Supplementary
EvDNeRF: Reconstructing Event Data with Dynamic Neural Radiance Fields

	Jet-Down intensity image reconstruction			
	PSNR↑	SSIM↑	LPIPS↓	MAE↓
DNeRF	27.43	0.873	0.183	0.012
E2VID+DNeRF	24.50	0.533	0.262	0.028
EvDNeRF+DNeRF	27.13	0.807	0.140	0.013
EvDNeRF (ours)	26.98	0.835	0.138	0.013

Table 1. Quantitative metrics for intensity image reconstruction, comparing EvDNeRF to baselines for the simulated Jet-Down dataset.

A. Code, datasets, and pretrained weights

Code, datasets, and pretrained weights can be found here: https://github.com/anish-bhattacharya/EvDNeRF.

B. Image reconstruction

We design EvDNeRF to predict events reconstructions well, and do not expect it to perform well on absolute intensity images or depth of the scene. Training purely from events data does not provide complete scene information to achieve good absolute-value reconstructions. However, for completeness and discussion, we present examples of image and depth reconstructions by EvDNeRF in Figure 2, and quantitative metrics for image reconstruction in Table 1 and Figure 3. The objects can be made out by the human eye, but there are floating artifacts, loss of fine details, and background depth inaccuracies. The problem of image reconstruction is challenging since we are only training from events; this is especially noticeable in the loss of fine details in the image of the jet and in the poor quality of the stationary cabin of the lego tractor (the only supervision for training EvDNeRF on the cabin is when the bucket passes in front of the cabin, triggering events dependent on the bucket-cabin contrast). A previous work (Klenk, et al. (2023)) used RGB data jointly with events to constrain the color predictions, which might help here. Additional loss terms constraining flow and regularizing density values throughout the scene might improve these aspects of EvDNeRF performance.

C. E2VID+DNeRF+VID2E

As mentioned in the paper, since our goal is to create an events simulator, it follows that we may explore using VID2E on top of standard DNeRF methods as a competitive baseline. However, we found that VID2E actually generated strong background events from the DNeRF predictions from *E2VID+DNeRF*, as shown in Figure 4. It is possible that regularizing background events or those predicted in static areas of the scene, similar to Klenk, *et al*. (2023), might improve results on this particular baseline, though it would require some tuning.

D. Implementation details

Implementation details. Our code for EvDNeRF is a modified version of the DNeRF open-source code (github.com/albertpumarola/D-NeRF (2021)). As in the original NeRF implementation, two NeRF MLPs for both coarse and fine sampling are used for training EvDNeRF on real data, but results were found to be satisfactory with one NeRF model on simulated data. For paper results, we train every model for 200k iterations on a single Tesla V100 GPU (taking 1-2 days), where each iteration samples 1024 rays, 50% of which are cast through event-triggered pixels and 50% through random pixels. Furthermore, these rays are uniformly selected from all available viewpoints upon each training iteration, which we observed results in a more stable training than selecting all rays from a single viewpoint. Further training details follow.

We implement a learning rate warm-up for the first 1k iterations of training, train only on a cropped portion of event batches centered on high populations of events for 2k iterations, and progressively introduce scene timesteps over the first 20k iterations. This scheduling significantly improved training stability. For varied batching of our supervisory eventstream, we halve the time window of batch sizes twice, once at 100k and again at 150k of training.

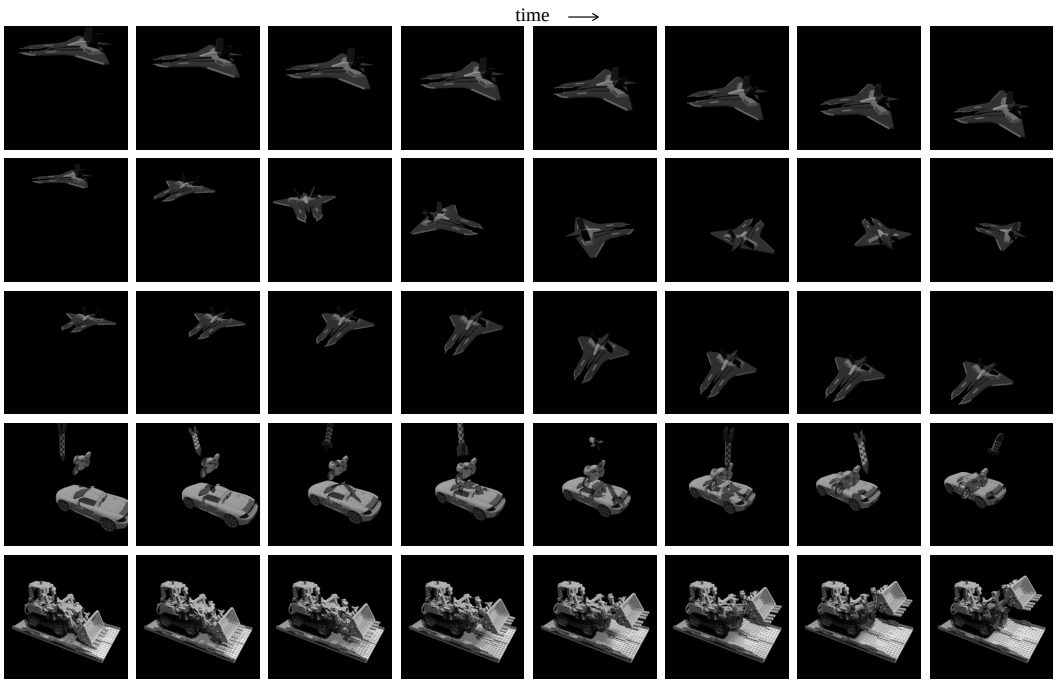

Figure 1. Ground truth intensity image snapshots of all simulated datasets: Jet-Down, Jet-Spiral, Jet-Land, Multi, and Lego; time progresses from left to right.

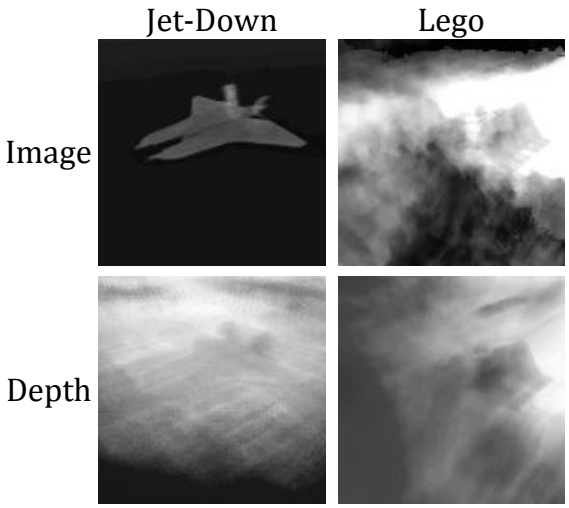

Figure 2. Sample image and depth reconstructions of Jet-Down and Lego datasets, at test viewpoints. Quality of these reconstructions is low, as we expect since we design EvDNeRF to only predict events well. But the objects can be made out, and it is possible that some tuning may improve results. Images were shifted to match ground truth brightness, and depth maps were contrast adjusted.

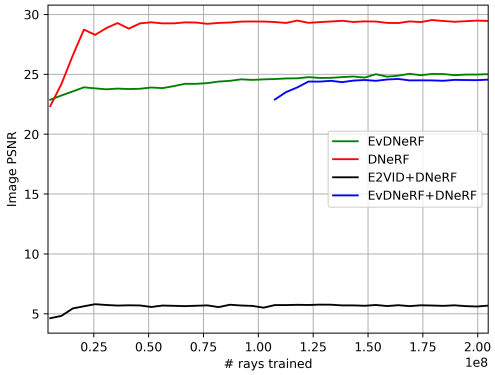

Figure 3. PSNR curves of intensity image generation of our method versus baseline methods, on the Jet-Down dataset (note that these curves are without brightness shifting, whereas tabulated metrics are with brightness adjustment).

E. Real world data

E.1. Notes on time-synchronized real data generation with many views

Generation of real-world data from multi-view event cameras with a consistent motion was a challenging task. While the servo-actuated motion and physical placement of the dynamic object relative to the camera was carefully cal-

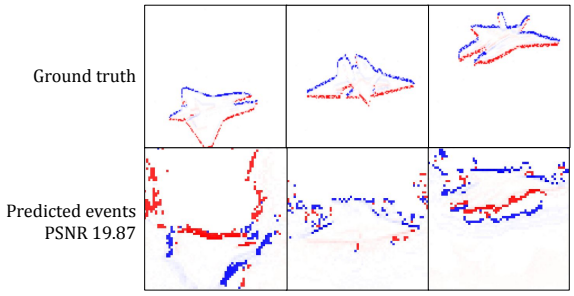

Figure 4. Event reconstructions of the **baseline** *E2VID+DNeRF* but using VID2E to reconstruct events. Since these results are very poor with high levels of background events, we do not present a *E2VID+DNeRF+VID2E* baseline in the paper.

ibrated, it is easily prone to small errors in position or time-synchronization. This is another key benefit of using a neural implicit representation for an events simulator; MLPs can learn smooth functions from noisy input data (within some bounds). This is also why we were able to manually time-synchronize the eventstreams using approximate calculations of motion timestamps (see Figure 3b in paper). Initial attempts to generate real-world data was done with three hardware time-synchronized event cameras, rigidly arranged to be 25° apart from each other, along a circle's arc with 30cm radius to the target. However, as noted in Section 4.3 of the paper, events reconstructions improved as number of views increased, and the front-facing three-view data was not sufficient to constrain the spatial geometry constructed by EvDNeRF. In this case, training views overfitted to achieve low training loss, but intermediate validation viewpoints returned poor reconstructions with multiple hallucinated objects in the renderings (similar to the poor positioning of the Jet in the sample events reconstructions in Figure 9 in the paper). Again, additional density or flow consistency loss terms might improve performance on number-of-views-limited, front-facing datasets.

E.2. Filtering events reconstructions

As mentioned in Results, we filter out low-valued events from real-world reconstructions. Comparisons between original and filtered events are shown in Figure 5. Note that the true, object-triggered events are consistently high valued, and therefore are easy to filter from the background. It's possible that the low levels of background event noise in the Real-Fork dataset cause the consistent levels of background event predictions; however, simple attempts to filter out event noise in the gathered data (via a median filter) caused divergence of EvDNeRF training, likely since we violate brightness consistency of underlying image predictions by manipulating the events in such a way.

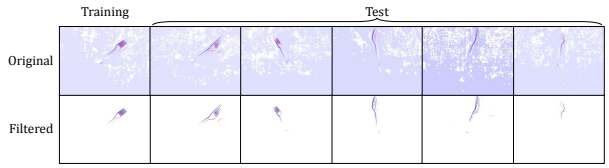

Figure 5. Comparison of original events reconstructions on Real-Fork by EvDNeRF and filtered versions.

F. Additional event reconstruction results

We present additional event reconstructions across test time windows and viewpoints. See Figures 6 - 11.

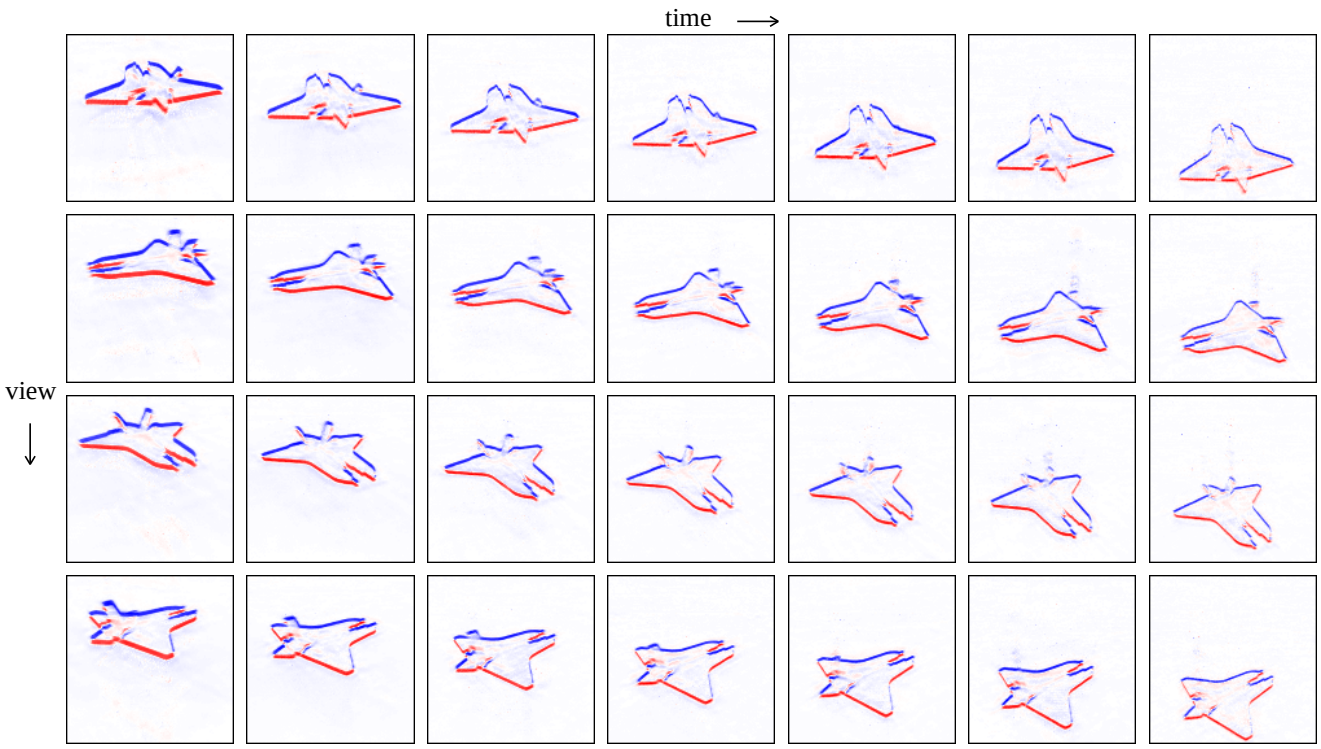

Figure 6. Jet-Down dataset: event reconstructions across time and viewpoint.

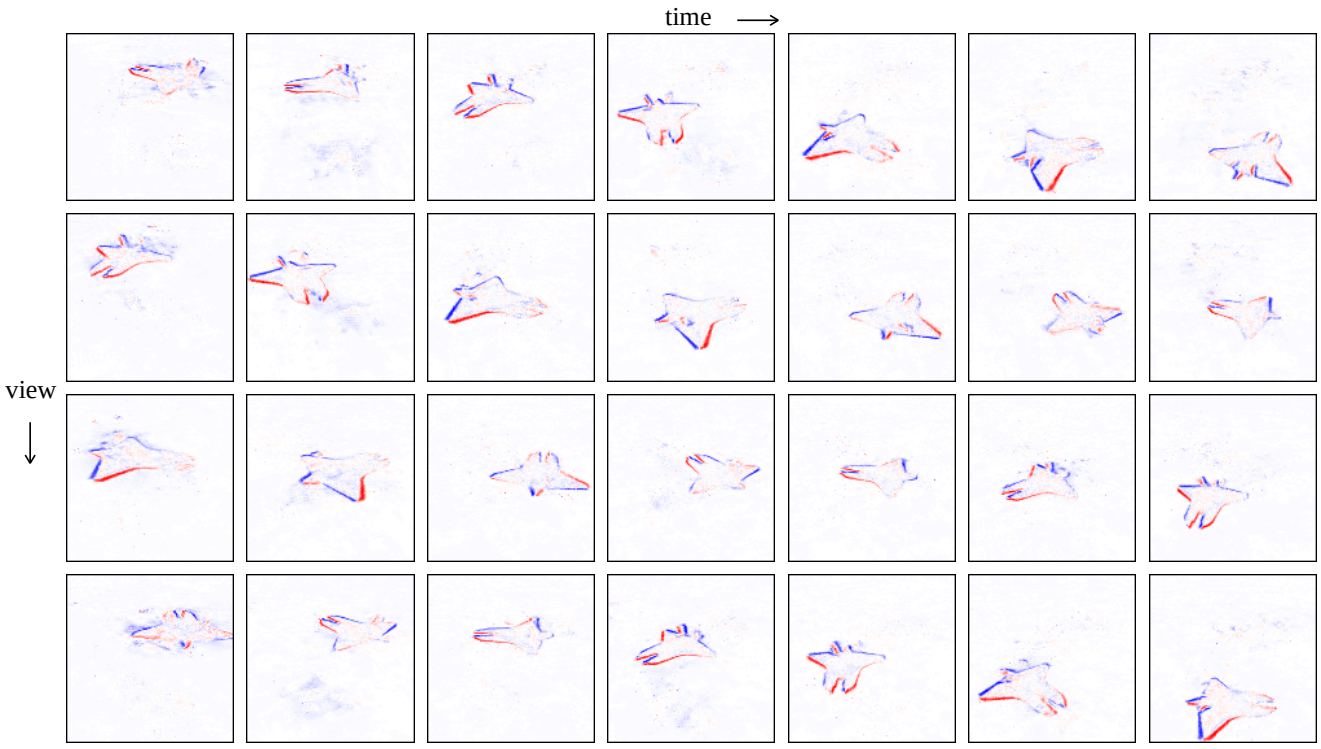

Figure 7. Jet-Spiral dataset: event reconstructions across time and viewpoint.

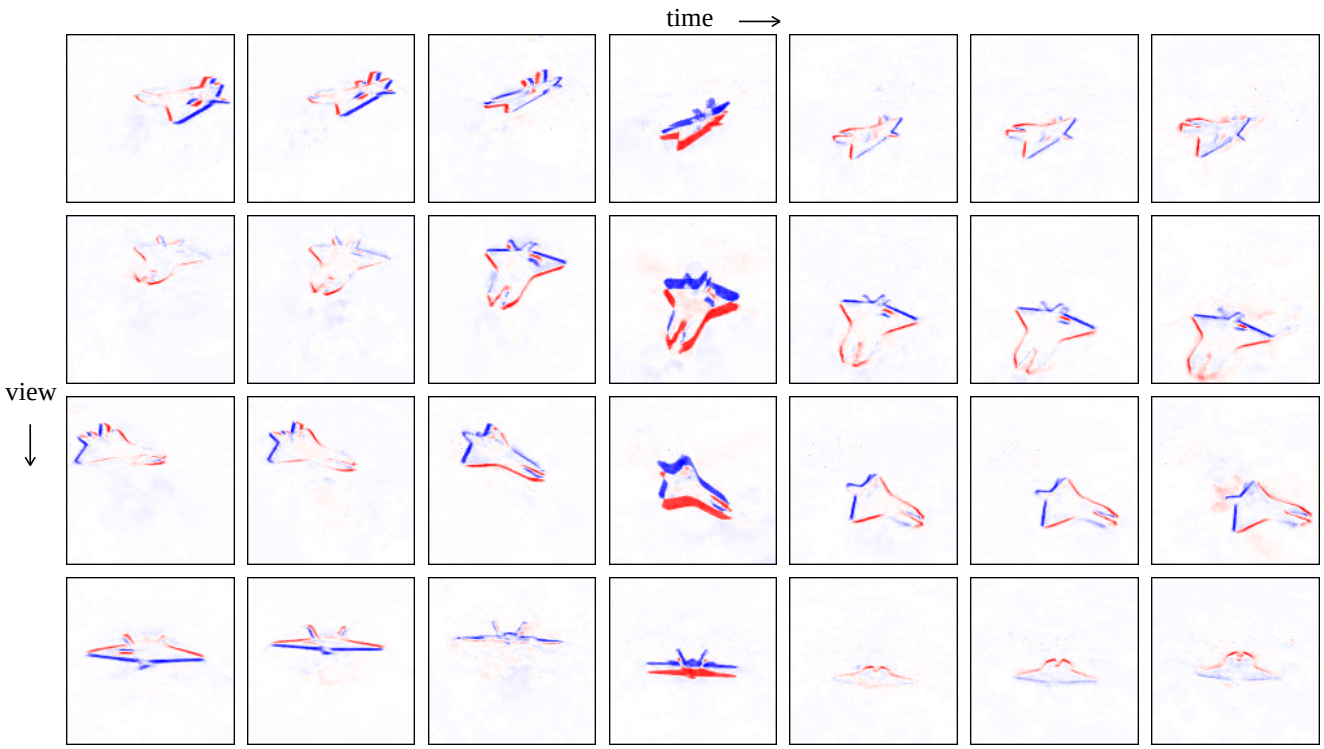

Figure 8. Jet-Land dataset: event reconstructions across time and viewpoint.

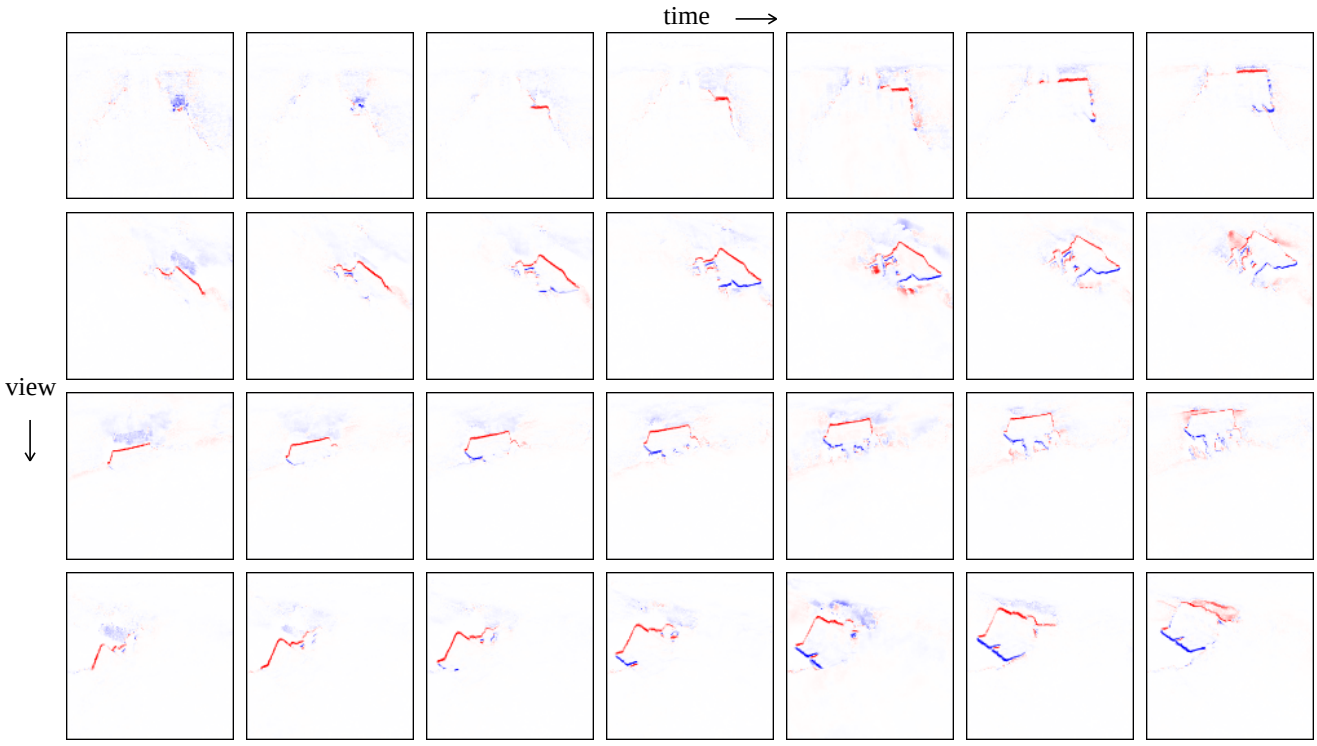

Figure 9. Lego dataset: event reconstructions across time and viewpoint.

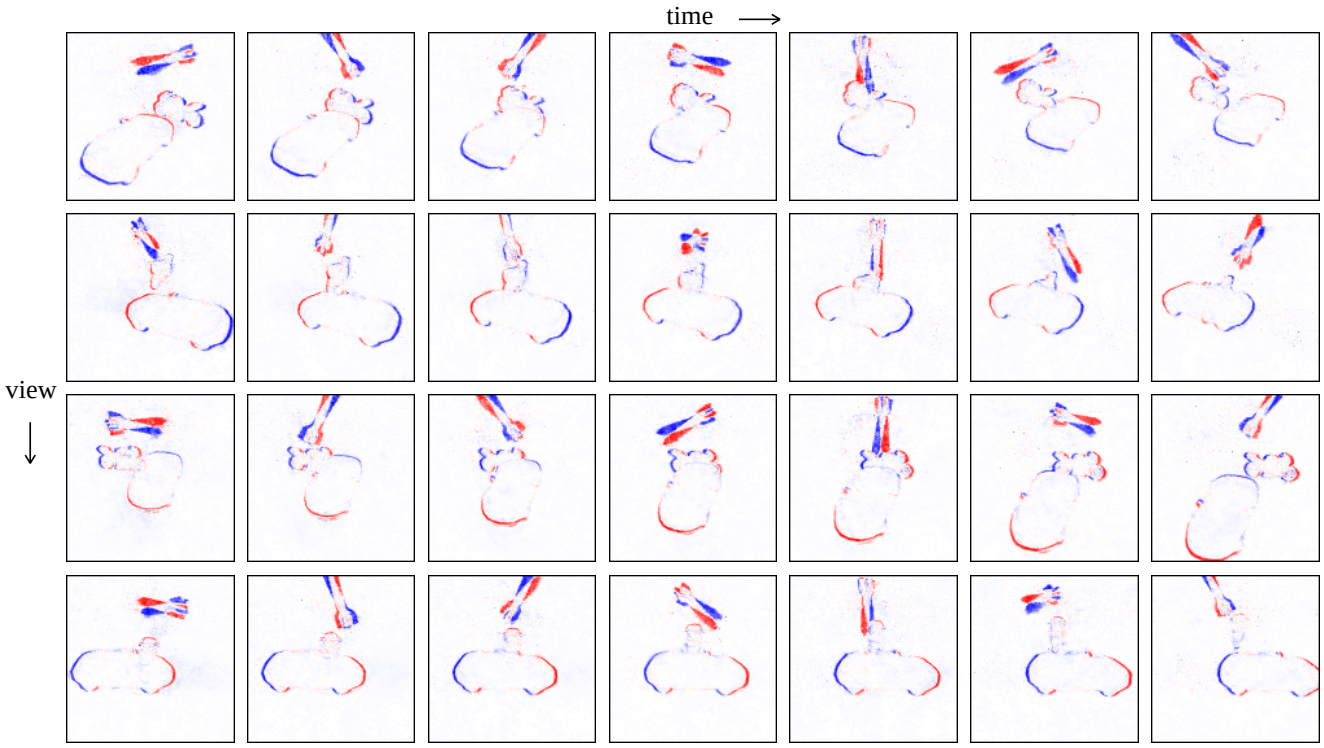

Figure 10. Multi dataset: event reconstructions across time and viewpoint.

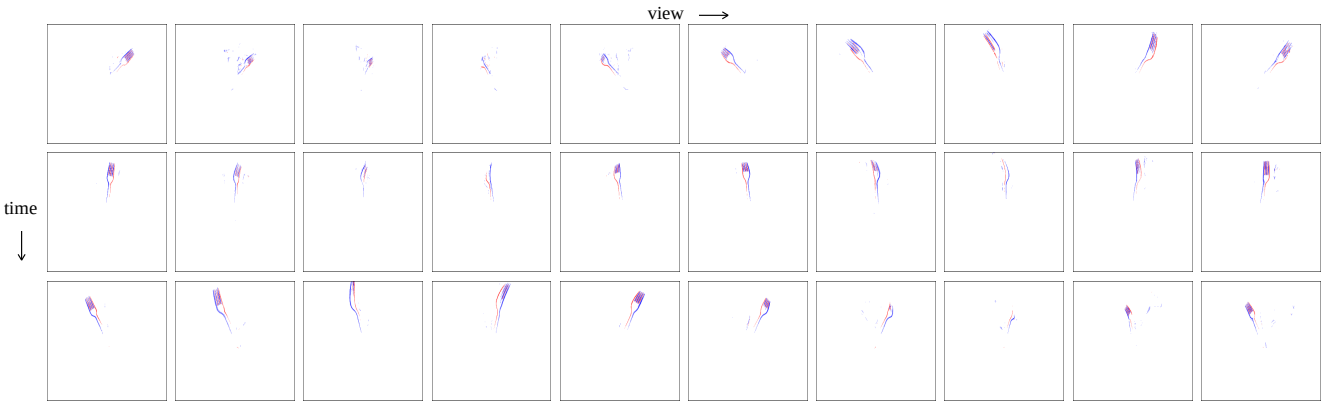

Figure 11. Real-Fork dataset: event reconstructions across time and viewpoint. Note that view and time axes are swapped for this figure.