Mitigating Motion Blur in Neural Radiance Fields with Events and Frames

Marco Cannici and Davide Scaramuzza
Robotics and Perception Group, University of Zurich, Switzerland

Abstract

Neural Radiance Fields (NeRFs) have shown great potential in novel view synthesis. However, they struggle to render sharp images when the data used for training is affected by motion blur. On the other hand, event cameras excel in dynamic scenes as they measure brightness changes with microsecond resolution and are thus only marginally affected by blur. Recent methods attempt to enhance NeRF reconstructions under camera motion by fusing frames and events. However, they face challenges in recovering accurate color content or constrain the NeRF to a set of predefined camera poses, harming reconstruction quality in challenging conditions. This paper proposes a novel formulation addressing these issues by leveraging both model- and learning-based modules. We explicitly model the blur formation process, exploiting the event double integral as an additional model-based prior. Additionally, we model the event-pixel response using an end-to-end learnable response function, allowing our method to adapt to non-idealities in the real event-camera sensor. We show, on synthetic and real data, that the proposed approach outperforms existing deblur NeRFs that use only frames as well as those that combine frames and events by +6.13\,dB and +2.48\,dB, respectively.


1. Introduction

Neural Radiance Fields (NeRFs) [27] have completely revolutionized the field of 3D reconstruction and novel view synthesis, achieving unprecedented levels of details [2, 3, 44]. As a result, they have quickly found applications in many subfields of computer vision and robotics, such as pose estimation and navigation [37, 54, 60], image processing [12, 24, 28, 48], scene understanding [17, 22, 52], surface reconstruction [1, 49, 55], and many others.

Leveraging multi-view consistency from calibrated images, NeRF exploits supervision from multiple view-points, enabling generalization to novel camera poses and the ability to render view-dependent color effects [44]. However, akin to other methods relying on photometric consistency, NeRF can only deliver high-quality reconstructions when the images used for training are perfectly captured and free from any artifact. Unfortunately, perfect conditions are seldom met in the real world.

For example, in robotics, camera motion is prevalent when capturing images, often resulting in motion blur. Under such conditions, NeRFs are unable to reconstruct sharp radiance fields, thereby impeding their practical use in real-world scenes. Although recent works [6, 18, 24, 50] have shown promising results in reconstructing radiance fields from motion-blurred images by learning to infer the camera motion during the exposure time, the task of recovering motion-deblurred NeRFs still remains significantly ill-posed. Existing image-based approaches typically fail when training images exhibit similar and consistent motion [24], and they are inherently limited by the presence of motion ambiguities and loss of texture details that cannot be recovered from blurry images alone.

In this regard, recent works have shown that event-based cameras can substantially aid the task of deblurring images captured with standard cameras [34, 38, 46, 57]. These
Neural Radiance Fields (NeRFs) have gained widespread attention in the research community due to their impressive performance in generating high-quality images from novel viewpoints [8, 40]. As a result, ongoing research is constantly broadening NeRFs range of capabilities, extending their use even under unideal settings. Among these, recent works have tackled the problem of recovering sharp neural radiance fields from blurry images. Deblur-NeRF [24] proposes to simultaneously learn the latent sharp radiance field and a view-dependent blurring kernel, using only blurry images as input. PDRF [6] further extends the approach by employing a coarse-to-fine architecture that exploits additional scene features to guide the blur estimation and speed up convergence, while DP-NeRF [18] improves the motion estimation by imposing rigid motion constraints on all pixels. An alternative approach is BAD-NeRF [50], which directly recovers the camera trajectory within the exposure time, taking inspiration from bundle-adjusted NeRF [21]. Despite impressive results, these methods often fail in the presence of severe camera motion or when the training views share similar motion trajectories, challenging their use with in-the-wild recordings. Our approach has a similar backbone architecture but, crucially, it additionally leverages the advantages of event-based cameras to help the reconstruction of sharp NeRFs. This allows us to recover texture and fine-grained details, resulting in improved performance and higher-quality reconstructions, even in the presence of challenging motion.

Event-based image deblurring In recent years, event-based cameras have become increasingly popular in the field of computational photography [9, 25, 41, 42, 51] due to their high dynamic range and temporal resolution. Several methods have been proposed to exploit the unique characteristics of event cameras for image deblurring, starting from model-based methods, such as the event-based double integral (EDI), which explicitly model the relationship between events triggered during the exposure time and the resulting blurry frame [29, 29]. Subsequent works build on these approaches by refining predictions with learning-based modules [14, 47] or directly learning to deblur the image by fusing events and frames [10, 38, 39, 53, 58]. These networks often pair the deblurring task with that of frame interpolation [10, 39], or make use of attention-based modules to further improve quality [38].

Recently, event-based cameras have also been used to recover sharp images from a fast-moving camera by leveraging an implicit NeRF model of the scene. Ev-NeRF [13], later improved in Robust e-NeRF [23], exploits the event generation model [7] to recover the underlying scene brightness, while EventNeRF [33] extends this approach by incorporating color event-cameras. Recent methods [16, 31] have also explored combining event-based cameras with motion-blurred images. E-NeRF [16] shows that incorporating an event supervision loss can enhance the recovery of sharp edges, but it struggles to restore sharp colors due to
the lack of explicit blur modeling. Similar to ours, E²NeRF [31] follows Deblur-NeRF [24] by modeling the camera motion during the exposure time. Notably, in our approach, we exploit continuous event-by-event supervision and employ a novel learnable camera response function that better adapts to real data, resulting in improved reconstruction under fast motion.

3. Method

The proposed Ev-DeblurNeRF aims to recover a latent sharp representation of the scene given a sequence of timestamped blurry colored images \( \{(C_{u,t}^{\text{blur}}, t_i)\}_{i=1}^{N} \) and events \( E = \{e_j = (u_j, t_j, p_j)\}_{j=1}^{N_E} \), specifying that either an increase or decrease in brightness (as indicated by the polarity \( p_j \in \{-1, 1\} \)) has been detected at a certain time instant \( t_j \) and pixel \( u_j = (u_{j1}, u_{j2}) \). Our method employs recent NeRF-based deblurring modules [6, 18] for fast convergence and adapts them to effectively exploit event-based information. Events in our approach serve a threefold purpose: (i) as sharp brightness supervision obtained through a single integral loss [7, 13, 16, 33], (ii) as prior, in the form of the Double Integral (EDI) [29], and (iii) as a learnable event-based camera response function (CRF) that enables adapting to real event-based data. Fig. 2 provides an overview of our proposed method. In the following section, we introduce the basics of event integrals, while in Sec. 3.2 we describe the building blocks of our network.

3.1. Preliminaries

Event-based Single Integral. Let’s denote the instantaneous intensity at a monochrome pixel \( u \) on a given time \( t \) as \( I(u, t) \). An event \( e_j \) indicates that at time \( t_j \), the logarithmic brightness measured at the pixel location has changed by \( p_j \cdot \Theta_{p_j} \) from the last time \( t_{j-1} \); an event has been generated from the same pixel location. The quantity \( \Theta_{p_j} \in \mathbb{R}^{+} \) is a predefined threshold that controls the sensitivity to brightness changes. It follows that:

\[
\log(I(u, t_j)) - \log(I(u, t_{j-1})) = \Delta L(t_{j-1}, t_j) = p_j \cdot \Theta_{p_j}.
\]

(1)

Considering the events collected in a time period \( \Delta t \) and denoting as \( L(u, t) = \log(I(u, t)) \) the logarithmic intensity, the following relation, called Event-based Single Integral (ESI), holds:

\[
L(t + \Delta t) - L(t) = \Theta \cdot E(t) = \Theta \int_{t}^{t+\Delta t} p \delta(\tau) d\tau,
\]

(2)

where we dropped the dependency from the pixel location and the polarity \( p_j \) in the threshold \( \Theta \) for readability, with \( \delta(\tau) \) an impulse function with unit integral. Besides providing a relation between the difference in instantaneous brightness perceived at two instants and the events captured in between, Equation (4), rewritten as \( I(t + \Delta t) = I(t) \cdot \exp(\Theta \cdot E(t)) \), also introduces a way of warping the instantaneous brightness forward or backward in time using the accumulated brightness \( \Theta \cdot E(t) \) measured by the event camera. This relation is utilized in the following.

Event-based Double Integral. Let’s now recall that the physical image formation process of a standard frame-based camera can be mathematically represented as integrating a sequence of latent sharp images acquired during a fixed exposure time \( \tau \):

\[
I^{\text{blur}}(u, t) = \frac{1}{\tau} \int_{t-\tau/2}^{t+\tau/2} I(u, h) dh, \tag{3}
\]

where \( I^{\text{blur}} \) is the captured image, which we consider affected by motion blur.

Following [29], by combining Equation (4) with (3), we can finally draw a connection between the blurred image observed at time \( t \), the events recorded during the exposure interval \( \Delta T = [t - \tau/2, t + \tau/2] \) and the underlying latent sharp image \( I(u, t) \) at time \( t \):

\[
I^{\text{blur}}(u, t) = \frac{1}{\tau} \int_{t-\tau/2}^{t+\tau/2} \exp(\Theta E(h)) dh. \tag{4}
\]

Solving for \( I(u, t) \), we obtain a model-based deblur of \( I^{\text{blur}} \), guided by the events. In the following, we use this quantity as a prior to supervise our network during training.

3.2. Event-Aided Deblur-NeRF

Our architecture takes inspiration from prior works [6, 18, 24], and is depicted in Figure 2. We aim to recover the scene as a radiance field, implemented by an MLP \( F_{\Omega} \), blindly, by directly modeling the blur formation process at each exposure. Analogous to Equation (3), a blurry color observation generated by the ray \( r(u, t_i) \) cast by pixel \( u \) during its exposure can be described as the integral of the sharp colors observed by the ray in a time interval \( \Delta T_i = [t_i - \tau/2, t_i + \tau/2] \).

Similarly to [18], we learn to estimate the motion of each ray implicitly using a neural module \( G_{\Phi} \). We discretize motion in a finite set of \( M \) observations and learn an \( SE(3) \) field that rigidly warps pixel rays to each position \( q \):

\[
(e^r_q, t_q, w_q) = G_{\Phi}(R(l_i); T(l_i); W(l_i)), \tag{5}
\]

where \( l \in \mathbb{R}^{E} \) is a shared learned image embedding, and \( R, T \) and \( W \) are independent MLPs that predict, respectively, a set of rotation matrices \( e^r_q \in SO(3) \), translation vectors \( t_q \in \mathbb{R}^{3} \), and view weights \( w_q \in \mathbb{R} \), one for each discrete position \( q \). The warped rays can thus be finally obtained as \( \hat{r}_q = e^r_q \cdot r(u, t_i) + t_q \).

Following NeRF [8], we render the color at each ray by first sampling a set of 3D points along each ray, and then query a pair of MLPs, one coarse- and one fine-grained, \( F_{\Omega}^{c} \)
and $F^t_{ij}$, to obtain colors and density at each location. Volumetric rendering is then finally used to estimate colors $\hat{C}_q$ at the predicted camera positions, which are finally fused into a blurry observation

$$\hat{C}_{\text{blur}}(r(u, t)) = g \left( \sum_{q=1}^{M-1} w_q C_q \right), \quad (6)$$

where $g(\cdot)$ is a gamma correction function. Inspired by [18], we further refine the composite weights using an adaptive weight proposal network $\lambda_q = AWP(\zeta_q, I, d_q)$, which takes the ray’s samples features $\zeta_q$, directions $d_q$ and image embedding $I$, to produce refined weights. We use these refined weights in Equation (6) in place of $w_q$ to obtain refined colors $C_{\text{blur}}$.

The thus rendered synthetic blurry pixel is finally supervised with a ground truth observation $C_{\text{gt}}$ through:

$$E^b(\hat{C}_r) = \| C_{\text{blur}} - C_{\text{gt}} \|_2^2, \quad (7)$$

$$\mathcal{L}_{\text{blur}} = \frac{1}{|R_b|} \sum_{r \in R_b} E^b(\hat{C}_r) + E^b(\hat{C}_{\text{blur}}) + E^b(\hat{C}_{\text{blur}}), \quad (8)$$

where we consider a batch of pixels $R_b$, and rewrite $C_{\text{blur}} = C_{\text{blur}}(r)$. Subscripts $c$ and $f$ indicate if values are obtained through $F^c_{ij}$ or $F^f_{ij}$, while $\sim$ if adaptive weights are used.

**Event-based supervision via learned event-CRF.** When the scene is also captured by an event-based camera, as in our case, blur-free microsecond-level measurements can be exploited to further assist the reconstruction of a sharp radiance field, leveraging the relation in Equation (1) between brightness and generated events. We do so by synthesizing the left-hand side of (1), i.e., the log brightness difference perceived by the event pixel, through volumetric rendering while we take the right-hand side as a ground truth supervision, given recorded event pairs.

In particular, we estimate the log-brightness at each event $e_j$, produced by the event pixel $u$ at time $t_j$, as:

$$\hat{L}(t_j, u) = \log(h(eCRF_q(\hat{C}(r_j), p_j))), \quad (9)$$

where we obtain $\hat{C}(r_j)$ via volumetric rendering [27] by rendering the ray $r_j = r(u, t_j)$ cast from the camera pose $T(t_j) \in SE(3)$, approximated via spherical linear interpolation [35] of the available known camera poses. Here, $eCRF_q$ is an MLP that produces a modulated signal $\hat{C}_q \in \mathbb{R}^3$ from the rendered color $C$ and the polarity $p_j$, while $h(\cdot)$ is a luma conversion function, implemented following the BT.601 [43] standard.

Given a pair of consecutive events at time $t_{j-1}$ and $t_j$, we first estimate the log-brightness difference as $\Delta \hat{L}(u, t_j) = \hat{L}(u, t_j) - \hat{L}(u, t_{j-1})$ and then compare it with that observed by the event camera, $\Delta L$, as follows:

$$E^e(\Delta \hat{L}^t_{ij}) = \| \Delta \hat{L}^t_{ij} - \Delta L^t_{ij} \|_2^2, \quad (10)$$

$$\mathcal{L}_{ev} = \frac{1}{|\mathcal{U}|} \sum_{(u, t) \in \mathcal{U}} E^e(\Delta \hat{L}^t_{ij}) + E^e(\Delta \hat{L}^t_{ij}) + E^e(\Delta \hat{L}^t_{ij}), \quad (11)$$

where we use the compact form $\hat{L}^t_{ij}$ for $\hat{L}(t, u)$, and apply the supervision on fine and coarse levels, as well as on adaptively refined colors. $\mathcal{U}$ selects pairs of pixels $u$ and timestamps $t$ corresponding to received events.

Figure 2. Architecture of the proposed Ev-DeblurNeRF model. For each given ray $r(u, t)$, placed at the center of the exposure time $\tau$, we estimate a set of warped rays $r_q$ using $G_F$. We then sample features from an explicit volume $\mathcal{V}$ and fed these features to $F_\Phi$ to compute blurry colors through weighted averaging with $L_{\text{blur}}$. We supervise the color at the center of the exposure time through $L_{\text{EDI}}$ by recovering a prior-based sharp color using the event double integral, considering all events in the exposure time. Finally, we sample a pair of two consecutive events, and supervise their brightness difference, modulated by $eCRF$, using the observed polarity value via $L_{\text{ev}}$. 


reveal that applying $L_{ev}$ not only during image exposures but also between frames, similar to [16], helps in viewpoints with scarce RGB coverage, as common with fast motion.

In Equation (11), we assume the ideal event generation model of (2). However, real event pixels deviate from the ideal case [11]. Our proposed event CRF function $cCRF_{ev}$ learns to compensate for potential mismatches between the ideal model and that of the camera at hand, filling the gap between the rendered color space and the brightness change perceived by the event sensor. Note that, when a color event camera is used, as the one in [19], pixels record color intensity changes following a Bayer pattern. We remove the luma component prior to the color channel each pixel is responsible for. We refer to this version of the loss as $L_{ev-color}$.

**Double integral supervision.** The eCRF just introduced provides an effective way of handling unmodeled event pixel behaviors. However, blindly recovering the event camera response to colors is not trivial since the only direct source of color supervision comes from Equation (8). In practice, the optimization problem in (11) is under constrained, as the loss, acting on the event camera, is free to enhance texture details in the radiance field as long as they correctly render once blurred through Equation (3). Inspired by recent works [20], which suggest facilitating NeRF optimization through priors, we propose here to exploit the relationship in (4) to further constrain the NeRF training.

In particular, we consider every original ray $r \in \mathcal{R}_b$ sampled when optimizing Eq. (8) originating from the mid-exposure pose of image $I_t^{blur}$, i.e., the rays rendering the latent sharp pixels $C(r)$. If we simplify and assume these pixels are monochrome, they correspond to $I_t$ in Equation (4). Given this observation, we first rewrite (4) by solving for $I_t(u, t_i)$, and then evaluate it channel-wise for the given image at time $t_i$ and ray $u$, using the observed blurry color $C^{blur}$ and the events received at pixel $u$. We finally collect channels into $C_{r}^{EDI} = \left[ I_{r}^{R}(u, t_i), I_{r}^{G}(u, t_i), I_{r}^{B}(u, t_i) \right]$, obtaining a model-based sharp latent color. We use this color as a prior in:

$$f_{s,k}^c = \mathcal{V}_s(x_k^c), \quad f_{s,k}^f = \mathcal{V}_s(x_k^f),$$

with $\mathcal{V}_s$ and $\mathcal{V}_f$, respectively, a small and a large TensoRF volume. We use $f_{s,k}^c$ as additional features in $F_{\Omega}^s$, while we employ all the features as input to the fine-grained MLP $F_{\Omega}^f$. The structure of $F_{\Omega}^s$ and $F_{\Omega}^f$ is analogous to that of the original NeRF [27], with the only difference that the MLP predicting $\sigma$ also takes these extra features as input.

4. Experiments

4.1. Implementation Details.

**Training.** We build our event-based architecture starting from the Pytorch implementation of DP-NeRF [18]. We use a batch size of 1024 for rays $\mathcal{R}_b$ and 2048 for rays $U_c$, and sample 64 coarse and additional 64 fine points along each ray. Following [6], we set the number of motion locations to $M = 9$. We use Adam [15] to optimize the multi-objective loss $L = \lambda_b L_{blur} + \lambda_e L_{event} + \lambda_{EDI} L_{EDI}$, where we set $\lambda_b = 1$, and $\lambda_e = 0.1$. We train the model for a total of 30,000 iterations, using an initial learning rate of $5 \cdot 10^{-3}$, which we decrease exponentially to $5 \cdot 10^{-6}$ over the course of the training. Further details on the network architectures are provided in the supplementary material.

**Ev-DeblurBlender dataset.** We evaluate our method on four synthetic scenes derived from the original DeblurNeRF [24] work, namely, factory, pool, tanabata, and trolley. We exclude cozy room from our conversion as the Blender rendering for this scene relies on an image denoising post-processing step. This step causes the rendered images to show temporally inconsistent artifacts when rendered at high FPS, thereby causing unrealistic event simulation. Differently from [24], where blurry images are obtained by randomly moving the camera at each pose, we use a single fast continuous motion, derived from DeblurNeRF’s original poses, lasting around 1s. We simulate a 40s exposure time by averaging together, in linear RGB space, images rendered at 1000 FPS. We then use the same set of images to generate synthetic events using event simulation [32], making use of a balanced $\Theta = 0.2$ event threshold and monochrome events.

**Ev-DeblurCDAVIS dataset.** Given the lack of real-world datasets for event-based NeRF deblur that incorpo-
rate ground truth sharp reference images for quantitative assessment, we introduce a novel dataset composed of 5 real-world scenes. We use the Color-DAVIS346 [19] camera for recording, which captures both color events and standard frames at 346 × 260 pixel resolution using a RGBG Bayer pattern. We mount the camera on a motor-controlled linear slider to capture frontal-facing scenes and use the motor encoder to obtain poses at 100 Hz. We configure the Color-DAVIS346 with a 100ms exposure time and collect ground truth still images first, followed by a fast motion. Scenes feature 11 to 18 blur training views and 5 ground truth sharp poses with both seen and unseen views.

**Baselines.** We evaluate our method against frame-only methods as well as methods fusing both images and events. For the first category we follow previous works [18, 18, 24], and select Deblur-NeRF [24], BAD-NeRF [50], DP-NeRF [18] and PDRF [6] as the most recent NeRF-based baselines, as well as single-image and video deblurring methods, namely MPRNet [56] and PVDNet [36], followed by NeRF [27]. Similarly, for the second category, we select E-NeRF [16] and E²-NeRF as event-based deblur NeRF architectures, and also combine frames deblurred via the events+frames EFNet [38] network with NeRF [27]. We run all baselines with default hyperparameters using the official codebases. We utilize Blender poses in Ev-DeblurBlender and motor encoder poses in Ev-DeblurCADAVIS for all baselines, including E²-NeRF, where we compute exposure poses via spherical linear interpolation of the available ones.

### 4.2. Experimental Validation

**Results on Ev-DeblurBlender.** We start the evaluation on the synthetic Ev-DeblurBlender dataset to first assess the performance of our method on an ideal case, i.e., where camera poses are accurate and the event generation model is close to the ideal case. Results are reported in Table 1. We test two versions of our network. The first, which we call Ev-DeblurNeRF- - , does not make use of the proposed eCRF module and EDI supervision, while the second, Ev-DeblurNeRF, incorporates the complete architecture presented in Section 3. We found Ev-DeblurNeRF- - to exhibit slightly superior performance on average on this data. As discussed in Section 3, indeed, we designed the eCRF specifically to handle possible variations between RGB and events’ response functions, as well as to compensate for mismatches on the event generation model. These issues are not predominant in simulated data, explaining why adding a learnable response function does not improve performance.

Despite this, both versions largely outperform all other baselines, both event-based and frame-based. Compared to DP-NeRF [18], which uses a similar backbone architecture, our method achieves on average a +3.34dB higher PSNR, a 52.9% lower LPIPS [59] and 7.14% higher SSIM, highlighting the improvement gained by effectively integrating event-based supervision. This is also evident when considering baselines utilizing an image-deblurring stage prior to NeRF training, which also achieve better performance when events are used. This is the case of EFNet [38], and its variant, which we name EFNet*, that we fine-tune on the other 3 scenes before deblurring images of a given scene. Despite the high accuracy, these methods fail to produce scene-level consistent deblurring, causing the NeRF to reconstruct floaters and thus decreasing novel-view synthesis performance. Finally, our approach also surpasses both previous event-based deblurring NeRF methods with an average increase of +5dB in PSNR, a 50% reduction in LPIPS, and a 16.7% increase in SSIM. Notably, ENeRF, which does
not explicitly model the blur formation process, struggles to recover sharp color information, while E$^2$NeRF, exclusively employing event supervision during the exposure time, fails at fully exploiting event-based data. Our method, on the contrary, overcomes both limitations, showcasing the effectiveness of the proposed approach.

**Results on **Ev-DeblurCDAVIS. In Table 2, we report results obtained on data collected with a real ColorCDAVIS346 camera. We select the top-performing NeRF models from the previous evaluation, namely E$^2$NeRF [31] and DP-NeRF [18], which we modify here by integrating the TensoRF modules discussed in Section 3 for a better comparison. Additionally, we include the performance metrics obtained by initially deblurring images using the model-based EDI deblur method, followed by NeRF. An extended analysis including all other baselines is provided in the supplementary materials. Once again, our proposed approach significantly outperforms all baselines, exhibiting an improvement of +2.5dB in PSNR and a 4.6% increase in SSIM. A qualitative comparison, depicted in Figure 3, illustrates the capability of the proposed Ev-DeblurNeRF network in reconstructing textures and details, ultimately resulting in a higher-quality novel view synthesis.

**Synthesis from sparse blurry views.** Utilizing the same setup used for collecting the Ev-DeblurCDAVIS dataset, we study here the robustness of the proposed approach to sparse supervision to highlight the advantage of using events not only within exposure but also in between frames. We collect an additional, longer, sequence with a back-and-forth motion and train the proposed approach with an increasing number of frames $N_f \in \{5, 9, 17, 33\}$, such that each set is a subset of the next and making sure that test poses are within training views but as furthest away as possible from them. Results are reported in Figure 4. Remarkably, Ev-DeblurNeRF attains the highest performance of all methods we tested, with its performance only decreasing by 3.46dB in PSNR when passing from 33 to just 5 views. In contrast, E$^2$NeRF and EDI+NeRF experience a decrease of 13.71dB and 15dB, respectively. These methods struggle to correctly reconstruct the radiance field from viewpoints that are only weakly supervised by blurred images. Our approach, instead, is only marginally affected. More details are provided in the supplementary material.

**Robustness to motion blur.** In Figure 4, we analyze how the performance of the proposed approach changes as we vary the motion blur intensity. We follow the same setup as before but this time vary the slider speed from 0.1m/s to
Ablations. We conclude the evaluation by studying, in Table 3, the contribution of all the modules introduced in Section 3, using a scene derived from the Figures sample of Ev-DeblurCDAVIS. Adding event supervision from Equation (11) improves PSNR by +0.69dB, which is further increased by +1.04dB when the events’ color channel is considered. Similarly, adding $\mathcal{L}_{EDI}$ in Equation 13 as well as the proposed eCRF module, with and without additional polarity features, also results in increased performance. Next, we study the contribution of adding the $\mathcal{L}_{EDI}$ in Equation 13 and the eCRF module. Performance increases in both cases, with a +3.15dB increase when adding $\mathcal{L}_{EDI}$ and a +1.49dB when adding the eCRF. The highest performance is achieved when both are combined and when the eCRF also utilizes polarity as input, with an increase of +0.74dB, and an overall improvement of +5.62dB in PSNR with respect to only using images. We finally validate the use $\mathcal{V}_{c,f}$ on the full configuration. Using explicit features guarantees faster training times without sacrificing performance. We obtain a slight boost in PSNR and LPIPS but, most notably, a $\times 10.8$ speedup in training convergence. This model only takes around 3 hours and 30 minutes for training on an NVIDIA A100 GPU, while the same network without $\mathcal{V}_{c,f}$ takes around 38 hours on the same hardware, as it requires more iterations at a lower learning rate. Moreover, in comparison to $\text{E}^2_{\text{NeRF}}$, which takes around 24 hours to train, our model is 6.9 times faster.

Limitations. We structure the proposed Ev-DeblurNeRF assuming that events and frames can be recorded from the same image sensor. While this is possible with the suggested hardware, namely a ColorDAVIS camera, not all event cameras feature both modalities. While the proposed $\mathcal{L}_{EDI}$ loss requires pixel alignment to work effectively, we believe the proposed method could still be applied in more advanced stereo setups, such as the ones in [25, 42], especially exploiting the proposed eCRF to compensate for different sensor responses. Moreover, our method, similar to [16], estimates event camera poses via interpolation of available ones. This could lead to a performance decrease in case estimated poses are far from actual ones or they are provided at a low frequency. However, we believe refinement of camera poses through event-based methods [26, 45], or a modified approach that only computes $\mathcal{L}_{\text{event}}$ at known camera views, could help in mitigating this issue.

5. Conclusions

We present Ev-DeblurNeRF, a novel deblur NeRF architecture that fuses blurry frames with events for sharp NeRF recovery. Our method, exploiting explicit features for fast training convergence, integrates a learnable event-based camera response function and ad-hoc event-based supervision that facilitates fine-grained details recovery. Ev-DeblurNeRF, despite being supervised by model-based priors, can adapt to non-idealities in the camera response, potentially departing from the model-based solution. We validate our method on both synthetic and real data, achieving an increase of +4.42dB and +2.48dB in PSNR, respectively, when compared to the previous best-performing event-based baseline, and an increase of +2.74dB and +6.13dB when compared to the top-performing image-only baseline.

6. Acknowledgements.

This work was supported by the National Centre of Competence in Research (NCCR) Robotics (grant agreement No. 51NF40-185543) through the Swiss National Science Foundation (SNSF), and the European Research Council (ERC) under grant agreement No. 864042 (AGILEFLIGHT).
Table 4. Extended quantitative comparison on the real-world Ev-DeblurCDAVIS dataset. Best results are reported in bold.

<table>
<thead>
<tr>
<th></th>
<th>BATTERIES</th>
<th>POWER SUPPLIES</th>
<th>LAB EQUIPMENT</th>
<th>DRONES</th>
<th>FIGURES</th>
<th>AVERAGE</th>
</tr>
</thead>
<tbody>
<tr>
<td>BAD-NeRF* [59]</td>
<td>27.32</td>
<td>0.26</td>
<td>0.82</td>
<td>26.42</td>
<td>0.32</td>
<td>0.79</td>
</tr>
<tr>
<td>DP-NeRF [18] + TensoRF [5]</td>
<td>28.64</td>
<td>0.27</td>
<td>0.81</td>
<td>25.74</td>
<td>0.32</td>
<td>0.77</td>
</tr>
<tr>
<td>PDRF [6]</td>
<td>28.62</td>
<td>0.25</td>
<td>0.81</td>
<td>25.79</td>
<td>0.31</td>
<td>0.77</td>
</tr>
<tr>
<td>MP-VDRN [36] + NeRF</td>
<td>27.99</td>
<td>0.21</td>
<td>0.81</td>
<td>26.80</td>
<td>0.23</td>
<td>0.78</td>
</tr>
<tr>
<td>EFNet [39] + NeRF</td>
<td>29.85</td>
<td>0.15</td>
<td>0.88</td>
<td>30.28</td>
<td>0.18</td>
<td>0.88</td>
</tr>
<tr>
<td>EDG [29] + NeRF</td>
<td>28.66</td>
<td>0.12</td>
<td>0.87</td>
<td>28.16</td>
<td>0.09</td>
<td>0.88</td>
</tr>
<tr>
<td>ENVRF [18]</td>
<td>27.83</td>
<td>0.26</td>
<td>0.74</td>
<td>27.91</td>
<td>0.21</td>
<td>0.76</td>
</tr>
<tr>
<td>E2-neRF [11]</td>
<td>30.57</td>
<td>0.12</td>
<td>0.88</td>
<td>29.98</td>
<td>0.11</td>
<td>0.87</td>
</tr>
<tr>
<td>(Ours) Ev-DeblurNeRF</td>
<td>33.17</td>
<td>0.05</td>
<td>0.92</td>
<td>32.35</td>
<td>0.06</td>
<td>0.91</td>
</tr>
</tbody>
</table>

Figure 5. Qualitative ablation study of the main components of the proposed Ev-DeblurNeRF network. Tables below each picture are drawn from Table 3 of the paper, and report the configuration used and the PSNR metric achieved in each case.

A. Implementation Details

Training. We implement Ev-DeblurNeRF building upon the DP-NeRF [18] official codebase implemented in PyTorch [30], and incorporating additional features from PDRF [6] and TensoRF [5]. We train both our Ev-DeblurNeRF network and the baselines on full-resolution images using either an NVIDIA V100, an NVIDIA RTX A6000, or an NVIDIA A100 GPU. In particular, we use 600 × 400 images for Ev-DeblurBlender and 346 × 260 for Ev-DeblurCDAVIS. Similar to [6, 18, 24], we warm up the training for the first 1, 200 iterations, by using at first only the Lc,EDI and Lc,w losses and without utilizing the eCRF module. Subsequently, we introduce Lblur, along with the proposed eCRF, which we initialize as the identity function, and the blur estimation module GΦ, keeping the λ parameters (λb = λEDI = 1, and λc = 0.1) fixed for the entire duration of the training. To implement LEDI, we pre-compute CEDI images using Eq. (4) and directly sample them during training. When using the Lc,w loss, we weigh the events’ contributions by 0.4, 0.2, or 0.4 depending on whether the event corresponds to a red, green, or blue channel, as green pixels appear twice as often in an RGB Bayer pattern. We use symmetric constant thresholds for the events, setting Θ = 0.2 for synthetic events, and Θ = 0.25 when using a real camera.

Architecture. The motion estimation module GΦ is implemented following DP-NeRF [18] hyperparameters’ choice, and using M = 9 exposure poses. Differently from [18], we implement the image embedding Ii using a simple set of learnable 32-dimensional parameters, instead of predicting them through an additional 4-layers MLP. We found this design to be easier to optimize and yield overall better results. We follow [18] to implement the refinement AWP module and employ the coarse-to-fine scheduling strategy to weight Cblur and Cblur in Lblur. However, we weigh their contribution equally in Lcv through the whole training, as we found the coarse-to-fine scheduling strategy not to improve the results. We implement CFI as a 2-layers MLP with ReLU activation, hidden dimension 64, and output dimension 16, followed by a 3-layers MLP with the same activation and hidden dimension, but output dimension 3. We use one of the output channels of the first MLP as the predicted density, while the rest is used by the second MLP to predict colors. The structure of CF is analogous, but we use an output dimension of 128 for the first MLP and a 256 hidden dimension for both MLPs. We implement Ψ and Ψ with vector-matrix decomposition [5], using 16.7 million voxels in Ψ and 134.2 million voxels in Ψ, and setting to {64, 16, 16} the channel dimensions of the decomposed {X, Y, Z} axes in both Ψ and Ψ. The proposed Ev-DeblurNeRF architecture trains in around 3 hours and 30 minutes on an NVIDIA A100 GPU.

B. Extended Analysis on Ev-DeblurCDAVIS

State-of-the-art comparison. Section 4.2 of the paper provides an analysis on the Ev-DeblurCDAVIS dataset focused on the top-performing architectures selected from the synthetic evaluation. For completeness, we report in Table 4 of this supplementary material a comprehensive evaluation against all other baselines used in the paper. The trend follows that of the synthetic analysis, with image-only baselines performing worse than networks making use of
either images deblurred through events or event-enhanced NeRFs. Notice that, we could not finetune EFNet [38] on Ev-DeblurCDAVIS as, differently from simulation, we do not have corresponding sharp images for each blurry training view. We designed the Ev-DeblurCDAVIS dataset in such a way as to ensure reliable ground truth collection, but also to showcase the ability of our network to tackle a known limitation of image-only DeblurNeRF-like architectures. While these networks work particularly well on random motion patterns, they fail in the presence of consistent blur, i.e., when the motion pattern is similar in each exposure. This is the case of Ev-DeblurCDAVIS, where image-only baselines such as DP-NeRF [18] and PDRF [6] struggle to remove blur (see Figures 5 and 9 of this supplementary material and Figure 3 of the paper). For similar reasons, BAD-NeRF diverges after a few training iterations on this dataset. We address this by fixing the rotation matrix to ground truth and optimizing the translation vector only (reported as BAD-NeRF* in the table). Despite this, our method still significantly outperforms BAD-NeRF. Our architecture, indeed, eliminates ambiguities in motion estimation as it leverages additional event-based supervision on an event-by-event basis, as suggested in the paper and proposed in [16, 23], in contrast to accumulating events occurring over temporal windows [13, 33], as well as applying supervision only at specific times during the exposure time, as in $E^2$-NeRF [31]. Results are reported in Figure 6. As the supervision frequency decreases, especially in sparse training views regimes, the performance also decreases. This observation aligns with the findings in [23], which suggest that noise effects and threshold variations in the event stream amplify with event accumulation, ultimately leading to a decrease in overall performance. Moreover, when only a few images are available for training, leveraging the continuous event stream to propagate absolute brightness measurements across unseen image views proves crucial for achieving top performance. Lever-

Effect of using eCRF. In Figure 5 of this document, we complement the ablation study reported in Table 3 of the paper with a qualitative assessment of our network’s key components. As discussed in the previous paragraph, the image-only architecture struggles in consistent blur conditions. Notably, incorporating event supervision significantly aids in the recovery of sharp details, as evident when comparing the first two settings in Figure 5. The performance further increases when adding the proposed eCRF module, as can be noticed in the checkerboard patterns on the background, the globe in the foreground, and the facial details of the figures. However, as discussed in the main paper, this improvement comes at the cost of over-augmented details and increased contrast, which are not present in the ground truth reference images. We attribute this phenomenon to the under-constrained optimization setting, which allows the eCRF module to freely augment these details as long as they appear correct once blurred though $L_{blur}$. We solve this issue by adding an additional prior, in the form of $L_{EDI}$, which further constrains the network in reconstructing accurate details. The improved quality is clearly demonstrated in Figure 5, where over-augmented details are removed, but without compromising essential details.

Event-by-event vs. Event-window loss. In this section, we extend the analysis of the robustness to training views reported in Figure 4-left of the paper. Specifically, utilizing our Ev-DeblurNeRF network, we examine the impact of implementing event supervision on an event-by-event basis, as we suggest in the paper and proposed in [16, 23], in contrast to accumulating events occurring over temporal windows [13, 33], as well as applying supervision only at specific times during the exposure time, as in $E^2$-NeRF [31]. Results are reported in Figure 6. As the supervision frequency decreases, especially in sparse training views regimes, the performance also decreases. This observation aligns with the findings in [23], which suggest that noise effects and threshold variations in the event stream amplify with event accumulation, ultimately leading to a decrease in overall performance. Moreover, when only a few images are available for training, leveraging the continuous event stream to propagate absolute brightness measurements across unseen image views proves crucial for achieving top performance. Lever-

![Figure 6. Analysis on event-by-event vs. event-batch losses.](image)

![Figure 7. Analysis on the robustness to model mismatches.](image)

Table 5. Extended study on motor encoder vs. COLMAP’s poses on Ev-DeblurCDAVIS. Best results in bold, second-best underlined.

<table>
<thead>
<tr>
<th>Training views</th>
<th>Motor</th>
<th>COLMAP</th>
<th>Ours</th>
<th>Ours w/o eCRF</th>
<th>Ours w/o EDI</th>
</tr>
</thead>
<tbody>
<tr>
<td>NeRF loss</td>
<td>0.05</td>
<td>0.92</td>
<td>0.91</td>
<td>0.91</td>
<td>0.91</td>
</tr>
<tr>
<td>PSNR</td>
<td>33.17</td>
<td>0.05</td>
<td>0.92</td>
<td>0.91</td>
<td>0.91</td>
</tr>
<tr>
<td>LPIPS</td>
<td>32.50</td>
<td>0.06</td>
<td>0.91</td>
<td>0.91</td>
<td>0.91</td>
</tr>
<tr>
<td>SSIM</td>
<td>0.08</td>
<td>0.91</td>
<td>0.91</td>
<td>0.91</td>
<td>0.91</td>
</tr>
</tbody>
</table>

Table 5. Extended study on motor encoder vs. COLMAP’s poses on Ev-DeblurCDAVIS. Best results in bold, second-best underlined.
C. Additional Results

Robustness to blur. In Figure 8 of this document, we supplement the analysis in Figure 4 of the main paper by comparing our Ev-DeblurNeRF network against the top-performing image-based baselines under different blur. We utilize the factory sample of the EV-DeblurBlender dataset for this analysis since it allows us to easily control the blur intensity, and it does not constitute a corner case for the image-only baselines. We change the exposure time $\tau$ of the simulated camera in the range $\tau \in \{5, 10, 20, 30, 40\}$, which results in an average pixel displacement of $\{3, 5, 11, 16, 20\}$, and a maximum displacement of $\{15, 24, 50, 75, 96\}$ in each configuration, respectively. The quantitative and qualitative comparison in Figure 8 shows that using events not only helps in cases of extreme motion but also helps when the motion is not extreme. While image-only baselines recover details blindly, by trying to estimate the blur formation through a limited set of camera poses, our network can achieve higher-quality results as it exploits blur-free information carried by events at microseconds resolution. Notably, our solution shows great robustness to motion blur, while image-only performance decreases significantly as the blur increases. While these results consider synthetic data, where the effect of noise and non-idealities is limited, they underscore the promise of event cameras as complementary sensors for attaining high-quality image synthesis even in non-ideal conditions.

Robustness to model mismatches. In this section, we analyze the proposed $e$CRF module in terms of increased robustness to model mismatches. We do so in a real setup, i.e., on the figures sample of the Ev-DeblurCDAVIS dataset, by analyzing the sensitivity of our network to the $\Theta$ event-camera threshold. While in the paper we select $\Theta$ via manual inspection, i.e., by utilizing the event double integral [29] as visual feedback following [29], we evaluate here the performance of our model when the $\Theta$ used in $\mathcal{L}_{ev}$ deviates...
from this value. We compare our network against a configuration that does not use the proposed eCRF, as well as a network where we also remove $\mathcal{L}_{\text{EDI}}$. Results are reported in Figure 7. By acting in between the rendered color space and the $\mathcal{L}_{\text{ev}}, eCRF$ can modulate the brightness, or color, $\hat{L}$ used for computing the loss, acting as a residual between the model-based supervision (Equation (1) of the paper) and the brightness actually perceived. As a result, our solution
achieves increased consistency across different choices of Θ, showcasing its ability to deviate from the model-based solution in case needed.

D. Qualitative Results and Video

We conclude this supplementary material by including extended qualitative results. In Figure 9, we complement Figure 3 of the paper by comparing the proposed method against top-performing networks across all remaining samples on the Ev-DeblurBlender and Ev-DeblurCDAVIS. Additionally, we provide a supplementary video showing qualitative results on all the samples of our proposed datasets. We strongly advise readers to watch our additional video where our method outperforms all baselines in rendering a novel-view continuous path, showing increased image quality and fewer artifacts than the other methods.

References

[6] Peng Cheng and Chellappa Rama. Pdrf: Progressively de-blurring radius field for fast and robust scene reconstruction from blurry images, 2023. 1, 2, 3, 5, 6, 9, 10


[43] International Telecommunication Union. Studio encoding parameters of digital television for standard 4: 3 and wide-


