

Dynamic EventNeRF: Reconstructing General Dynamic Scenes from Multi-view RGB and Event Streams

Supplementary Material



Figure 7. Our portable setup in one of the recording rooms. It consists of six hardware-synchronised iniVation DAVIS 346C colour event cameras on tripods, connected to a workstation with 10 m optic fibre USB3 extension cables. We installed two additional PCIe USB3 extension cards into the workstation to connect all cameras with the required bandwidth.

This supplementary document provides additional experiments and details on calibration, hyperparameters and baselines. We show how we calibrate our event camera response function in Sec. 8. In Sec. 9, we describe the architecture of our MLP network. Next, Sec. 10 includes additional details on the baselines and their hyperparameters. We then describe our datasets and explain the capture process in Sec. 11. We demonstrate the performance of our method with additional ablations on the number of supporting RGB frames (Sec. 12), number of views (Sec. 13), and provide the full per-scene ablation results in Tab. 3.

7. RGB frame quality

As a result of the low-light conditions (see Sec. 5.2), the exposure durations of the RGB frames recorded by the DAVIS camera are rather long, resulting in severe motion blur (see Fig. 8).

8. Camera CRF Calibration

To combine the event generation model with the RGB intensity frames, both obtained through the same lens with DAVIS 346C event cameras, we need a precise measurement of the camera response function (CRF) that we obtain as follows: First, we place the camera in front of a constant brightness light source. Then we use the fact that the amount of captured light is directly proportional to the exposure time. The DAVIS 346 software allows varying the exposure time



Figure 8. RGB frames recorded by the DAVIS camera are very blurry, because low-light conditions require longer exposure times.

at μs precision. Thus by varying it, we can record the relative amount of light needed for the recorded pixel intensity to reach any value from 0 to 255.

We show the raw measurements in Fig. 9a. The CRF is obtained by plotting the RGB values over the exposure, which we show in Fig. 9b. Due to the vignetting of the lens and view-dependent effects, different pixel locations respond differently. Because of that, averaging the values from different pixel locations results in smooth roll-offs at the extremes of the RGB values, which do not correspond to the actual sensor properties. The results show that the measured CRF can be approximated well as a linear function with a vertical shift of $\epsilon = 3 \cdot 10^{-2}$ over the y-axis, indicating that RGB value 0 can be reported even when a non-zero amount of light reaches the sensor.

9. MLP Architecture and Hyperparameters

We inherit the NeRF MLP architecture used in EventNeRF [48], which we also optimise with Adam [18]. However, we modify the model to use the same shared network for both fine and coarse levels of prediction rather than using separate ones. This allows for better optimisation stability and speed, as only half as many parameters are optimised, compared to the original version. Due to the small number of input views in our setting, we modified the code to compose training batches such that they contain an equal number of rays from each view. We found this to increase the stability of the training and the accuracy of the predictions.

	Blender			Dress			Spheres		
Method	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow
TensoRF-CP [7]	24.727	0.879	0.227	28.091	0.915	0.202	23.971	0.866	0.280
NGP [30]	25.687	0.888	0.184	29.131	0.933	0.179	28.755	0.915	0.120
HexPlane [4]	23.119	0.864	0.275	23.580	0.866	0.336	21.636	0.853	0.334
w/o clipping	19.959	0.851	0.426	27.926	0.926	0.147	28.786	0.916	0.093
w/o L_{sparsity}	27.501	0.926	0.135	31.292	0.950	0.081	29.358	0.920	0.086
w/o L_{event}	19.959	0.851	0.426	29.884	0.941	0.108	27.744	0.913	0.114
w/o L_{acc}	26.835	0.899	0.125	30.650	0.946	0.079	28.847	0.920	0.087
w/o L_{RGB}	27.321	0.928	0.122	30.540	0.946	0.086	29.129	0.920	0.089
only L_{event}	27.362	0.905	0.148	31.308	0.951	0.073	29.447	0.921	0.084
only L_{acc}	25.683	0.912	0.159	30.234	0.944	0.106	28.132	0.915	0.107
Full Model	27.431	0.929	0.119	31.396	0.952	0.076	29.765	0.924	0.081

	Lego			Static Lego			Average		
Method	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow
TensoRF-CP [7]	21.444	0.752	0.408	22.604	0.783	0.310	24.167	0.839	0.285
NGP [30]	17.134	0.594	0.556	15.490	0.552	0.598	23.239	0.777	0.327
HexPlane [4]	20.093	0.711	0.452	20.042	0.725	0.423	21.694	0.804	0.364
w/o clipping	22.996	0.814	0.272	22.687	0.814	0.262	24.471	0.864	0.240
w/o L_{sparsity}	22.416	0.798	0.281	23.698	0.839	0.203	26.853	0.887	0.157
w/o L_{event}	21.980	0.792	0.305	22.986	0.821	0.226	24.511	0.864	0.236
w/o L_{acc}	23.178	0.820	0.247	16.057	0.714	0.483	25.113	0.860	0.204
w/o L_{RGB}	22.332	0.799	0.292	23.644	0.836	0.206	26.593	0.886	0.159
only L_{event}	23.009	0.814	0.262	16.500	0.729	0.444	25.525	0.864	0.202
only L_{acc}	22.088	0.794	0.312	22.840	0.819	0.228	25.795	0.877	0.182
Full Model	23.029	0.818	0.258	23.863	0.842	0.193	27.097	0.893	0.145

Table 3. Quantitative ablation and design choice study done with all synthetic scenes. While some ablated models performed better in single scenes, the averaged metrics clearly favour our full model.

10. Baseline Implementation Details

NGP, TensoRF-CP and HexPlane were reimplemented on top of our codebase. For NGP, we used a hash grid implementation in tiny-cuda-nn [29]. For the hash-grid encoding, we used the following configuration:

```
"otype": "HashGrid",
"n_levels": 8,
"n_features_per_level": 2,
"log2_hashmap_size": 19,
"base_resolution": 8,
"per_level_scale": 2.0
```

For the subsequent MLP network, we use two layers with 16 hidden and 10 geometry features. Then for the colour network, we use three layers with 64 hidden features. In total, we train the NGP method for $3 \cdot 10^4$ iterations. The resulting model diverged when training in the sparse-view setting. To significantly improve its sparse-view performance, we annealed the cylinder bound radius from 0 to 100% of the full value in the first 10^4 iterations. Despite that, its sparse-view performance is still lacking compared to the full model and other ablated architectures. TensoRF-CP was reimplemented from scratch in PyTorch. In addition to the usual three spatial dimensions, we also decomposed the temporal dimension, turning it into a spatio-temporal representation. We started with a $16 \times 16 \times 16 \times 16$ grid and gradually progressed towards a $500 \times 500 \times 500 \times 24$ grid in 10 steps throughout

$2 \cdot 10^3$ iterations. We set the factorisation rank to 8 as the highest value that did not cause out-of-memory errors with our NVIDIA A40 GPU. In total, we train the method for 10^4 iterations. Similarly, HexPlane was also reimplemented from scratch. We also used a $500 \times 500 \times 500 \times 24$ grid with a factorisation rank of 8.

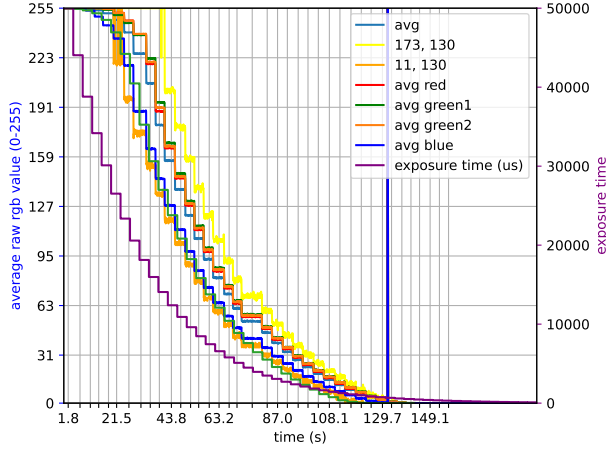
11. Dataset Composition

The proposed synthetic dataset consists of

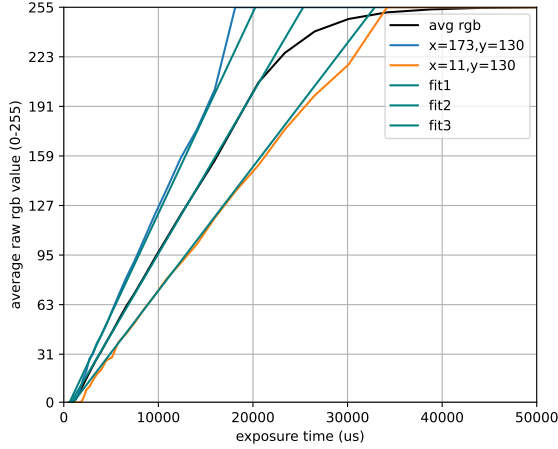
1. Three new original scenes: “Spheres”, “Blender”, “Dress” (licensed CC-BY4.0), and
2. Two scenes that were based on the data provided in [26]: “Lego”, “Static Lego”.

The proposed real dataset contains over 18 min of simultaneous multi-view event and RGB frame streams, recorded on our six event-camera rig described in Sec. 5.2.

We captured ten subjects. Each of them was instructed about the recording and signed the release form for the use of the recorded data in our experiments and subsequent public release. The instructions were as follows: “Enter the recording area. Run around the centre of the area, perform kicks, punches, jumping jacks, and then whatever fast motions you like for a total of a minute. Afterwards, take one of the available objects (towel, ball, bass guitar, paper poster ‘sword’, box, bucket) and perform fast motions for about another minute.”



(a) Raw recorded RGB values when varying the exposure time of the camera at different pixel locations and averaged over the colour channels. There is an outlier on the right of the blue channel average curve, which we ignore during calibration.



(b) Measured CRF and our linear fit. “avg rgb” is the RGB value averaged over all pixels in the view. Lens vignetting results in the values close to white (255) being rolled off. To mitigate this issue, we use RGB values of single pixels instead, labelled as “ $x = \dots, y = \dots$ ” on the plot. “fit1”, “fit2”, and “fit3” indicate our respective linear fits to these curves with $\epsilon = 3 \cdot 10^{-2}$ shift over the y axis (in 0-1 range; corresponds to 7.65 in 0-255 range of the plot).

Figure 9. Event camera RGB intensity frame CRF calibration.

12. Ablation on FPS of Supporting RGB Frames

We ablate the number of supporting RGB frames used for training (Fig. 10 and Tab. 5). There is only a minimal difference in the results if we use only one RGB frame for reconstruction (0.5 FPS), compared to using 100 FPS RGB inputs. This indicates that our method does not depend on the RGB inputs much, using mostly the events. That could lead to a follow-up work that eliminates the RGB inputs.

Method	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow
Ours (TensorRF-CP [7])	26.202	0.784	0.164
Ours (NGP [30])	26.278	0.787	0.163
Ours (HexPlane [4])	24.941	0.758	0.210
w/o clipping	25.288	0.810	0.163
w/o decay	27.119	0.816	0.133
w/o multi-segment	26.673	0.809	0.142
w/o L_{sparsity}	26.920	0.814	0.128
w/o L_{event}	27.620	0.820	0.123
w/o L_{acc}	25.517	0.802	0.139
w/o L_{RGB}	27.062	0.818	0.120
only L_{event}	25.029	0.799	0.143
only L_{acc}	27.754	0.821	0.122
only L_{RGB}	26.019	0.806	0.144
Our Full Model (Final)	27.048	0.819	0.120

Table 4. Quantitative ablation and design choice study on the “Sword” real scene. SSIM and LPIPS metrics favour our full model. In particular, omitting event decay (“w/o decay”) makes SSIM and LPIPS scores worse.

FPS	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow
0.5	26.950	0.817	0.123
1	26.932	0.817	0.123
2	27.055	0.818	0.119
5	27.061	0.820	0.115
10	27.289	0.821	0.117
20	27.226	0.821	0.114
30	27.337	0.821	0.116
50	27.310	0.820	0.117
100	27.328	0.821	0.116

Table 5. Quantitative study on the number of supporting RGB frames. By default, we use 5 FPS, same frequency as our raw data. SSIM and LPIPS metrics favour values above or equal 5 FPS, but do not drop too much with fewer frames, indicating that the model primarily uses event information and not RGB.

13. Ablations on View Count

When reducing the number of training views, we see that the model does not diverge even when using only three views (Fig. 11 and Tab. 6). However, we note that the accuracy of geometry increases significantly when using more views. We also used synthetic data to test more possible setups, up to 36 views. There is clear improvement in PSNR with the increase in the number of training views as well. This confirms that our method indeed benefits from additional improvements to the setup, as stated in the conclusion.

14. Ablation on Design Choices

We provide a detailed ablation study of our design choices in Tab. 3, listing quantitative results for all our synthetic scenes individually. Tab. 3 clearly shows that our full model performs best overall. We also provide a similar table for one of the real scenes in Tab. 4. SSIM and LPIPS also clearly favour the full model in that case.

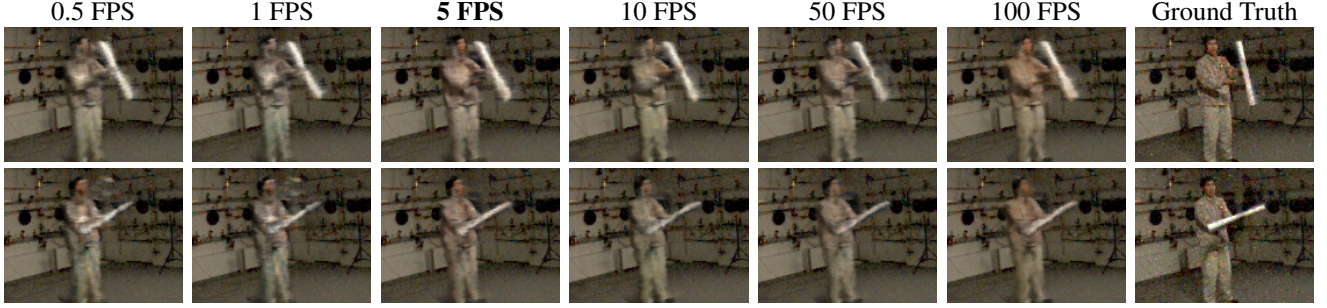


Figure 10. Ablation on the number of supporting RGB images used for training. We show novel views from two different times in two rows; bold indicates the default value. The 0.5 FPS model uses only one set of RGB images. As the number of images increases, there is a slight reduction in artefacts. However, even with one set of images (0.5 FPS), the results are close to the full model (5 FPS). This indicates that the method uses primarily event information and does not rely on the RGB images much.

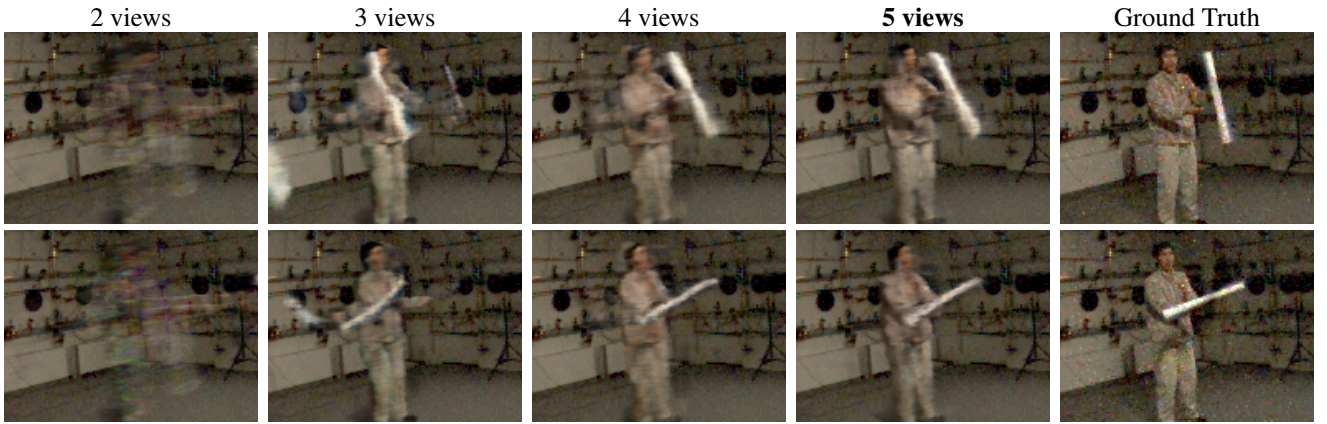


Figure 11. Ablation on the number of views used for training. We show novel views at two different times in two rows; bold indicates the default value. With more input views, the quality indeed gets significantly better.

Input Views	PSNR \uparrow	SSIM \uparrow	LPIPS \downarrow
2	24.667	0.784	0.166
3	26.081	0.808	0.149
4	26.891	0.815	0.118
5	27.061	0.820	0.115

Table 6. Quantitative study on the number of input views on the “Sword” real scene. All metrics clearly confirm the improvement with additional views.

2 v.	3 v.	4 v.	6 v.	12 v.	24 v.	36 v.
16.05	28.97	27.90	30.88	32.19	33.36	33.27

Table 7. PSNR scores on “Dress” with varying number of views. As stated in the conclusion, our method could easily benefit from an increase in the number of views or the resolution of the cameras, which is evident from the results. Bold indicates two best performing model results.

15. Accumulation Stability

We have observed that accumulating long sequences of events leads to unstable results. This means that even in a pixel of constant brightness, spurious “noise” events will accumulate to ever increasing deviations from the true brightness level. In this section we prove the existence of this phenomenon analytically. We also show that our decay approach successfully limits this problem to a tolerable bound.

To see that noise events can destroy all information about the true brightness level, let us consider a pixel of constant brightness, that nevertheless reports noise events with polarities $p_i \in \{-1, +1\}$. For the sake of simplicity we assume that the p_i are independent and identically distributed random variables with fixed, but arbitrary probabilities $q^+ := \mathbb{P}(p_i = +1)$ and $q^- := \mathbb{P}(p_i = -1)$. The accumulated polarity after n such events can then be expressed as the random variable

$$E_n := \sum_{i=1}^n p_i \quad (13)$$

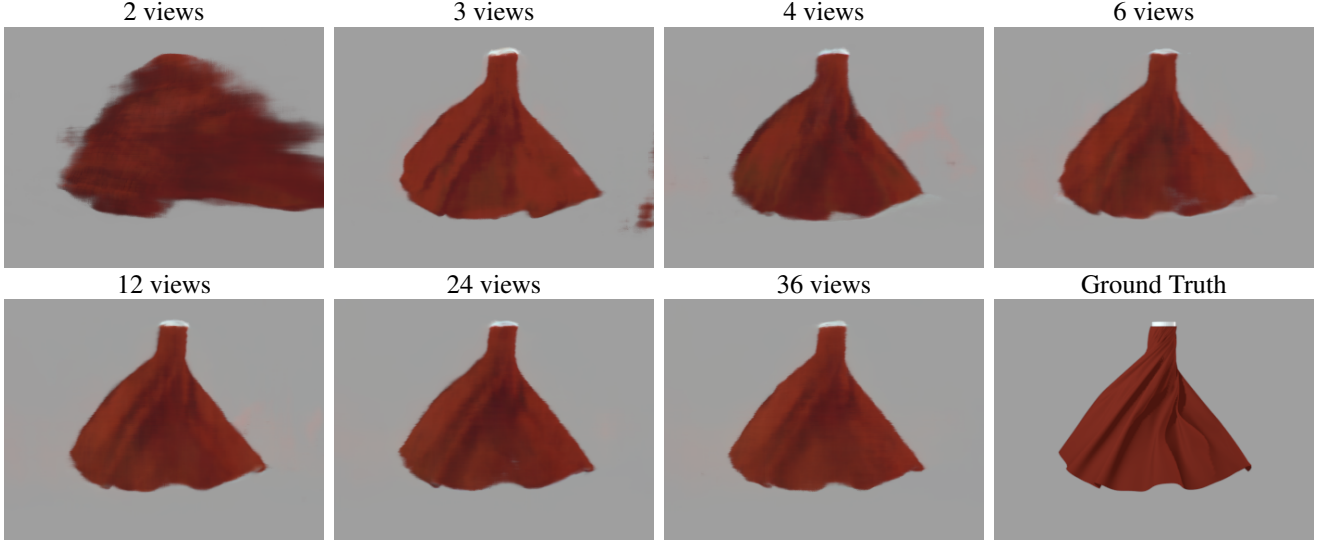


Figure 12. Additional synthetic-data ablation on the number of views used for training. We show novel views at two different times in two rows; bold indicates the default value. With more input views, the quality gets better, albeit hitting diminishing returns at 24 views.

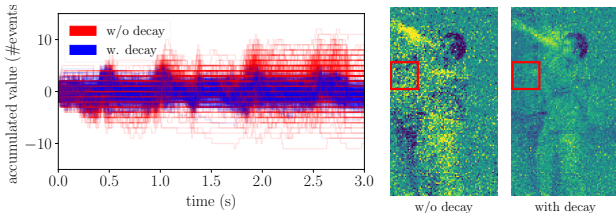


Figure 13. Demonstration of accumulation decay (Sec. 4.4). In a small patch of pixels (marked red on the right), we accumulate events for each pixel individually and show the resulting signals. A naive method (red) becomes unstable, as all pixels have completely different values at the end. In contrast, our proposed method with decay (blue) is stable: all pixels keep similar values at any time. Visually, the image with decay (right) has much fewer artefacts too.

which is a simplified version of Eq. (11). The expected value of E_n is

$$\mathbb{E}(E_n) \stackrel{\text{Def.}\mathbb{E}}{=} \sum_{i=1}^n (+1) \cdot q^+ + (-1) \cdot q^- = n \cdot (q^+ - q^-) \quad (14)$$

If $q^+ \neq q^-$, then $\lim_{n \rightarrow \infty} \mathbb{E}(E_n) = \pm\infty$, *i.e.* as one accumulates more and more noise, one can safely expect to drift arbitrarily far away from the true brightness level. But even when $q^+ = q^-$, which would make $\mathbb{E}(E_n) = 0$ for all n , the *variance* of, and therefore the confidence in the accumulated polarity will decrease with growing n :

$$\begin{aligned} \mathbb{V}(E_n) &\stackrel{\text{Def.}\mathbb{V}}{=} \mathbb{E}((E_n - \mathbb{E}(E_n))^2) \\ &\stackrel{\text{Binom. thm.}}{=} \mathbb{E}(E_n^2 - 2E_n \cdot \mathbb{E}(E_n) + \mathbb{E}(E_n)^2) \\ &\stackrel{\text{Additivity } \mathbb{E}}{=} \mathbb{E}(E_n^2) - \mathbb{E}(E_n)^2 \\ &\stackrel{\text{Def.}E_n}{=} \mathbb{E}\left(\sum_{\substack{1 \leq i \leq n \\ 1 \leq j \leq n}} p_i p_j\right) - \mathbb{E}(E_n)^2 \\ &\stackrel{\text{Arithmetic}}{=} \mathbb{E}\left(\sum_{i=1}^n p_i^2 + 2 \cdot \sum_{1 \leq i < j \leq n} p_i p_j\right) - \mathbb{E}(E_n)^2 \\ &\stackrel{\text{Additivity } \mathbb{E}}{=} \sum_{i=1}^n \mathbb{E}(p_i^2) + 2 \cdot \sum_{1 \leq i < j \leq n} \mathbb{E}(p_i p_j) - \mathbb{E}(E_n)^2 \\ &\stackrel{\text{Def.}\mathbb{E}}{=} n \cdot (q^+ + q^-) \\ &\quad + 2 \cdot \sum_{1 \leq i < j \leq n} q^{+2} - 2q^+ q^- + q^{-2} \\ &\quad - n \cdot (q^+ - q^-)^2 \\ &\stackrel{\text{Binom. thm.}}{\stackrel{\text{Arithmetic}}{=}} 2(q^+ - q^-)^2 \cdot \sum_{i=1}^{n-1} i + 2nq^- \\ &\stackrel{\text{Gau\ss sum}}{\stackrel{\text{Arithmetic}}{=}} n \cdot (n-1) \cdot (q^+ - q^-)^2 + 2nq^- \end{aligned} \quad (15)$$

This term can exceed any arbitrary bound if one lets n grow large enough: If $q^- = 0$, then $q^+ = 1$ and $\mathbb{V}(E_n) = n^2 - n$. If $q^+ - q^- = 0$, then $q^+ = q^- = \frac{1}{2}$ and $\mathbb{V}(E_n) = n$. In all other cases we have $q^- > 0$ and $(q^+ - q^-)^2 > 0$ and thus also $\lim_{n \rightarrow \infty} \mathbb{V}(E_n) = +\infty$

This shows that the noise events will drown out all information about the actual brightness level, if one just waits long enough. Since this effect would deteriorate background pixels (which should not cause any events), but also foreground pixels (where noise is interleaved with legitimate

events), we need to achieve $\lim_{n \rightarrow \infty} \mathbb{V}(E_n) \in \mathbb{R}$ and thus introduce decay:

When accumulating polarities, we value older events less than younger events:

$$\hat{E}_n := \sum_{i=1}^n p_i \cdot b^{n-i} \quad (16)$$

where the decay strength $b := 0.93$ was empirically found to be a useful value.

We now have:

$$\begin{aligned} \mathbb{E}(\hat{E}_n) &\stackrel{\text{Additivity } \mathbb{E}}{=} \sum_{i=1}^n \mathbb{E}(p_i) \cdot b^{n-i} \\ &\stackrel{\text{Def. } \mathbb{E}}{=} (q^+ - q^-) \cdot \sum_{i=1}^n b^{n-i} \\ &\stackrel{\text{Arithmetic}}{=} (q^+ - q^-) \cdot \sum_{i=0}^{n-1} b^i \\ &\stackrel{\text{Geometric sum}}{=} (q^+ - q^-) \cdot \frac{b^n - 1}{b - 1} \end{aligned} \quad (17)$$

Since $\lim_{n \rightarrow \infty} b^n = 0$, we have that $\lim_{n \rightarrow \infty} \mathbb{E}(\hat{E}_n)$ is finite.

In a derivation similar to Eq. (15), we get

$$\begin{aligned} \mathbb{V}(\hat{E}_n) &\stackrel{\text{Def. } \mathbb{V}}{\stackrel{\text{Binom. thm.}}{\stackrel{\text{Additivity } \mathbb{E}}{=}}} \mathbb{E}(\hat{E}_n^2) - \mathbb{E}(\hat{E}_n)^2 \\ &\stackrel{\text{Def. } \hat{E}_n}{\stackrel{\text{Arithmetic}}{\stackrel{\text{Additivity } \mathbb{E}}{=}}} \sum_{i=1}^n \mathbb{E}(p_i^2) \cdot b^{2(n-i)} \\ &\quad + 2 \cdot \sum_{1 \leq i < j \leq n} \mathbb{E}(p_i p_j) \cdot b^{n-i} \cdot b^{n-j} \\ &\quad - \mathbb{E}(\hat{E}_n)^2 \\ &\stackrel{\text{Def. } \mathbb{E}}{\stackrel{\text{Binom. thm.}}{\stackrel{\text{Arithmetic}}{=}}} (q^+ + q^-) \cdot \sum_{i=0}^{n-1} (b^2)^i \\ &\quad + 2 \cdot (q^+ - q^-)^2 \cdot \sum_{i=0}^{n-1} b^i \cdot \sum_{j=0}^{i-1} b^j \\ &\quad - \mathbb{E}(\hat{E}_n)^2 \\ &\stackrel{\text{Geometric sum}}{=} (q^+ + q^-) \cdot \frac{b^{2n} - 1}{b^2 - 1} \\ &\quad + 2 \cdot (q^+ - q^-)^2 \cdot \sum_{i=0}^{n-1} b^i \cdot \frac{b^i - 1}{b - 1} \\ &\quad - \mathbb{E}(\hat{E}_n)^2 \\ &\stackrel{\text{Arithmetic}}{=} (q^+ + q^-) \cdot \frac{b^{2n} - 1}{b^2 - 1} \\ &\quad + 2 \cdot \frac{(q^+ - q^-)^2}{b - 1} \cdot \left(\sum_{i=0}^{n-1} (b^2)^i - \sum_{i=0}^{n-1} b^i \right) \\ &\quad - \mathbb{E}(\hat{E}_n)^2 \\ &\stackrel{\text{Geometric sum}}{=} (q^+ + q^-) \cdot \frac{b^{2n} - 1}{b^2 - 1} \\ &\quad + 2 \cdot \frac{(q^+ - q^-)^2}{b - 1} \cdot \left(\frac{b^{2n} - 1}{b^2 - 1} - \frac{b^n - 1}{b - 1} \right) \\ &\quad - \mathbb{E}(\hat{E}_n)^2 \end{aligned} \quad (18)$$

All summands in the last term of Eq. (18) converge to a finite number as n grows larger, so $\lim_{n \rightarrow \infty} \mathbb{V}(\hat{E}_n) \in \mathbb{R}$. This

shows that decay is effectively bounding the deviation from the true brightness level to a finite error, even for arbitrary numbers of noise events.

Fig. 13 (right) shows that after applying decay, accumulated events become clear, even though they were unrecognisable without decay, proving the effectiveness of our strategy. Additionally, we take a small patch of that view and plot the accumulated values with and without decay in Fig. 13 (left): We see that, as predicted analytically, without decay, the accumulation grows unbounded due to the noise, while with decay, it remains in a constant range, exactly as the signal should be.

The smaller one chooses b , the smaller $\lim_{n \rightarrow \infty} \mathbb{V}(\hat{E}_n)$ becomes and the faster it converges. However, since decay affects not only noise events, but also legitimate ones, we have to explain why it does not distort the actual signal too much. To see this, consider a single pixel: As long as the foreground is not moving through this pixel, it shows a constant level of background brightness. All events it emits are noise, and can safely be suppressed by decay (because the accumulated polarity is supposed to be zero). When the foreground enters the pixel, it will trigger a number of legitimate events. As long as these are recent, the decay will not suppress them too much, so \hat{E} is approximately at foreground brightness level. Assuming that b is chosen suitably, motion is usually fast enough that the object will leave the pixel again before decay “undoes” the entry events completely. Leaving the pixel again triggers a set of legitimate events, that are first sufficiently recent to not be suppressed by damping. As they move further into the past, so do the entry events, so \hat{E} will then correctly approximate the background brightness level again. Of course decay does still negatively impact the legitimate events (especially when motion is occasionally slower than what b was tuned for, such that the foreground is inside the pixel for longer durations), but the underlying MLP, supervised by all our losses, can compensate for that sufficiently. The ablations in Tab. 4 and Fig. 5 show that SSIM scores, LPIPS and visual results are indeed improved by our decay technique.