EventNeRF: Neural Radiance Fields from a Single Colour Event Camera

Viktor Rudnev[1, 2] Mohamed Elgharib[1] Christian Theobalt[1] Vladislav Golyanik[1]

[1]Max Planck Institute for Informatics, SIC [2]Saarland University, SIC

This supplemental document provides more detail on the experiments and technical aspects for the proposed technique. In Sec. A we discuss our experimental setup for capturing real data and describe our camera calibration and other details. We then show more results on synthetic (Sec. B) and real data (Sec. C). Here, we show visual results for ssl-E2VID [5], which as mentioned in the main document are clearly worse than E2VID [6]. We then show an application of running our approach in real-time through an instant-ngp implementation [7] (Sec. D). We demonstrate the ability to extract meshes from our trained models in Sec. F. Then we provide details on the window sampling in Sec. G. Finally, we include an additional ablation study for our method, *i.e.,* on real data (Sec. H) and study the effect of inaccurate camera poses (Sec. I) and the robustness to noise events in the training data (Sec. J).

Figure 1. Our real data recording setup. The object is placed on a 45 RPM direct-drive vinyl turntable and lit by a 5W USB ring light mounted directly above it. The scene is recorded with a DAVIS 346C colour event camera (right bottom).

A. Real Data Capture

We use the DAVIS 346C colour event camera to record our real sequence. Fig. 1 shows photos of the setup we used to record the real data. We used the default camera settings in the DV software provided with the camera.

A.1. Camera Pose Calibration

We estimate the extrinsic parameters of the event camera as follows. We noticed that due to the constant rotation speed of the turntable, camera extrinsics can be computed analytically as a point uniformly moving on a circle looking at its centre. In practice, this requires both precise mechanical and computational calibration.

First, we adjusted the camera tripod as precise as possible so that the vertical through the optical centre of the camera matches the turntable rotation axis (Fig. 2)

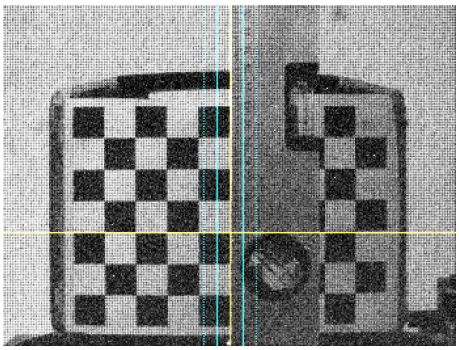

Figure 2. Results of the mechanical adjustment of the camera tripod as seen through the camera itself. The chequerboard and the ruler are placed exactly at the turntable rotation axis. Here, yellow is the vertical going through the optical centre of the camera. As seen in the picture, both axes match up to a pixel. Please note that the blue lines mark +/-10 and +/-20 pixels from the centre.

Second, we estimated the residual offset using the following protocol. We placed the chequerboard pattern with its centre as close on the turntable's axis of rotation as possible. Then we slowly rotated the plate and recorded corresponding RGB frames using the event camera. Using this data, we found the positions and rotations of the camera in

space wrt. the chequerboard. These positions lie on the circle, which corresponds to the correct camera poses, and they are tilted to the rotational axis with an unknown angle offset α. We solve for α, circle radius and circle centre coordinates via optimisation. The optimisation objective is such that all the rays coming through the optical centre of the cameras must meet in the same point in space that is the centre of the circle at a distance that equals to the radius of the circle. We use Adam [1] optimiser with its learning rate reduced on plateaus. We show the converged results and convergence process on Fig. 3. In our recordings, we found

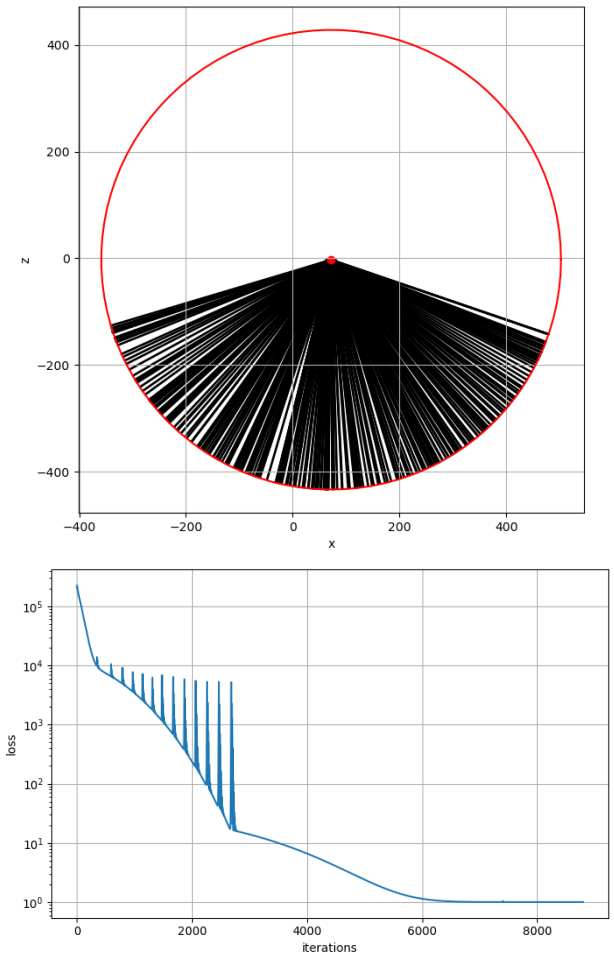

Figure 3. Camera extrinsic optimisation results (top) and convergence process (bottom) using the chequerboard protocol. In red is the optimised circle and the circle centre. In black are the arrows starting from the extracted camera poses meeting in the circle centre. In blue is the loss function convergence; the peaks in the optimisation plot are caused by learning rate scheduling.

that $\alpha = 2.85°$ for the Goatling and Sewing recordings and $\alpha = 0.2388°$ for the rest of the sequences.

A.2. Density Clipping

For the real scenes, we know that the object always lies inside the cylinder defined by the turntable plate. Hence, to filter the noise and artefacts in the unobserved areas, we force the density to zero everywhere outside of this cylinder:

$$\sigma(x, y, z) = 0, \text{ if } x^2+y^2 > r_{\max}^2 \text{ or } z > z_{\max} \text{ or } z < z_{\min}. \quad (1)$$

The cylinder parameters $z_{\min}$, $z_{\max}$ and $r_{\max}$ are tuned manually to fit the recorded experimental setup. In our case, $z_{\min} = -0.35$, $z_{\max} = 0.15$ and $r_{\max} = 0.25$.

B. Synthetic Data Results

We provide more synthetic data results and comparisons in Fig. 4 and in the supplementary video. E2VID [6]+NeRF [4] struggles with background reproduction and separation, resulting in less clear images and background artefacts. In addition, the colour and detail reproduction are also a concern for E2VID+NeRF. Our method does not suffer from such problems. It produces photo-realistic results, capturing specularities (Lego, Materials, Microphone), thin structures (Drums, Ficus, Lego, Microphone), and textured regions (Hotdog, Chair).

Note that Tab. 1 in the main document shows that our approach clearly outperforms E2VID [6]+NeRF [4].

C. Real Data Results

We provide more real data results in Fig. 5 and in the supplementary video. In the "Plant" scene, we can reconstruct every stem and thin leaf. In the "Sewing" scene, we recover even a one-pixel-wide needle of the machine (best viewed with zoom). In the "Microphone" scene, we can reconstruct fine details such as a microphone grid. In the "Controller" scene, we preserve its details in the dark regions despite having low contrast. Similarly, in the "Goatling" scene, we can reconstruct both the details in the dark and bright highlights on the glasses. In the "Cube" scene, EventNeRF recovers sharp colour details. "Multimeter", "Cube" and "Sewing" show how we recover view-dependent effects. In the "Bottle" scene, we can see the drawings on the reconstructed label. All our results are halo-free.

We show more visual results for DeblurNeRF [3] in Fig. 6 and Fig. 7 (top). As stated in the main document, Deblur-NeRF can not handle view-consistent blur and thus produces clearly worse results than our method. We also show in the same figures visual results for E2VID [6]+NeRF [4] and ssl-E2VID [5]+NeRF [4]. As stated in the main document, results produced by ssl-E2VID are clearly worse than E2VID, In addition, ssl-E2VID can only generate grayscale images. Our approach, however, outperforms all related methods.

Figure 4. Additional results and E2VID [6]+NeRF [4] comparisons on all of the used synthetic scenes (from top: Hotdog, Chair, Drums, Ficus, Lego, Materials and Microphone).

D. Real-Time Implementation

We show an interactive application of our approach that runs in real-time. For this, we implement our method using torch-ngp [7] instead of the original NeRF representation [4]. Training a model using this implementation takes around a minute using a single NVIDIA GeForce RTX 2070 GPU. At the test time, the method runs in real-time. We show in Fig. 8 visual results for this implementation. For image sequence results, please refer to the supplemental video. Our real-time implementation produces highly photorealistic results that can be viewed from an arbitrary viewpoint. This, however, can come with some trade-off in the rendering quality as shown in Tab. 1.

E. Tone-Mapping in Different Scenarios

The primary difference between the appearance of real, synthetic and the real-time results is the tone-mapping used. As real events have no ground-truth tone-mapping, we tune

Figure 5. Additional EventNeRF reconstructions of the real scenes. We show two arbitrary views per scene.

it manually. To show the details in the dark regions we opt for the reduced contrast in Fig. 4 of the main document which results in images that could look washed out. The speed gain of the real-time implementation is from using torch-ngp [7], which is significantly faster than the original NeRF [4].

F. Mesh Extraction

As another application, we show in Fig. 9 that we can extract a mesh from our NeRF reconstruction using marching

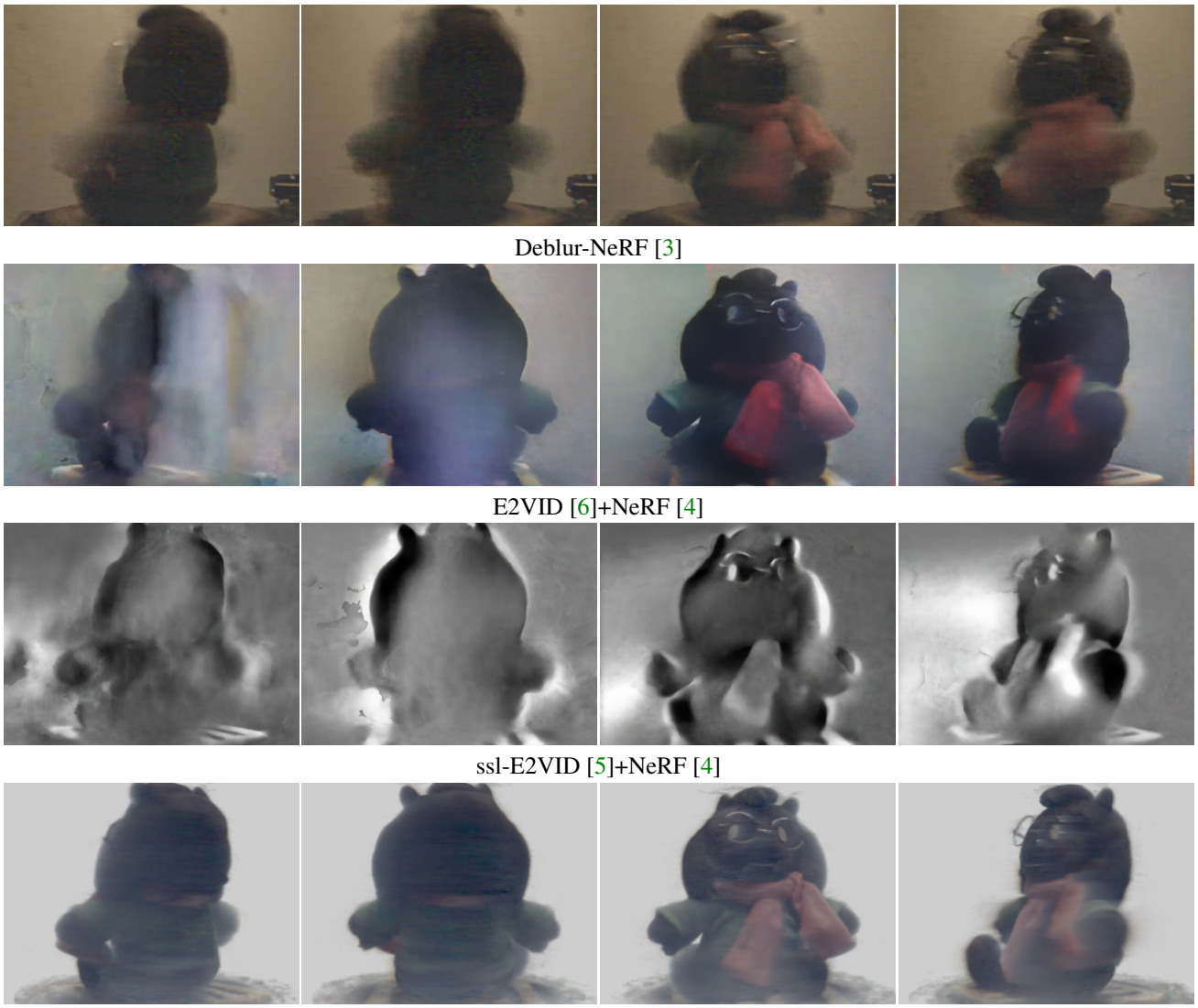

Figure 6. Results generated by different approaches. Our EventNeRF clearly outperforms all the methods.

	Our EventNeRF			Our Real-time Implementation		
Scene	PSNR ↑	SSIM ↑	LPIPS ↓	PSNR ↑	SSIM ↑	LPIPS ↓
Drums	**27.43**	**0.91**	**0.07**	26.03	**0.91**	**0.07**
Lego	**25.84**	**0.89**	0.13	22.82	**0.89**	**0.08**
Chair	**30.62**	**0.94**	**0.05**	27.97	**0.94**	**0.05**
Ficus	**31.94**	**0.94**	**0.05**	26.77	0.92	0.12
Mic	**31.78**	**0.96**	**0.03**	28.34	0.95	0.04
Hotdog	**30.26**	**0.94**	**0.04**	23.99	0.93	0.10
Materials	24.10	**0.94**	**0.07**	**26.05**	0.93	**0.07**
Average	**28.85**	**0.93**	**0.06**	25.99	0.92	0.07

Table 1. Comparing our method using the original NeRF implementation [4] (EventNeRF) against a real-time implementation based on torch-ngp [7]. While the real-time implementation takes significantly less training and testing time, it can compromise some of the rendering quality.

cubes [2]. The extracted mesh is textured and can be rendered from an arbitrary camera viewpoint. For more results, please see the interactive demo in our video.

G. Event Window Temporal Bounds Sampling

In Sec. 3.7 of the main document, the ends of the time intervals t of all windows are fixed and uniformly distributed through the whole length of the stream. This way, all views are sampled uniformly. As our recordings are perfect loops, we concatenate corresponding events from the end when there are not enough events from the start of the stream for the sampled window length. Window lengths $L_{\min}$ and $L_{\max}$ were chosen empirically as the highest and lowest val-

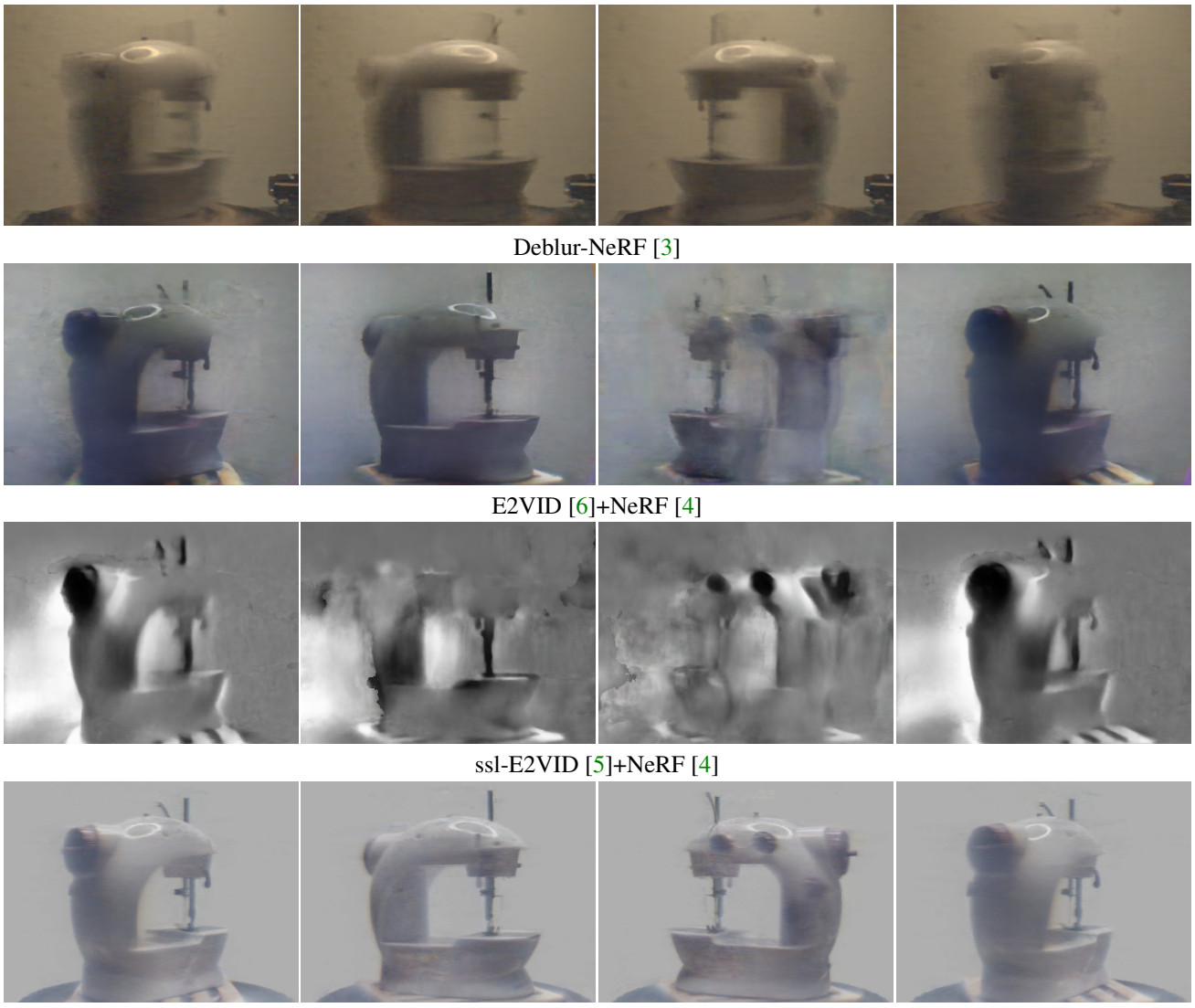

Figure 7. Results generated by different approaches. Our EventNeRF clearly outperforms all the methods.

ues that did not result in the model diverging.

H. Ablation Study on Real Data

We ran the same ablation studies as in Sec. 4.4 of the main document on real data; see Fig. 10 for the qualitative results on the "Sewing" sequence. Our full method produces the best results. Using short constant window length instead of a randomised one results in significant artefacts. As only views close to each other are used to supervise the model in this case, long-term consistency and low-frequency lighting does not propagate well. Using long constant window length leads to the noticeable blur in the reconstruction as short-time details and high-frequency lighting information is not present in long windows. Using only positive sampling causes more artefacts than in the full model with negative sampling. We note that this effect is less severe than in the case of synthetic data in Fig. 6 of the main document. The differences are perhaps explained by the inaccuracies in the camera parameters for the real data. If the camera poses are highly accurate, the negative sampling becomes more important; otherwise, the differences will be smaller than the artefacts caused by the inaccuracies in camera parameters. Nevertheless, these differences still can be noticed in both real and synthetic data models. To explore this hypothesis further, we analyse the effect of introducing error to the camera parameters in the next section.

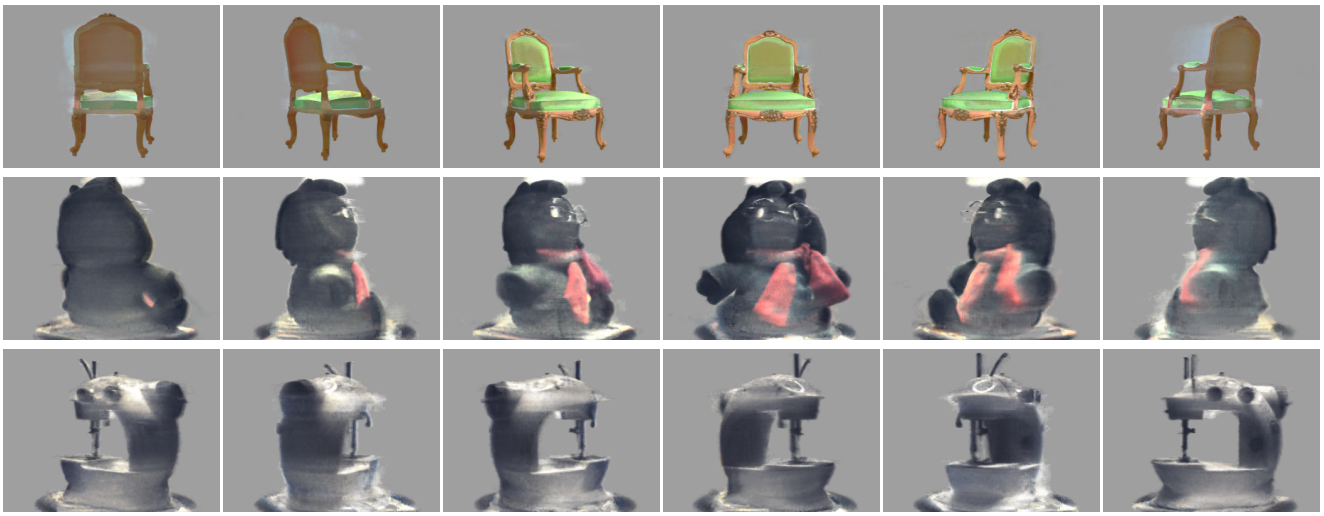

Figure 8. Results on synthetic (top) and real sequences using our real-time implementation. This approach takes around a minute to train and runs in real-time during the test.

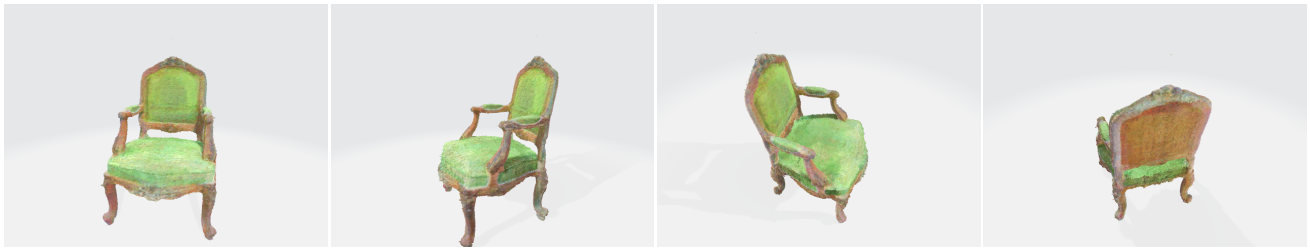

Figure 9. We extract a textured mesh using marching cubes [2]. The mesh can be viewed from an arbitrary viewpoint.

I. Robustness to Camera Pose Errors

We noticed that camera pose errors lead to trailing artefacts. Hence, we developed a camera pose calibration technique for our setup which we describe in detail in Sec. A.1. This led to cleaner predictions.

To measure this effect, we introduce error into synthetic data camera poses and measure the performance on "Drums" (see Fig. 11). $0°$, $0.01°$, $0.1°$, $1°$, $2°$ errors results in 27.43, 27.26, 26.18, 18.11, 17.49 dB PSNR, correspondingly ($1°$ translates to 10-15 pixels offset at 346×260 pixels image resolution). The steep change starting at $1°$ is due to significant increase in trailing artefacts. This suggests that at some threshold level of camera pose error between $0.1°$ and $1°$, the reconstruction quality starts to rapidly degrade.

J. Robustness to Noisy Events

For real data, we measure the amount of noise when recording a static scene with a static camera. We record the number of noise events per second ($\mathrm{ev/s}$) using the lowest and the highest brightness settings of our light source. Both settings result in similar noise measurements of around $1.1 \cdot 10^5$ $\mathrm{ev/s}$. Hence, we believe the amount of noise with our camera settings is almost constant regardless of the scene brightness. In our real data experiments, this amounts to $10-18\%$ of the training event streams being noise.

With improved lighting, we do not expect our results to change significantly. The event camera is reporting in the log-space and, hence, would emit the same number of events as long as the contrast between the darkest and brightest parts is the same.

For synthetic data, we add no noise to the event stream in the main document. However, if we add 1, 5, 15% of noisy events to the training event stream, we obtain 27.35, 27.36, 27.31 dB PSNR on "Drums" (w/o noise: 27.43 dB). The visualisations of the experimental results given in Fig. 12 also confirm that the change in quality is slight between all tested models. This suggests that our method is sufficiently robust to event noise in the training data and that it is not the primary cause of artefacts in the case of real data.

Full EventNeRF | No negative sampling | Const. short 10ms windows | Const. long 100ms windows

Figure 10. Ablation studies on real data for the "Sewing" scene. The full model provides the best results with the most detail and fewest artefacts. In case of "No negative sampling", note the artefacts above and around the object. In the case of constant long windows, note the blurriness. For the detailed analysis of these results, please refer to Sec. H.

0° (27.43 dB PSNR) | 0.01° (27.26 dB PSNR) | 0.1° (26.18 dB PSNR) | 1° (18.11 dB PSNR) | 2° (17.49 dB PSNR)

Figure 11. Study on the robustness of EventNeRF to the added error in camera parameters on "Drums". Rotation error (specified in degrees) is added to the camera extrinsics to simulate inaccuracies in camera pose calibration described in Sec. A.1. The results suggest that above a certain threshold between $0.1°$ and $1°$, the reconstruction quality starts to rapidly decay due to the trailing artefacts. Please refer to Sec. I for our detailed analysis of these results.

0% (27.43 dB PSNR) | 1% (27.35 dB PSNR) | 5% (27.36 dB PSNR) | 15% (27.31 dB PSNR)

Figure 12. Study on the robustness of EventNeRF to the added noise events on "Drums". A uniform number of random events (specified in percentages) is added to the training event stream to simulate noise found in real data. The subtlety of the differences suggests that our model is robust to noise events in amounts found within real data captures. Please refer to Sec. J for our detailed analysis of these results.

References

[1] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR) (Poster)*, 2015. 2

[2] William E Lorensen and Harvey E Cline. Marching cubes: A high resolution 3d surface construction algorithm. *SIGGRAPH*, 21(4):163–169, 1987. 5, 7

[3] Li Ma, Xiaoyu Li, Jing Liao, Qi Zhang, Xuan Wang, Jue

Wang, and Pedro V Sander. Deblur-nerf: Neural radiance fields from blurry images. In *Computer Vision and Pattern Recognition (CVPR)*, pages 12861–12870, 2022. 2, 5, 6

[4] 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)*, 2020. 2, 3, 4, 5, 6

[5] Federico Paredes-Vallés and Guido C. H. E. de Croon. Back to event basics: Self-supervised learning of image reconstruction for event cameras via photometric constancy. In *Computer Vision and Pattern Recognition (CVPR)*, pages 3445–3454, 2021. 1, 2, 5, 6

[6] Henri Rebecq, René Ranftl, Vladlen Koltun, and Davide Scaramuzza. High speed and high dynamic range video with an event camera. *Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2019. 1, 2, 3, 5, 6

[7] Jiaxiang Tang. Torch-ngp: a pytorch implementation of instant-ngp, 2022. https://github.com/ashawkey/torch-ngp. 1, 3, 4, 5