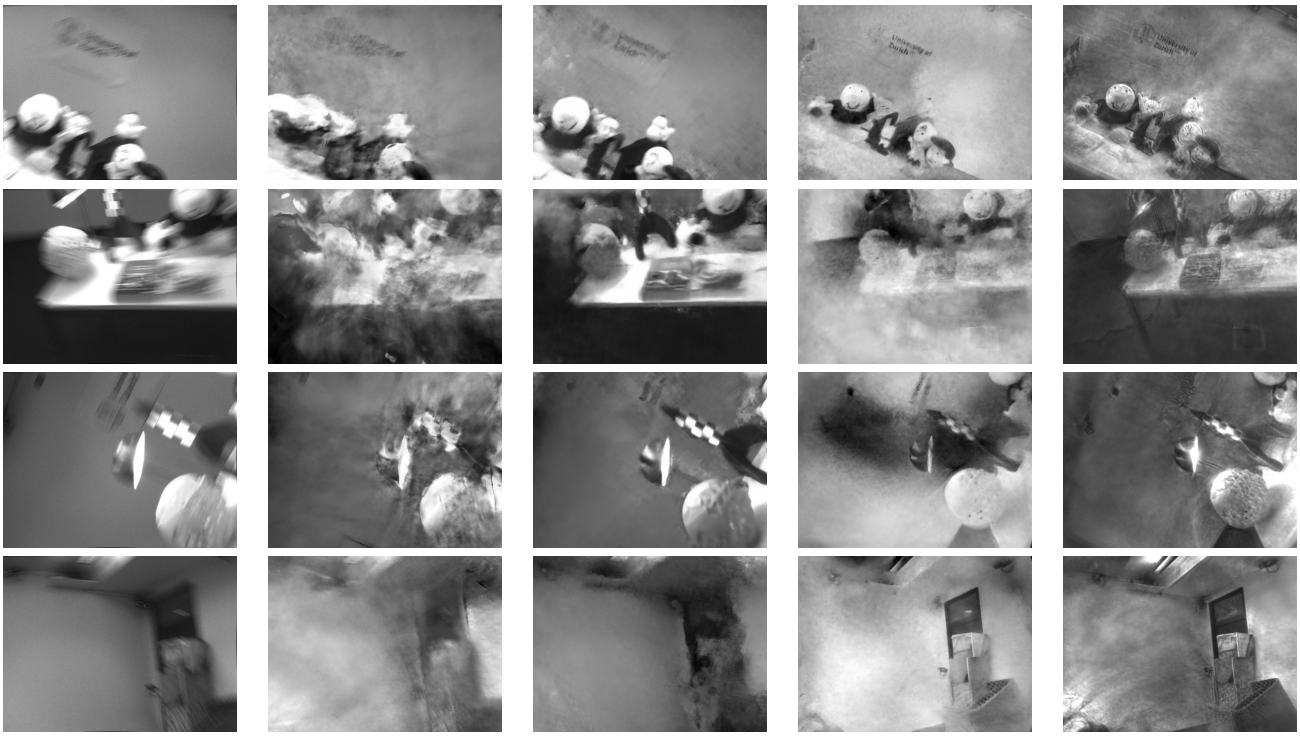
In line (iii) of table Table III we show the influence of adding the no-event loss Equation 5 with a weight of 0.1 to the event loss function. We search for no-events in the time interval $T_{\text{noevs}} = 25$ms and allow up to one fourth of all event rays in the batch to be no-events. This configuration also yields a minor drop in performance. However, we find that the no-event loss helps to remove artefacts in uniformly colored areas, see Figure 6. Moreover, using the no-event loss as well as the event accumulation, can help to improve the visual quality on real data, e.g. in Figure 3 and Figure 4.

B. Real Data

a) EDS: We evaluate our method on the public dataset accompanying EDS [16], which uses a beamsplitter to capture aligned frames and events from the same viewpoint[2]. We would like to note that for real data, there is no quantitative evaluation possible, since we pick sequences where the frames are severely degraded by motion blur or their low dynamic range. We use sequence 00 to evaluate our method in the dark, as well as sequence 11 to evaluate high speed motion. The selected scenarios are highly challenging for both camera modalities. The darkness in sequence 00 causes high noise levels in both cameras [22]. In sequence 11 the camera is moving at very high speed, which induces strong motion blur in the frames and a very high event rate which can lead to single events being dropped [28]. For the high-dynamic-range sequence 00 we show the results in Figure 4. While frame-based torch-ngp can give decent reconstructions in the bright areas of the scene, it struggles with the dark parts. Both event-based methods, E2VID+torch-ngp and E-NeRF, show more details. The proposed E-NeRF shows slightly fewer artefacts than E2VID+torch-ngp especially on foreground objects. For the high-speed sequence 11 we show the results in Figure 3. Deblur-NeRF provides better reconstructions than frame-based torch-ngp, but both methods fail to reconstruct sharp details like lettering in the background. E2VID+torch-ngp and E-NeRF can reconstruct these details and furthermore the proposed E-NeRF reconstructs the geometry of the table, which E2VID+torch-ngp fails to achieve. Overall, event-based methods are superior for the presented real-world low light and high-speed scenarios. Furthermore, E-NeRF delivers reconstructions with better high-frequency details and better geometry in comparison to E2VID+torch-ngp. No method achieves photorealistic quality, however, this is not necessary for many robotics applications and might be infeasible due to the difficulty of the tackled scenario.

b) TUMVIE: We evaluate our method on the sequence mocap-desk2. The camera follows a forward facing, circular path inducing mild motion blur in the frames. The results in Figure 5 show that E-NeRF can reconstruct fine details on the book cover and fine-grained text on the paper as well as on the opened book page which is not possible with frame-based torch-ngp, although the dataset features a high-quality uEye

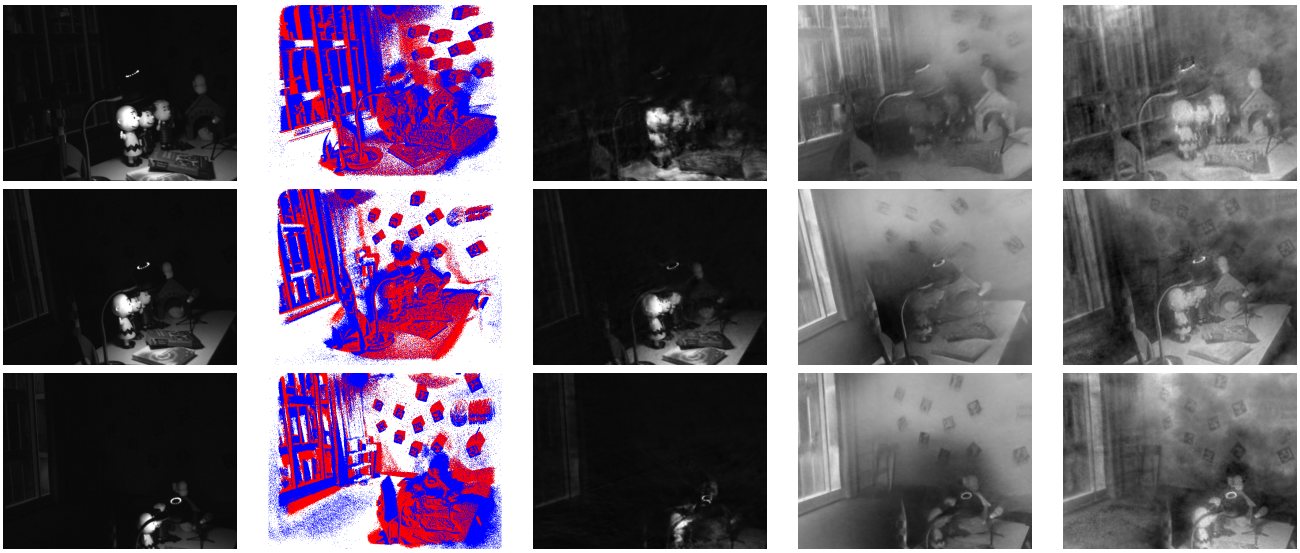
[2]Since the event and frame camera have different intrinsics parameters (which we use for training and evaluation), the field of view differs slightly in the shown renderings Figure 4 and Figure 3.



(a) holdout frame (b) torch-ngp [25] (c) Deblur-NeRF [4] (d) E2VID [27]+torch-ngp (e) E-NeRF (event-only)

Fig. 3: Qualitative evaluation on EDS-11 during high speed motion. Holdout frames (a) are not used during optimization. Due to very strong motion blur, the NeRF baseline (b) can not recover proper geometry. The Deblur-NeRF baseline (c) can reconstruct the scene, but shows artefacts on the wall and is missing details such as the eyes of the toy figure or the university logo. The E2VID+torch-ngp baseline (d) shows good performance on uniformly colored areas, while not being able to reconstruct detailed geometry and texture. Our method (e) can better reconstruct sharp details, such as the lettering, the book cover, the grid structure of the ball, as well as the texture of the boxes in the last row. The E-NeRF result is obtained by picking a fixed threshold of $C = 0.2$, and by adding $\mathcal{L}_{\text{noevs}}$ Equation 5 with weight one to the loss ($T_{\text{noevs}} = 30$ms), which improves visual quality on the white background. E2VID uses upsampling of factor 16. We use a batch size of 30096 event pairs.

.



(a) holdout frame (b) Events (c) torch-ngp [25] (d) E2VID [27]+torch-ngp (e) E-NeRF (event-only)

Fig. 4: Qualitative evaluation on EDS-00. We show the holdout frames (a), as well as the closest input events (b) to E2VID (d) and E-NeRF (e). Notice how the NeRF-baseline (c) shows geometry artefacts for some novel viewpoints and can not reconstruct the wall at the back of the room due to limited dynamic range. The E2VID+torch-ngp (d) baseline can recover details in the dark regions, but shows overly dark reconstructions of the illuminated figures. E-NeRF (e) can reconstruct sharp details in the foreground, such as the book cover and detailed clothing. E-NeRF can even reconstruct the writing on the background wall in the top-right corner of the third and fourth row. The E-NeRF result is obtained by using the normalized loss Equation 4, adding $\mathcal{L}_{\text{noevs}}$ Equation 5 with weight one to the loss ($T_{\text{noevs}} = 30$ms), and sampling from event windows of up to 130ms. We perform a simple contrast normalization on the E-NeRF renderings for better visual clarity. E2VID uses upsampling of factor of 4. We use a batch size of 30096 event pairs.

.

UI-3241LE-M-GL frame camera. We show that by combing events and frames, it is possible to improve the reconstruction in uniformly colored areas, e.g. on the table and on the dark keyboard. We empirically notice that the visual quality of E-NeRF is better on TUMVIE than on EDS. This might be due to the novel event camera model used in TUMVIE (Prophesee Gen4HD compared to Gen3 in EDS), as well as better illumination. In the second row of Figure 5 we show that torch-ngp can not reconstruct correct geometry when we decrease the number of input views to six. E2VID+torch-ngp can reconstruct uniform color areas on the mocap-desk2 sequence but it does not recover the text. To qualitatively study the influence of the no-event loss $\mathcal{L}_{\text{noevs}}$ (Equation 5) on our method, we perform a hyperparameter sweep over λ_{noevs} in Figure 6 on the scene mocap-1d-trans. The camera motion is slow and the illumination is good in this sequence. It can be noticed that uniform brightness areas, which do not trigger events, appear more smooth as λ_{noevs} increases.

c) Discussion: For challenging real-world scenarios in low-light conditions and with fast camera motion, E-NeRF clearly outperforms frame-based torch-ngp. Additionally, E-NeRF often shows clearer details and fewer artifacts than E2VID+torch-ngp; however, the gap is smaller than for synthetic data. The major reason is that the event generation model of Equation 1 is only an approximation and less accurate for these challenging scenarios. Switching to a different event camera model generally amounts to changing the loss function, which makes E-NeRF an excellent vehicle for studying event camera models for 3D reconstruction. Ultimately, a model that takes into account all spatial and temporal dependencies might be hard to specify and should be learned from data. We consider this a fruitful direction for future research that could lead to real-world E-NeRF reconstructions that are on par with reconstructions generated from synthetic data.

V. Conclusion

We present E-NeRF, the first method that reconstructs a neural radiance field from a fast-moving event camera. We specifically focus on scenarios that are common in robotics, such as strong motion blur or non-ideal illumination in the dark. We show on synthetic and real-world data that E-NeRF outperforms multiple baselines, and we ablate our design choices. In particular, we show that E-NeRF is able to better reconstruct high-frequency details such as text. We show that, if only a few input views are provided, an additional event stream helps to estimate NeRFs. Our method can even combine color frames with grayscale events to obtain a sharp, colored reconstruction that utilizes the strengths of both sensors. We see promising future work in the direction of devising better sensor models for event cameras (under challenging capturing conditions) and incorporating learning-based priors.

References

[1] Mildenhall, B., Srinivasan, P., Tancik, M., Barron, J., Ramamoorthi, R. & Ng, R. Nerf: Representing scenes as neural radiance fields for view synthesis. *ECCV*. (2020)

[2] Xie, Y., Takikawa, T., Saito, S., Litany, O., Yan, S., Khan, N., Tombari, F., Tompkin, J., Sitzmann, V. & Sridhar, S. Neural Fields in Visual Computing and Beyond. *Computer Graphics Forum*. (2022)

[3] Tewari, A., Thies, J., Mildenhall, B., Srinivasan, P., Tretschk, E., Wang, Y., Lassner, C., Sitzmann, V., Martin-Brualla, R., Lombardi, S., Simon, T., Theobalt, C., Niessner, M., Barron, J., Wetzstein, G., Zollhoefer, M. & Golyanik, V. Advances in Neural Rendering. *Computer Graphics Forum*. (2022)

[4] Ma, L., Li, X., Liao, J., Zhang, Q., Wang, X., Wang, J. & Sander, P. Deblur-NeRF: Neural Radiance Fields from Blurry Images. *CVPR*. (2022)

[5] Szeliski, R. Computer Vision - Algorithms and Applications, Second Edition. (2022)

[6] Mildenhall, B., Hedman, P., Martin-Brualla, R., Srinivasan, P. & Barron, J. NeRF in the Dark: High Dynamic Range View Synthesis from Noisy Raw Images. *CVPR*. (2022)

[7] Vidal, A., Rebecq, H., Horstschaefer, T. & Scaramuzza, D. Ultimate SLAM? Combining Events, Images, and IMU for Robust Visual SLAM in HDR and High-Speed Scenarios. *IEEE RA-L*. (2018)

[8] Zuo, Y., Yang, J., Chen, J., Wang, X., Wang, Y. & Kneip, L. DEVO: Depth-Event Camera Visual Odometry in Challenging Conditions. *IEEE ICRA*. (2022)

[9] Gallego, G., Delbrück, T., Orchard, G., Bartolozzi, C., Taba, B., Censi, A., Leutenegger, S., Davison, A., Conradt, J., Daniilidis, K. & Scaramuzza, D. Event-based Vision: A Survey. *IEEE T-PAMI*. (2020)

[10] Rebecq, H., Ranftl, R., Koltun, V. & Scaramuzza, D. High Speed and High Dynamic Range Video with an Event Camera. *IEEE T-PAMI*. (2019)

[11] Pan, L., Hartley, R., Scheerlinck, C., Liu, M., Yu, X. & Dai, Y. High Frame Rate Video Reconstruction based on an Event Camera. *IEEE T-PAMI*. (2020)

[12] Bardow, P., Davison, A. & Leutenegger, S. Simultaneous Optical Flow and Intensity Estimation from an Event Camera. *CVPR*. (2016)

[13] Scheerlinck, C., Barnes, N. & Mahony, R. Continuous-time Intensity Estimation Using Event Cameras. *IEEE ACCV*. (2018)

[14] Tulyakov, S., Bochicchio, A., Gehrig, D., Georgoulis, S., Li, Y. & Scaramuzza, D. Time Lens++: Event-based Frame Interpolation with Parametric Non-linear Flow and Multi-scale Fusion. *CVPR*. (2022)

[15] Zhou, Y., Gallego, G. & Shen, S. Event-Based Stereo Visual Odometry. *IEEE T-RO*. (2021)

[16] Hidalgo-Carrió, J., Gallego, G. & Scaramuzza, D. Event-aided Direct Sparse Odometry. *CVPR*. (2022)

[17] Zhou, Y., Gallego, G., Rebecq, H., Kneip, L., Li, H. & Scaramuzza, D. Semi-dense 3D Reconstruction with a Stereo Event Camera. *ECCV*. (2018)

[18] Rudnev, V., Elgharib, M., Theobalt, C. & Golyanik, V. EventNeRF: Neural Radiance Fields from a Single Colour Event Camera. (2022), arXiv:2206.11896v1

[19] Hwang, I., Kim, J. & Kim, Y. Ev-NeRF: Event Based Neural Radiance Field. (2022), arXiv:2206.12455v1

[20] Müller, T., Evans, A., Schied, C. & Keller, A. Instant Neural Graphics Primitives with a Multiresolution Hash Encoding. *ACM T-OG*. (2022)

[21] Tancik, M., Srinivasan, P., Mildenhall, B., Fridovich-Keil, S., Raghavan, N., Singhal, U., Ramamoorthi, R., Barron, J. & Ng, R. Fourier Features Let Networks Learn High Frequency Functions in Low Dimensional Domains. *NeurIPS*. (2020)

[22] Delbrück, T., Hu, Y. & He, Z. V2E: From video frames to realistic DVS event camera streams. (2020), arXiv

[23] Wang, Z., Bovik, A., Sheikh, H. & Simoncelli, E. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions On Image Processing*. **13** (2004)

[24] Zhang, R., Isola, P., Efros, A., Shechtman, E. & Wang, O. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. *CVPR*. (2018)

[25] Tang, J. Torch-ngp: a PyTorch implementation of instant-ngp. (2022), https://github.com/ashawkey/torch-ngp

[26] Rebecq, H., Gehrig, D. & Scaramuzza, D. ESIM: an Open Event Camera Simulator. *CoRL*. (2018)

[27] Rebecq, H., Ranftl, R., Koltun, V. & Scaramuzza, D. High Speed and High Dynamic Range Video with an Event Camera. *IEEE T-PAMI*. (2021)

[28] Delbruck, T., Graca, R. & Paluch, M. Feedback control of event cameras. *(CVPR Workshops)*. (2021)

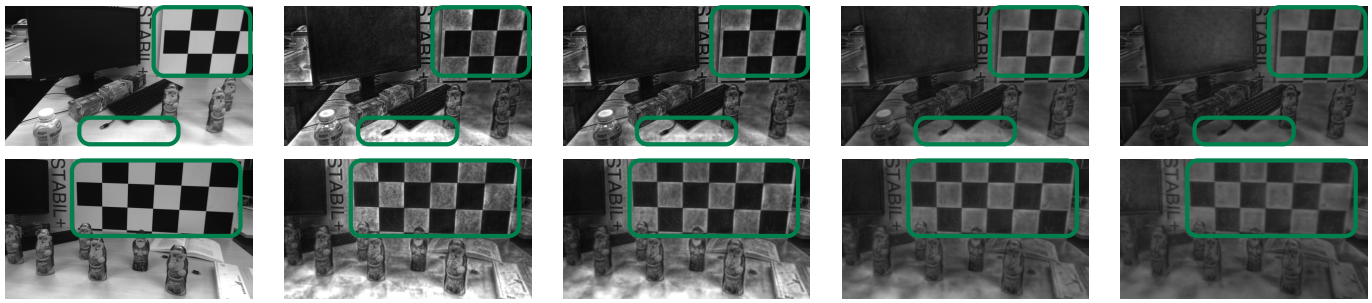
[29] Klenk, S., Chui, J., Demmel, N. & Cremers, D. TUM-VIE: The TUM Stereo Visual-Inertial Event Dataset. *IEEE IROS*. (2021)

[30] Newcombe, R., Lovegrove, S. & Davison, A. DTAM: Dense tracking and mapping in real-time. *ICCV*. (2011)

(a) holdout frame (b) torch-ngp [25] (c) E2VID [27]+torch-ngp (d) E-NeRF (event-only) (e) E-NeRF (events + frames)

Fig. 5: Qualitative evaluation on TUM-VIE mocap-desk2. We crop the renderings of all methods to highlight the same part of the scene. (i) Due to strong specularities at the table and mild motion blur, the torch-ngp baseline with 60 train views (b) is not fully sharp and shows artefacts around the book cover and text. E2VID+torch-ngp (c) can recover more details on the dark keyboard but is not fully sharp and fails to recover text. E-NeRF (d) can reconstruct high-frequency details such as the outline of the book cover and fine-grained text on the papers. Combining events and frames (e) removes artefacts on uniformly colored areas on the table and on white spaces of the book cover and papers, while preserving high frequency details. (ii) By using only six train views, torch-ngp is not able to reconstruct proper geometry. E2VID+torch-ngp gives decent reconstructions on the table but fails to capture fine details such as the monitor buttons and text, which E-NeRF can reconstruct properly. Combining the six train views with events (e) improves visual artefacts of E-NeRF in uniform brightness areas, e.g. on the monitor and table. The E-NeRF result is obtained by picking a fixed threshold of $C = 0.2$. E2VID uses upsampling of factor of 4. We use a batch size of 20096 event pairs.

.



(a) holdout frame (b) $\lambda_{\text{noevs}} = 0$ (c) $\lambda_{\text{noevs}} = 10$ (d) $\lambda_{\text{noevs}} = 100$ (e) $\lambda_{\text{noevs}} = 1000$

Fig. 6: The influence of the no-event loss $\mathcal{L}_{\text{noevs}}$ on the sequence mocap-1d-trans [29], where the camera is moving slowly and the illumination is good. By increasing the weight λ_{noevs}, uniform brightness areas which do not trigger events are regularized and appear more smooth, such as the checkerboard in the background or the table surface. We use $T_{\text{noevs}} = 20$ miliseconds to determine the no-event locations and a batch size of 20096 event pairs.

[31] Engel, J., Koltun, V. & Cremers, D. Direct Sparse Odometry. *IEEE T-PAMI*. (2018)
[32] Schönberger, J. & Frahm, J. Structure-from-Motion Revisited. *CVPR*. (2016)
[33] Koestler, L., Yang, N., Zeller, N. & Cremers, D. TANDEM: Tracking and Dense Mapping in Real-time using Deep Multi-view Stereo. *CoRL*. (2021)
[34] Muglikar, M., Gehrig, M., Gehrig, D. & Scaramuzza, D. How To Calibrate Your Event Camera. *CVPR Workshops*. (2021)
[35] Max, N. Optical Models for Direct Volume Rendering. *IEEE T-VCG*. (1995)