EGS-SLAM: RGB-D Gaussian Splatting SLAM with Events

Siyu Chen*, Shenghai Yuan*, Thien-Minh Nguyen,
Zhuyu Huang, Chenyang Shi, Jin Jing, Lihua Xie *Fellow, IEEE*

***Abstract*—Gaussian Splatting SLAM (GS-SLAM) offers a notable improvement over traditional SLAM methods, in enabling photorealistic 3D reconstruction that conventional approaches often struggle to achieve. However, existing GS-SLAM systems perform poorly under persistent and severe motion blur commonly encountered in real-world scenarios, leading to significantly degraded tracking accuracy and compromised 3D reconstruction quality. To address this limitation, we propose EGS-SLAM, a novel GS-SLAM framework that fuses event data with RGB-D inputs to simultaneously reduce motion blur in images and compensate for the sparse, discrete nature of event streams, enabling robust tracking and high-fidelity 3DGS reconstruction. Specifically, our system explicitly models the camera's continuous trajectory during exposure, supporting event and blur-aware tracking and mapping on a unified 3DGS scene. Furthermore, we introduce a learnable camera response function to align the dynamic ranges of events and images, along with a no-event loss to suppress ringing artifacts during reconstruction. We validate our approach on a new dataset comprising synthetic and real-world sequences with significant motion blur. Extensive experimental results demonstrate that EGS-SLAM consistently outperforms existing GS-SLAM systems in both trajectory accuracy and photorealistic 3DGS reconstruction. The source code will be available at https://github.com/Chensiyu00/EGS-SLAM.**

***Index Terms*—Gaussian Splatting, SLAM, Event Camera.**

I. Introduction

Simultaneous Localization and Mapping (SLAM) is fundamental to robotic autonomy, enabling agents to estimate their pose and build environmental maps in unknown settings [1]–[5]. While most visual SLAM systems [6]–[9] achieve high localization accuracy, they struggle to reconstruct detailed geometry and photorealistic appearance. Recently, 3D Gaussian Splatting (3DGS) has been introduced into SLAM [10]–[13], offering explicit scene representations with real-time rendering and richer reconstructions. However, these methods typically assume high-quality, blur-free images as input.

Nevertheless, under fast or continuous motion, conventional cameras often produce motion-blurred frames, leading to the loss of critical visual details. This degradation adversely affects both camera tracking and scene reconstruction, limiting the performance of 3DGS-based SLAM systems. Although existing SLAM approaches [14], [15] attempt to address this issue through image-only deblurring techniques, they remain ineffective under sustained or strong blur, as they fail to recover sufficient detail for reliable operation. Consequently, current high-accuracy, photorealistic SLAM systems struggle to operate reliably in environments affected by motion blur.

To address these limitations, incorporating additional sensing modalities is a promising direction. Event cameras offer a compelling alternative, as their microsecond-level asynchronous brightness sensing inherently avoids motion blur. In existing works, event data has been employed in SLAM [16], [17] systems for robust camera tracking and in 3D Gaussian reconstruction [18]–[20] to recover sharp and detailed scenes under severe motion blur. To date, no existing work has integrated event information into the GS-SLAM framework to simultaneously enable online tracking under blurred conditions and the construction of high-fidelity 3D Gaussian maps.

Integrating event data into a GS-SLAM framework faces two main challenges. First, event streams are sparse and respond to per-pixel brightness changes, while image frames are captured at discrete intervals based on exposure. This inconsistency makes it challenging to fuse events and images for joint tracking and mapping within the 3DGS. Second, events and images differ significantly in their dynamic range: event data has inherently high dynamic range (HDR), whereas standard images have low dynamic range (LDR) and are constrained by exposure settings. This mismatch complicates fusion and the construction of a unified 3D Gaussian Splatting representation that effectively leverages both modalities.

To overcome these challenges, we propose an Event RGB-D GS-SLAM framework that uses event, image and depth within the camera's exposure time for accurate tracking and mapping. We model the continuous camera trajectory during each exposure interval and use it to render blur-aware images and corresponding event maps from 3DGS. These rendered signals are compared against the accumulated event maps, obtained by integrating the raw event stream within the same interval, and the image for temporally consistent tracking and mapping. To bridge the dynamic range gap between HDR events and LDR images, we introduce a learnable Camera Response Function (CRF) that transforms both modalities into a shared intensity space. This unified design enables our system to robustly operate under motion blur, leveraging the complementary advantages of both sensing modalities.

Our contributions can be summarized as:

- To our knowledge, we present the first E-RGB-D GS-SLAM framework that incorporates event data alongside RGB and depth information. By jointly modeling these modalities within the camera's exposure time, our method enables robust tracking and mapping under severe blur.
- We design an event-aided tracker and mapper for GS-SLAM that operate on blurry images, events, and depth to achieve accurate tracking and high-quality mapping. In

*Equal contribution.

† This work is supported by any National Research Foundation, Singapore under its Medium Sized Center for Advanced Robotics Technology Innovation.

Manuscript received: April 29, 2025; Revised: July 11, 2025; Accepted: August 4, 2025. This paper was recommended for publication by Editor Javier Civera upon valuation of the Associate Editor and Reviewers' comments

Siyu Chen, Shenghai Yuan, Thien-Minh Nguyen, Lihua Xie are with the School of Electrical and Electronic Engineering, Nanyang Technological University, Singapore. {siyu010, shyuan, thienminh.nguyen, elhxie}@ntu.edu.sg

Zhuyu Huang, Chenyang Shi, Jin Jing is with the School of Instrumentation and Optoelectronics Engineering, Beihang University, Beijing, China. {shicy}@buaa.edu.cn

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

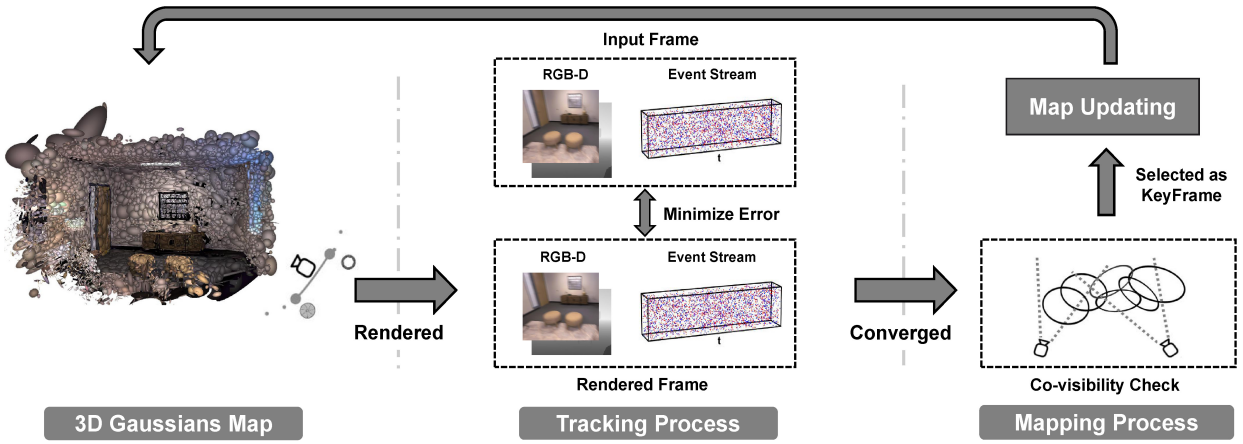

Fig. 1: **The pipeline of Gaussian Splatting SLAM with Event.** Our system integrates event–image-depth tracking and mapping within a unified 3DGS map. For each frame, the pose is estimated by jointly rendering events, image and depth. Once converged, the mapping module updates the keyframe window and the 3DGS map if selected.

the tracker and mapper, we incorporate a learnable CRF to align HDR events with LDR images and introduce a no-event loss to suppress ringing artifacts.
- We construct a dataset containing both synthetic and real-world sequences with challenging motion blur. Extensive experiments show that our method outperforms existing GS-SLAM and classical baselines, both in camera localization and in reconstructing high-fidelity 3DGS.

II. Related Works

This section presents deblur GS/NeRF, event-based GS/NeRF, and GS-SLAM which are relevant to this work.

Deblur GS/NeRF A line of research tackles 3D GS/NeRF reconstruction from motion-blurred images. Deblur-NeRF [21] introduces a deformable sparse kernel to recover sharp NeRFs from blurry inputs. BAD-NeRF [22] models dynamic blur trajectories to reconstruct clean scenes, and BAD-GS [23] extends this idea to 3DGS, yielding faster training and rendering while preserving detail. Seiskari *et al.* [24] incorporate velocity from visual-inertial odometry to mitigate both motion-blur and rolling-shutter artifacts, whereas BARD-GS [25] jointly models camera and object motion for dynamic scenes. Despite this progress, purely image-based methods still falter under extreme or persistent blur and all of these methods are offline.

Event-based GS/NeRF. Event cameras provide asynchronous, blur-free measurements that are well suited for 3D reconstruction. E^{2}NeRF [26] couples blur- and event-rendering losses to obtain sharp NeRFs from heavily blurred images. Ev-DeblurNeRF [27] learns an event-to-pixel response to denoise events and boost reconstruction quality, while E-NeRF [28] achieves event-only NeRF from a single sensor. Event3DGS [19] combines events with blur modeling for crisp 3DGS results, and IncEventGS [29] attains high-quality, pose-free 3DGS using events alone. However, all of these methods (except IncEventGS) require offline initialization of camera poses via structure-from-motion (SfM) before optimizing the neural or Gaussian scene representation and therefore cannot support online reconstruction and localization.

GS-SLAM Integrating 3DGS into SLAM has produced more photorealistic and efficient maps. GS-SLAM [11], SplaTAM [12], and MonoGS [10] unify tracking and mapping within a single 3DGS representation, while PhotoSLAM [13] reconstructs 3DGS from ORB-SLAM poses. These systems perform well on sharp inputs but severely degrade under motion blur. MBA-SLAM [15] embeds the camera-imaging process to handle blurred frames, and I^2-SLAM [14] incorporates the camera-response function for improved mapping. Nonetheless, all of these are image-only methods, which remain vulnerable to continuous and intense motion blur due to their reliance on blurred frame observations.

To our knowledge, only one NeRF-based SLAM [30] considers combining the event and frames to achieve online tracking and mapping. Crucially, no prior work has yet integrated events and RGB images into GS-SLAM that delivers online accurate tracking and high-quality 3DGS mapping.

III. Method

Our SLAM system is built around three tightly integrated components: (i) a unified 3DGS map, (ii) an event-image tracking process, and (iii) an incremental event-image mapping process, as illustrated in Fig. 1. The 3DGS map serves as the only scene representation for tracking and mapping. For each incoming frame, the tracking module jointly leverages the event stream, depth and image for tracking, estimating a continuous camera trajectory by rendering both signals from the 3DGS map over the frame's exposure interval. Once the pose of the incoming frame has converged, the mapping module evaluates its overlap with the latest keyframes to decide whether it should be selected as a new keyframe. If selected, it updates and maintains the current keyframe window and then updates the 3DGS map accordingly.

A. 3D Gaussian Splatting Map Representation

Following [31], a 3D scene is represented as a set of 3D Gaussians $\mathcal{G}$, each characterized by a mean $\boldsymbol{\mu}_i \in \mathbb{R}^3$, a covariance matrix $\Sigma_i \in \mathbb{R}^{3\times3} = R_i s_i s_i^T R_i^T$ represented by a rotation matrix $R \in \mathrm{SO}(3)$ given by a unit quaternion q_i and a scale $s_i \in \mathbb{R}^3$, an opacity $o_i \in \mathbb{R}$, and a color $\boldsymbol{c}_i \in \mathbb{R}^3$. For efficiency, the spherical harmonic representation is omitted, as

done in [10]. To render an image from these 3D Gaussians, we first project them onto the image plane using the camera pose:

$$\hat{\boldsymbol{\mu}}_i = \pi(T_{cw}, \boldsymbol{\mu}_i); \hat{\Sigma}_i = JR_{cw}\Sigma_i R_{cw}^T J^T, \tag{1}$$

where $\hat{\mu}_i$ is the projected mean, $\pi(\cdot)$ represents the projection function, and T_{cw} is the world-to-camera transformation. $\hat{\Sigma}_i$ is the projected covariance. J is the Jacobian of the affine transformation, and R_{cw} is the rotation matrix of T_{cw}. The rendered color $\boldsymbol{\mathcal{I}}(\boldsymbol{u})$ and depth $\boldsymbol{\mathcal{D}}(\boldsymbol{u})$ of the pixel at the $\boldsymbol{u}$ position are then computed by the $\boldsymbol{\alpha}$-blending of all reprojected Gaussians overlapping on the pixel, sorted by the depth:

$$\boldsymbol{\mathcal{I}}(\boldsymbol{u}) = \sum_{i\in\mathcal{N}} \boldsymbol{c}_i \alpha_i \prod_{j=1}^{i-1}(1-\alpha_j); \boldsymbol{\mathcal{D}}(\boldsymbol{u}) = \sum_{i\in\mathcal{N}} d_i \alpha_i \prod_{j=1}^{i-1}(1-\alpha_j), \tag{2}$$

where d_i denotes the projected depth of the center of the i-th 3D Gaussian and $\alpha_i = o_i e^{-\frac{1}{2}(\boldsymbol{u}-\hat{\boldsymbol{\mu}}_i)^T \hat{\Sigma}_i^{-1}(\boldsymbol{u}-\hat{\boldsymbol{\mu}}_i)}$. For simplicity, we denote T_{cw} as $\boldsymbol{T}$ in the rest of the paper.

B. Camera Motion Modeling during Exposure

To model the camera's continuous motion trajectory within a single exposure duration, we assume the motion is linear and express the pose at any instant $\eta \in [0, \tau]$ by interpolating between the start pose $\mathbf{T}_0$ and end pose $\mathbf{T}_\tau$:

$$\boldsymbol{T}(\eta) = \begin{bmatrix} \text{Slerp}(R_0, R_\tau, \frac{\eta}{\tau}) & (1-\frac{\eta}{\tau})\mathbf{t}_0 + \frac{\eta}{\tau}\mathbf{t}_\tau \\ \mathbf{0} & 1 \end{bmatrix}, \tag{3}$$

where $R_0, R_\tau \in \text{SO}(3)$ and $\mathbf{t}_0, \mathbf{t}_\tau \in \mathbb{R}^3$ are the rotation and translation components of $\mathbf{T}_0$ and $\mathbf{T}_\tau$, representing the camera poses at the start ($\eta = 0$) and end ($\eta = \tau$) of exposure. Slerp$(\cdot)$ denotes spherical linear interpolation in $\text{SO}(3)$.

C. Blur-Aware Tracking with Event

The tracking module in our SLAM system estimates the optimized camera poses (T_0^* and T_τ^*) within a pre-built 3D Gaussian map by jointly minimizing photometric, depth, and event residuals between rendered outputs and sensor observations when a new frame arrives. The rendering processes for both the image and the event are illustrated in Fig. 2.

Photometric Loss We adopt the physical imaging process formulation addressing the limitations of conventional static exposure assumptions. The static model ignores how intensity changes over time during capture, which directly causes motion blur. Following [14], [19], [23], digital camera imaging fundamentally involves two sequential stages: light capturing during sensor exposure with continuous photon accumulation over time, followed by photoelectric conversion that transforms the collected light into measurable electrical signals. This physical process is mathematically modeled as temporal integration of simulated latent sharp frames over the exposure duration and can be approximated by the discrete model as:

$$\tilde{\boldsymbol{I}}(\boldsymbol{u}) = \int_0^\tau \boldsymbol{\mathcal{I}}(\mathbf{T}(\eta), \boldsymbol{u}) d\eta \approx \frac{1}{K}\sum_{k=0}^{K-1} \boldsymbol{\mathcal{I}}(\mathbf{T}(\eta_k), \boldsymbol{u}), \tag{4}$$

where $\tilde{\boldsymbol{I}}(\boldsymbol{u})$ is integrated HDR image over the exposure time. τ is the exposure time, and $\boldsymbol{\mathcal{I}}(\mathbf{T}(\eta_k), \boldsymbol{u})$ specifies the

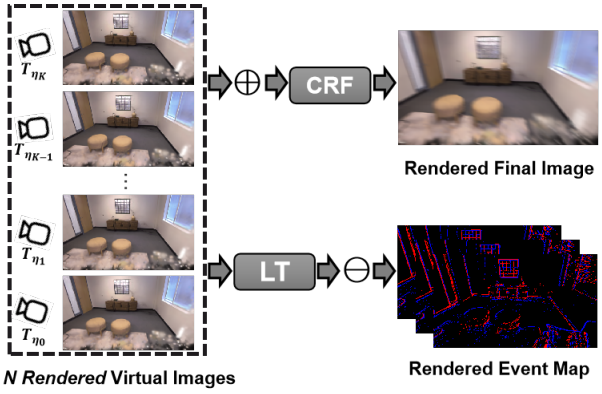

Fig. 2: **Illustration of blur-aware image and event map rendering.** Rendered images along the trajectory are aggregated via a CRF for the final image. Event maps are generated by computing logarithmic brightness differences between consecutive rendered frames.

instantaneous latent sharp image under pose $\mathbf{T}(\eta_k)$ derived by Eq. (1) and Eq. (2). The timestamps η_k are evenly divided based on the number of events, ensuring that each trajectory segment covers approximately the same distance. Since event data inherently possesses HDR characteristics while images collected by the normal camera are represented in LDR, the scene representation is HDR. Inspired by [30], we then introduce a Camera Response Function (CRF) to map the rendered HDR imagery into LDR space, thereby better preserving the joint characteristics of event data and image in tracking and mapping. Inspired by [14], [32], we obtain the synthesized LDR image $\hat{\boldsymbol{I}}(\boldsymbol{u})$ by applying a trainable CRF to the rendered HDR image $\tilde{\boldsymbol{I}}(\boldsymbol{u})$:

$$\begin{aligned} \hat{\boldsymbol{I}}(\boldsymbol{u}) &= \text{CRF}_{\text{leaky}}(\tilde{\boldsymbol{I}}(\boldsymbol{u})) \\ &= \begin{cases} \alpha\tilde{\boldsymbol{I}}(\boldsymbol{u}), & \text{if } \tilde{\boldsymbol{I}}(\boldsymbol{u}) < 0 \\ \text{Interp}(\tilde{\boldsymbol{I}}(\boldsymbol{u}), \boldsymbol{Q}) & \text{if } 0 \le \tilde{\boldsymbol{I}}(\boldsymbol{u}) \le 1, \\ -\frac{\alpha}{\sqrt{\tilde{\boldsymbol{I}}(\boldsymbol{u})}} + \alpha + 1, & \text{if } \tilde{\boldsymbol{I}}(\boldsymbol{u}) > 1 \end{cases} \end{aligned} \tag{5}$$

where $\text{Interp}(\cdot)$ represents the linear interpolation function, $\boldsymbol{Q}$ means the trainable control nodes, and α is set as 0.01 in our system. We uniformly fix N output levels in the LDR space and associate each with a trainable HDR intensity, forming the control nodes $\boldsymbol{Q}$ and shared all frames. Given an HDR input $\tilde{\boldsymbol{I}}(\boldsymbol{u})$, its corresponding LDR output is obtained by linearly interpolating between adjacent control nodes, to approximate a differentiable CRF. The photometric loss L_I is then designed:

$$L_I = \left\| \hat{\boldsymbol{I}} - \boldsymbol{I}_{obs} \right\|_1, \tag{6}$$

where I_{obs} denote the LDR image acquired by the image camera, and $\|\cdot\|_1$ represents the L_1-norm operator.

Event Loss The event stream $\mathbb{E} = \{e_m\}$ asynchronously captures spatiotemporal brightness changes, where each event $e_m = \{\boldsymbol{u}_m, t_m, p_m\}$ includes a pixel location $\boldsymbol{u}_m$, timestamp t_m, and polarity $p_m \in \{+1, -1\}$. An event is triggered when the logarithmic brightness change at pixel $\boldsymbol{u}_m$ exceeds a predefined threshold $\theta > 0$, i.e., $|\log(L(x, t_i + \Delta t)) - \log(L(x, t_i))| > \theta$. Due to their discrete and sparse nature, raw events are not directly suitable for training 3DGS. To address

Method	Metric	room0	room1	room2	office0	office1	office3	office4	Avg.	Rendering FPS
PhotoSLAM [13]	PSNR[dB]↑	18.61	19.66	–	23.85	–	17.74	14.98	–	1075.8
	SSIM↑	0.569	0.658	–	0.710	–	0.666	0.616	–	
	LPIPS↓	0.390	0.385	–	0.347	–	0.321	0.456	–	
MonoGS [10]	PSNR[dB]↑	20.47	21.97	24.05	26.75	26.76	21.41	21.79	23.32	1113.9
	SSIM↑	0.632	0.697	0.768	0.789	0.830	0.749	0.769	0.748	
	LPIPS↓	0.454	0.451	0.335	0.365	0.335	0.267	0.391	0.371	
MonoGS [10] (Refined)	PSNR[dB]↑	21.20	22.66	24.43	26.97	27.12	21.65	23.31	23.91	1052.7
	SSIM↑	0.651	0.709	0.777	0.798	0.836	0.762	0.798	0.762	
	LPIPS↓	0.383	0.388	0.307	0.320	0.299	0.249	0.328	0.325	
Ours	PSNR[dB]↑	**24.06**	**26.30**	**27.61**	**31.72**	**33.38**	**26.50**	**23.79**	**27.62**	**1134.2**
	SSIM ↑	**0.744**	**0.783**	**0.838**	**0.885**	**0.927**	**0.846**	**0.806**	**0.833**	
	LPIPS↓	**0.229**	**0.256**	**0.172**	**0.142**	**0.123**	**0.113**	**0.242**	**0.182**	

TABLE I: Rendering performance comparison of RGB-D SLAM methods on EventReplica. It should be noted that, although Photoslam may experience tracking loss in some sequences, it has the ability to reinitialize itself. The results presented here incorporate the mapping outcomes subsequent to the reinitialization process. **–** means the reinitialization attempt did not succeed. Our method outperforms the existing methods.

this, we aggregate events by position and polarity over short time intervals to construct a dense event map:

$$\boldsymbol{E}_k(\boldsymbol{u}) = \sum_{e_m \in \mathcal{E}_k} p_m, \tag{7}$$

where $\mathcal{E}_k = \{e_m \mid \boldsymbol{u}_m = \boldsymbol{u}, \eta_{k-1} < t_m < \eta_k\}$ is the subset of events within the n-th time window. Since event maps cannot be directly rendered from the Gaussian scene representation, we simulate them using the event generation model [19], [33] by computing the difference between the logarithmic brightness values of two consecutive rendered frames:

$$\hat{\boldsymbol{E}}_k(\boldsymbol{u}) = \log(\hat{\boldsymbol{B}}(\boldsymbol{T}(\eta_k), \boldsymbol{u})) - \log(\hat{\boldsymbol{B}}(\boldsymbol{T}(\eta_{k-1}), \boldsymbol{u})), \tag{8}$$

where $\hat{\boldsymbol{B}}$ denotes the grayscale brightness obtained from the rendered RGB image $\tilde{\boldsymbol{I}}$ via the BT.601 luma transform [34].

The event loss is then defined as the L1-distance between the accumulated event map $\boldsymbol{E}_k$ and rendered event maps $\hat{\boldsymbol{E}}$:

$$L_{HE} = \frac{1}{K} \sum_{n=0}^{K-1} \sum_{\boldsymbol{E}_k(\boldsymbol{u}) \neq 0} \left\| \theta \cdot \boldsymbol{E}_k(\boldsymbol{u}) - \hat{\boldsymbol{E}}_k(\boldsymbol{u}) \right\|_1. \tag{9}$$

To further leverage event-based supervision, inspired by [28], we define a no-event loss to penalize predicted undesired photometric changes at locations with no events:

$$L_{NE} = \frac{1}{K} \sum_{n=0}^{K-1} \sum_{\boldsymbol{E}_k(\boldsymbol{u}) = 0} \left\| \hat{\boldsymbol{E}}_k(\boldsymbol{u}) \right\|_1. \tag{10}$$

Unlike prior work [28], we assume that in the absence of events, no photometric change has occurred. This assumption helps enforce temporal consistency and accelerates convergence in our GS-SLAM. The final event loss is designed as:

$$L_E = L_{HE} + \lambda_{NE} L_{NE}, \tag{11}$$

where λ_{NE} is the weighting factor for the no-event loss.

Depth Loss The observed depth cannot be directly aligned with motion-blurred frames. Inspired by [14], we define the depth loss as the minimum discrepancy between the rendered depth $\mathcal{D}$ and the sensor depth $\boldsymbol{D}_{\text{obs}}$ during the exposure:

$$L_D = \min_k \left\| \boldsymbol{D}_{\text{obs}} - \boldsymbol{\mathcal{D}}(\boldsymbol{T}(\eta_k)) \right\|_1. \tag{12}$$

TABLE II: Comparison of tracking results ATE (cm) on EventReplica. L means the method loses tracking in the sequence. * indicates odometry-only methods that do not support consistent or photorealistic 3D scene reconstruction.

Scenes	RampVO* [35]	ORB-SLAM2* [6]	Photo-SLAM [13]	MonoGS [10]	Ours
room0	**4.27**	5.57	**6.64**	12.76	6.70
room1	**13.11**	L	L	8.45	**3.26**
room2	**3.43**	L	L	3.64	**3.16**
office0	**4.41**	L	L	7.44	**3.47**
office1	**3.09**	L	L	7.78	**3.53**
office3	**4.33**	6.48	6.78	7.91	**4.71**
office4	**5.87**	L	L	16.55	**12.41**
Avg.	**5.50**	-	-	9.22	**5.32**

This formulation encourages alignment between the depth map and the latent sharp image within the exposure window.

Pose Optimization To estimate the continuous trajectory during the current frame's exposure, we optimize the control nodes while keeping the 3D Gaussian map $\mathcal{G}$ fixed. The optimized control nodes T_0^* and T_τ^* are obtained by solving the following objective through iterative optimization:

$$T_0^*, T_\tau^* = \arg\min_{T_0, T_\tau} \left(\lambda_E L_E + \lambda_{ID} \left(\lambda_I L_I + \lambda_D L_D \right) \right), \tag{13}$$

where λ_E, λ_I, and λ_D are weights that control the individual contributions of each term, and λ_{ID} balances the combined image and depth terms relative to the event term.

D. Mapping Process

The mapping process comprises two core components: keyframe management and 3DGS map updating.

Keyframe Management After tracking converges, we follow [10] for keyframe selection and keyframe window maintenance. A keyframe is inserted when the overlapped visible 3D Gaussians (IoU) with the latest keyframe falls below a threshold, promoting viewpoint diversity or camera movement exceeds a threshold scaled by the current frame's average depth. To maintain a bounded keyframe set $\mathcal{W}_k$ for mapping for computational efficiency, we remove historical keyframes whose overlap with the new keyframe is below another lower threshold to preserve local relevance. If no keyframes are removed, we adopt the strategy from [36] to remove the most redundant keyframe to keep the window size fixed.

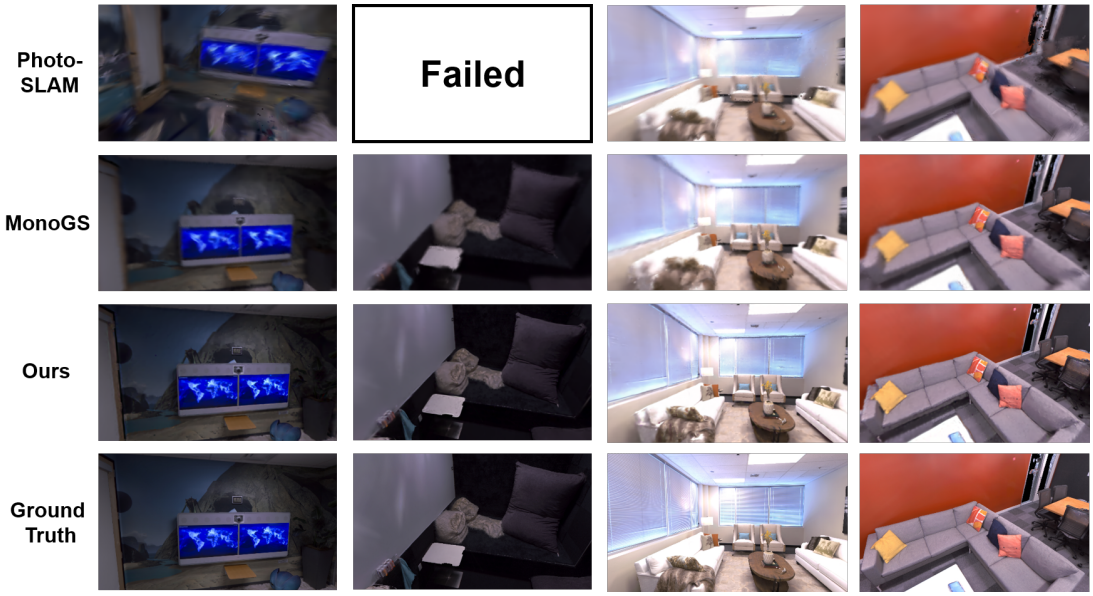

Fig. 3: **Comparison of rendering quality on the EventReplica dataset.** Our method can achieve sharper reconstructions.

3DGS Map Updating After the creation of a novel keyframe, new Gaussians are instantiated and incorporated into the established 3DGS representation. Following [10], the mean values $\boldsymbol{\mu}$ of the new Gaussians are initialized via depth back-projection leveraging estimated camera poses. The remaining parameters (including rotation q, scale s, opacity o, and color c) are initialized according to the strategy proposed in [31]. We then need to optimize the parameters. Subsequent optimization of 3DGSs incorporates an isotropic regularization term [10] to mitigate artifacts caused by skinny Gaussians:

$$L_{iso} = \frac{1}{|\mathcal{G}|} \sum_{i=1}^{|\mathcal{G}|} \|s_i - \mathbf{1} \cdot \overline{s}_i\|, \tag{14}$$

where $\overline{s}_i$ denotes the mean of the scale s_i. Two randomly selected historical keyframes, $\mathbf{w}_r^1$ and $\mathbf{w}_r^2$, are added to the current window $\mathcal{W}_c$ to form the extended mapping set $\mathcal{W}' = \mathcal{W}_c \cup \{\mathbf{w}_r^1, \mathbf{w}_r^2\}$. The final mapping loss can be designed as:

$$L_{map} = \lambda_{iso} L_{iso} + \sum_{w \in \mathcal{W}'} \left(\lambda_E L_E^w + \lambda_{ID}(\lambda_I L_I^w + \lambda_D L_D^w)\right), \tag{15}$$

where L_I^w, L_E^w, L_D^w are the photometric loss, the event loss, and the depth loss of the frame w and λ_{iso} is taken as 10. It is worth noting that the no-event loss plays a crucial role during mapping, as it significantly reduces the ringing problem—an artifact commonly observed in deblurring methods. We jointly optimize the parameters of the Gaussians and finetune the camera poses of the latest k keyframes in a sliding window to improve consistency. After the optimization of mapping, we prune the Gaussians for mapping stability.

IV. Experiments

In this section, we first introduce the dataset we collected and used in our experimental setup. We then present qualitative visual comparisons with existing methods to demonstrate the superior performance of our approach in both tracking and mapping. Next, we provide additional visualizations of the rendered results, illustrating that our method outperforms existing approaches in reconstructing sharper and higher-quality 3DGS scenes. Finally, we conduct an ablation study on the contributions of event information, the Camera Response Function, the no-event loss, and the integration of GS-SLAM with single-frame deblurring, demonstrating the importance of each component in the overall effectiveness of our system.

A. Dataset

EventReplica We created a synthetic event dataset, EventReplica, by extending the Replica dataset from [37]. The original images were resized to 459×260 pixels and cropped to 448×256 pixels. Following [25], [26], [30], we used FILM [38] to generate intermediate frames, which were then converted to event streams using VID2E [39]. The physical motion of the frames is synthesized by summing all interpolated frames within the exposure time, with the final frame serving as the sharp ground truth and its depth map being directly adopted as the depth ground truth for the corresponding blurred frame.

DEVD Our data acquisition system consists of a DAVIS346 color event camera for capturing both events and images, a RealSense D435i depth camera for acquiring depth information, and the FZMotion Motion Capture System for providing ground-truth poses. Since both the D435i depth module and the motion capture system emit 850nm infrared light—which introduces significant noise to the event camera—we installed an infrared-cut filter (transmitting only the 400-700nm visible spectrum) in front of the event sensor. Our dataset comprises four scenes: Mahjong, Mountain, Table, and Testbed.

B. Implementation details

We conducted our experiments on an NVIDIA RTX 4080 GPU. The event threshold θ was set to 0.25 for synthetic data and 0.3 for real data and $\lambda_{NE} = 0.4$. We combined RGB and depth losses with weights $\lambda_I = 0.9$ and $\lambda_D = 0.1$. For the

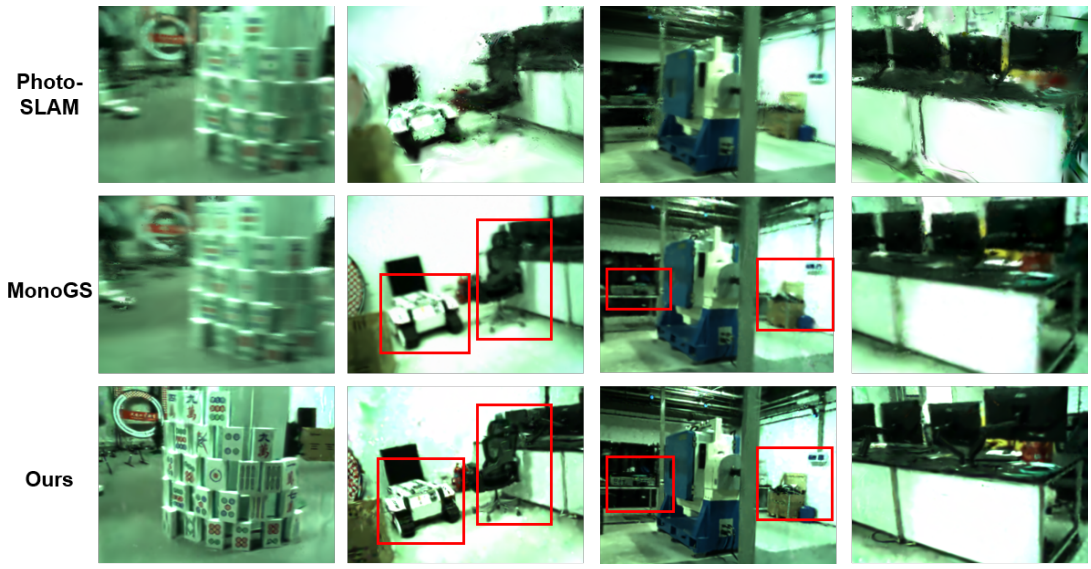

Fig. 4: **Comparison of rendering quality on the DEVD dataset.** Our method can achieve sharper reconstructions.

TABLE III: Comparison of tracking results ATE (cm) on DEVD. ✗ means the occurrence of some frame drops in this sequence. * indicates odometry-only methods that do not support consistent or photorealistic 3D scene reconstruction.

Scenes	RampVO* [35]	ORB-SLAM2* [6]	Photo-SLAM [13]	MonoGS [10]	Ours
Mahjong1	**4.90**	3.92✗	4.34✗	3.17	**1.45**
Mahjong2	**3.77**	3.72✗	5.29	2.55	**1.19**
Mountain1	**1.96**	3.47	4.50	4.30	**1.18**
Mountain2	**1.18**	4.85	4.98	3.52	**1.70**
Table1	**4.00**	10.27	25.29	6.73	**2.97**
Table2	**4.04**	4.41	4.21	10.77	**6.56**
Testbed1	**2.29**	5.26	3.69	5.15	**2.66**
Testbed2	**3.10**	4.45	**4.22**	12.41	7.58
Avg.	**3.16**	5.04	7.06	6.07	**3.16**

synthetic dataset, we set $\lambda_E = 0.05$ and $\lambda_{DI} = 0.95$; for the real dataset, $\lambda_E = 0.15$ and $\lambda_{DI} = 0.85$. A sliding window of size 10 was used for mapping, and the latest 5 frames were optimized in the backend. For fair comparison, we also set MonoGS to use the same window size. Other than that and the event-related parts, all settings are the same as MonoGS. **Boldface** indicates the best performing method and underline indicates the second best in all experimental tables.

C. Quantitative Evaluation

In this section, we benchmark our method on EventReplica and DEVD, comparing it against existing GS-SLAM approaches, including Photo-SLAM [13], and MonoGS [10]. We further include a comparison with ORB-SLAM2 [6], a classical RGB-D SLAM system, and RampVO [35], the SOTA learning-based frame-event VO, to provide a comprehensive performance assessment. We use the single-thread mode of MonoGS [10] and evaluate Absolute Trajectory Error (ATE) over the entire trajectory, along with PSNR, SSIM, and LPIPS [10], [11] for reconstruction quality comparisons.

Evaluation on synthesis dataset: EventReplica Scene reconstruction quality comparisons are shown in Tab. I, where the rendered images are evaluated against sharp ground-truth references. It is worth noting that although PhotoSLAM suffers from frequent and severe tracking loss, it can reinitialize and continue reconstruction; the results we report include such reinitialized reconstructions. MonoGS and PhotoSLAM still exhibit substantial performance degradation when reconstructing from motion-blurred images and are generally unable to recover clean and photorealistic scene appearances. Even when refined through offline reconstruction, MonoGS consistently fails to restore sharp and detailed content. In contrast, our method can significantly outperform the original MonoGS. Specifically, the PSNR improves from $23.32dB$ to $27.62dB$ $(+4.20dB)$, SSIM increases from 0.748 to 0.833 $(+0.085)$, and LPIPS drops from 0.371 to 0.183 (-0.189). These improvements demonstrate the superior capability of our approach in producing sharp and clear 3DGS.

Trajectory comparisons are reported in Tab. II. Our method outperforms existing approaches in 6 out of 7 sequences, and We reduced the average error from $9.22cm$ to $5.32cm$, achieving an improvement of approximately 42.23%. ORB-SLAM2 and PhotoSLAM experience significant tracking degradation due to motion blur, which hampers keypoint detection and causes frequent mismatches. MonoGS also shows a notable drop in tracking performance, as it relies on the assumption of blur-free inputs and struggles under blurred conditions. RampVO, with its strong learning-based trackers, achieves comparable tracking performance to our method; however, unlike RampVO, which only produces a sparse point cloud lacking photometric information, our method enables photo-realistic and high-quality clear scene reconstruction.

Evaluation on real dataset: DEVD Since ground-truth clean images are unavailable for the real-world dataset, direct quantitative comparisons are not feasible. Therefore, we present qualitative visual comparisons in the subsequent section to demonstrate the effectiveness of our method in recovering sharp and high-quality scene appearances.

In Tab. III, our method outperforms the existing baselines on the tracking performance, achieving the best ATE in 7 out of 8 sequences as well as the lowest average error overall. Compared to MonoGS, our approach yields a substantial

Event Tracking	Event Mapping	Room0				Room1				Office1			
		ATE[cm]↓	PSNR[dB]↑	SSIM↑	LPIPS↓	ATE[cm]↓	PSNR[dB]↑	SSIM↑	LPIPS↓	ATE[cm]↓	PSNR[dB]↑	SSIM↑	LPIPS↓
×	×	11.61	19.99	0.618	0.321	6.93	22.28	0.704	0.348	7.07	28.13	0.837	0.198
×	✓	11.47	19.88	0.617	0.310	6.84	21.81	0.688	0.346	8.02	27.35	0.827	0.199
✓	×	**6.45**	23.14	0.714	0.255	4.75	25.85	0.768	0.281	3.60	32.39	0.914	0.160
✓	✓	6.70	**24.06**	**0.744**	**0.229**	**3.26**	**26.30**	**0.783**	**0.256**	**3.53**	**33.38**	**0.927**	**0.123**

TABLE IV: The ablation study ATE(cm) analyzing the impact of event information on EventReplica.

TABLE V: Ablation study on ATE (cm) evaluating the impact of event information on the DEVD.

Event Tracking	Event Mapping	Mahjong1	Mahjong2	Mountain1	Table1
×	×	12.74	18.59	2.66	5.29
×	✓	14.77	6.54	2.56	5.18
✓	×	2.08	1.63	1.75	4.01
✓	✓	**1.45**	**1.19**	**1.18**	**2.97**

TABLE VI: The ablation study on ATE (cm) analyzing the effect of the CRF.

Settings	Mahjong1	Mahjong2	Mountain1	Table1
w/o CRF	1.49	1.21	1.24	3.28
w CRF (Ours)	**1.45**	**1.19**	**1.18**	**2.97**

improvement in tracking accuracy, with an average ATE reduction of approximately 47% from $6.07cm$ to $3.16cm$. Although ORB-SLAM2 and Photo-SLAM incorporate re-localization mechanisms to recover from tracking failures, they still suffer from overall inferior performance compared to ours. RampVO achieves comparable tracking performance, but can only reconstruct a sparse point cloud, while our method enables photorealistic and sharp scene reconstruction.

Runtime analysis The runtime analysis is carried out on the office1 sequence of the EventReplica dataset, where our method runs at 0.55 FPS and MonoGS reaches 1.75 FPS. The difference mainly comes from our design choice to perform more rendering times in order to combine events and images for improved performance. Importantly, this design enables our method to achieve robust tracking under challenging motion blur and to reconstruct sharp, high-fidelity 3D scenes—capabilities that MonoGS lacks despite its faster runtime. Overall, our approach prioritizes reconstruction quality and robustness, which are crucial for real-world deployment.

D. Qualitative Evaluation

In Fig. 3 and Fig. 4, we compare the mapping results of our method with PhotoSLAM and MonoGS. As shown, our rendered outputs can recover clean 3D images from blurry inputs, significantly outperforming the previous methods.

E. Ablation Study

Event Information We evaluate the contribution of event information to tracking and mapping in Tab. IV and Tab. V. We can see that when both event-based tracking and mapping are disabled, the system relies on multiple image renderings for direct tracking. This setup can be seen as a deblur-SLAM approach based solely on image and depth inputs [14], [15]. In such cases, the system struggles to recover sharp scene appearances from continuous motion blur and to achieve reliable 3D tracking. Although it may show marginal improvements over MonoGS, the overall performance remains limited.

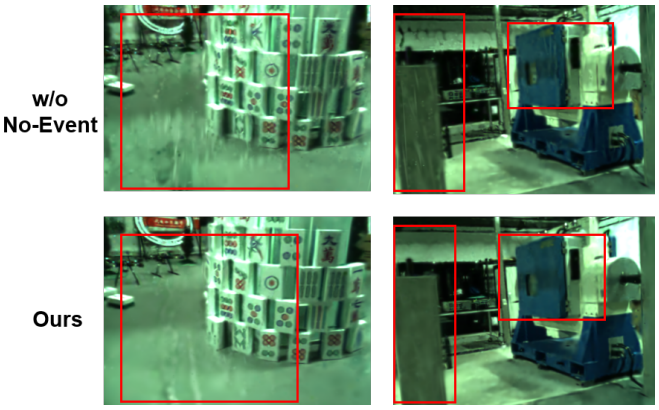

Fig. 5: **The ablation study on the No-Event Loss.** The no-event loss is highly effective in removing ringing artifacts that resemble ripple patterns.

TABLE VII: An ablation study on single-image deblurring.

Method	Room0			Room1		
	ATE[cm]↓	PSNR[dB]↑	LPIPS↓	ATE[cm]↓	PSNR[dB]↑	LPIPS↓
MonoGS	12.76	20.47	0.454	8.45	21.97	0.451
MonoGS+EDI	16.96	18.18	0.429	11.6	19.67	0.446
Ours	**6.70**	**24.06**	**0.229**	**3.26**	**26.30**	**0.256**

Enabling event-based mapping without incorporating event tracking does not yield performance gains. This is primarily due to the absence of a robust tracking mechanism, which leads to inaccurate pose estimates. These inaccuracies degrade mapping quality, which in turn further hinders tracking performance—creating a negative feedback loop that significantly impairs the system's overall effectiveness. In contrast, enabling event-based tracking without mapping results in a notable performance boost. This improvement stems from the precise pose estimation enabled by the event stream. With accurate poses, high-quality image reconstructions can be achieved using only the RGB frames and depth data. Our full model that employs both event-based tracking and mapping achieves the best overall performance, especially on real-world datasets.

No Event Loss We observe that in real-world scenarios where depth quality is limited, ringing artifacts—characterized by ripple-like distortions—easily appear in the rendered images. In Fig. 5, we compare the effectiveness of the no-event loss and find that it substantially suppress these artifacts.

Camera Response Function We evaluate the impact of the Camera Response Function (CRF) on the Majhong1, Mahjong2, Mountain1, and Table1 sequences. As shown in Tab. VI, the performance degrades when the CRF is not applied. This degradation is primarily due to the inherent difference in dynamic ranges between the event data and frame images in real-world datasets. Without CRF calibration, forcing the two modalities into a shared range leads to substantial inconsistencies. By introducing the CRF, we achieve better

alignment between modalities, reducing these inconsistencies and enhancing overall system performance.

Single-Frame Deblur For single-frame deblurring, we incorporate a widely used method, EDI [40], into MonoGS [10]. As shown in Tab. VII, the results reveal that the performance degrades after applying EDI, even compared to the baseline without deblurring. This degradation arises from artifacts introduced by single-frame deblurring methods, which act as noise and adversely affect the tracking and mapping results.

V. Conclusion

In this paper, we presented the first E-RGB-D Gaussian Splatting SLAM framework that integrates event data with image and depth inputs to enable robust tracking and high-fidelity mapping under motion blur. By explicitly modeling the camera's continuous trajectory during exposure and introducing a learnable CRF, our method effectively bridges the temporal and dynamic range discrepancies between asynchronous event streams and conventional image frames. Additionally, we proposed a no-event loss to suppress ringing artifacts, further improving the reconstruction quality. Extensive evaluations on both synthetic and real-world datasets demonstrate that our approach consistently outperforms existing GS-SLAM baselines in terms of localization accuracy and 3D scene fidelity. As the current system relies on depth input for both tracking and mapping, future work will focus on extending the framework to monocular setups.

References

[1] J. Liu and G. Hu, "Relative localization estimation for multiple robots via the rotating ultra-wideband tag," *IEEE Robot. Autom. Lett.*, vol. 8, no. 7, pp. 4187–4194, 2023.

[2] S. Yuan, B. Lou, T.-M. Nguyen *et al.*, "Large-scale uwb anchor calibration and one-shot localization using gaussian process," *arXiv preprint arXiv:2412.16880*, 2024.

[3] J. Li, X. Xu, J. Liu *et al.*, "Ua-mpc: Uncertainty-aware model predictive control for motorized lidar odometry," *IEEE Robot. Autom. Lett.*, 2025.

[4] C. Wang, D. Gao, K. Xu, J. Geng *et al.*, "PyPose: A library for robot learning with physics-based optimization," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023.

[5] Z. Yang, K. Xu, S. Yuan, and L. Xie, "A fast and light-weight noniterative visual odometry with rgb-d cameras," *Unmanned Syst.*, vol. 13, no. 03, pp. 957–969, 2025.

[6] R. Mur-Artal and J. D. Tardós, "Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras," *IEEE Trans. Robot.*, vol. 33, no. 5, pp. 1255–1262, 2017.

[7] C. Campos, R. Elvira, J. J. G. Rodríguez *et al.*, "Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam," *IEEE Trans. Robot.*, vol. 37, no. 6, pp. 1874–1890, 2021.

[8] Z. Teed and J. Deng, "Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras," *Adv. Neural Inf. Process. Syst.*, vol. 34, pp. 16 558–16 569, 2021.

[9] S. Chen, K. Liu, C. Wang *et al.*, "Salient sparse visual odometry with pose-only supervision," *IEEE Robot. Autom. Lett.*, 2024.

[10] H. Matsuki, R. Murai, P. H. Kelly, and A. J. Davison, "Gaussian splatting slam," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2024, pp. 18 039–18 048.

[11] C. Yan, D. Qu, D. Xu *et al.*, "Gs-slam: Dense visual slam with 3d gaussian splatting," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2024.

[12] N. Keetha, J. Karhade, K. M. Jatavallabhula *et al.*, "Splatam: Splat track & map 3d gaussians for dense rgb-d slam," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2024, pp. 21 357–21 366.

[13] H. Huang, L. Li, H. Cheng, and S.-K. Yeung, "Photo-slam: Real-time simultaneous localization and photorealistic mapping for monocular stereo and rgb-d cameras," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2024, pp. 21 584–21 593.

[14] G. Bae, C. Choi, H. Heo, S. M. Kim, and Y. M. Kim, "I^2-slam: Inverting imaging process for robust photorealistic dense slam," in *Proc. Eur. Conf. Comput. Vis.* Springer, 2024, pp. 72–89.

[15] P. Wang, L. Zhao, Y. Zhang, S. Zhao, and P. Liu, "Mba-slam: Motion blur aware dense visual slam with radiance fields representation," *arXiv preprint arXiv:2411.08279*, 2024.

[16] W. Chamorro, J. Sola, and J. Andrade-Cetto, "Event-based line slam in real-time," *IEEE Robot. Autom. Lett.*, vol. 7, no. 3, pp. 8146–8153, 2022.

[17] A. R. Vidal, H. Rebecq *et al.*, "Ultimate slam? combining events, images, and imu for robust visual slam in hdr and high-speed scenarios," *IEEE Robot. Autom. Lett.*, vol. 3, no. 2, pp. 994–1001, 2018.

[18] W. Yu, C. Feng, J. Tang *et al.*, "Evagaussians: Event stream assisted gaussian splatting from blurry images," *arXiv preprint arXiv:2405.20224*, 2024.

[19] T. Xiong, J. Wu, B. He *et al.*, "Event3dgs: Event-based 3d gaussian splatting for high-speed robot egomotion," *arXiv preprint arXiv:2406.02972*, 2024.

[20] H. Han, J. Li, H. Wei, and X. Ji, "Event-3dgs: Event-based 3d reconstruction using 3d gaussian splatting," *Adv. Neural Inf. Process. Syst.*, vol. 37, pp. 128 139–128 159, 2024.

[21] L. Ma, X. Li, J. Liao *et al.*, "Deblur-nerf: Neural radiance fields from blurry images," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 12 861–12 870.

[22] P. Wang, L. Zhao, R. Ma, and P. Liu, "Bad-nerf: Bundle adjusted deblur neural radiance fields," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2023, pp. 4170–4179.

[23] L. Zhao, P. Wang, and P. Liu, "Bad-gaussians: Bundle adjusted deblur gaussian splatting," in *Proc. Eur. Conf. Comput. Vis.*, 2024.

[24] O. Seiskari, J. Ylilammi, V. Kaatrasalo *et al.*, "Gaussian splatting on the move: Blur and rolling shutter compensation for natural camera motion," in *Proc. Eur. Conf. Comput. Vis.* Springer, 2024, pp. 160–177.

[25] Y. Lu, Y. Zhou, D. Liu *et al.*, "Bard-gs: Blur-aware reconstruction of dynamic scenes via gaussian splatting," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2025, pp. 16 532–16 542.

[26] Y. Qi, L. Zhu, Y. Zhang, and J. Li, "E2nerf: Event enhanced neural radiance fields from blurry images," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2023, pp. 13 254–13 264.

[27] M. Cannici and D. Scaramuzza, "Mitigating motion blur in neural radiance fields with events and frames," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2024, pp. 9286–9296.

[28] S. Klenk, L. Koestler, D. Scaramuzza, and D. Cremers, "E-nerf: Neural radiance fields from a moving event camera," *IEEE Robot. Autom. Lett.*, vol. 8, no. 3, pp. 1587–1594, 2023.

[29] J. Huang, C. Dong, and P. Liu, "Inceventgs: Pose-free gaussian splatting from a single event camera," *arXiv preprint arXiv:2410.08107*, 2024.

[30] D. Qu, C. Yan, D. Wang, J. Yin *et al.*, "Implicit event-rgbd neural slam," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2024, pp. 19 584–19 594.

[31] B. Kerbl, G. Kopanas, T. Leimkuehler, and G. Drettakis, "3D Gaussian Splatting for Real-Time Radiance Field Rendering," *ACM Trans. Graph.*, vol. 42, no. 4, pp. 1–14, 2023.

[32] K. Jun-Seong, K. Yu-Ji, M. Ye-Bin, and T.-H. Oh, "Hdr-plenoxels: Self-calibrating high dynamic range radiance fields," in *Proc. Eur. Conf. Comput. Vis.* Springer, 2022, pp. 384–401.

[33] H. Rebecq, D. Gehrig, and D. Scaramuzza, "Esim: an open event camera simulator," in *Conf. Robot Learn. (CoRL)*. PMLR, 2018, pp. 969–982.

[34] R. BT *et al.*, "Studio encoding parameters of digital television for standard 4: 3 and wide-screen 16: 9 aspect ratios," *Int. Telecommun. Union (ITU), CCIR Rep.*, 2011.

[35] R. Pellerito, M. Cannici *et al.*, "Deep visual odometry with events and frames," in *IEEE/RSJ Int. Conf. Intell. Robots Syst.*, June 2024.

[36] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 40, no. 3, pp. 611–625, 2017.

[37] Z. Zhu, S. Peng, V. Larsson *et al.*, "Nice-slam: Neural implicit scalable encoding for slam," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2022, pp. 12 786–12 796.

[38] F. Reda, J. Kontkanen, E. Tabellion, D. Sun, C. Pantofaru, and B. Curless, "Film: Frame interpolation for large motion," in *Proc. Eur. Conf. Comput. Vis.* Springer, 2022, pp. 250–266.

[39] D. Gehrig, M. Gehrig, J. Hidalgo-Carrió, and D. Scaramuzza, "Video to events: Recycling video datasets for event cameras," in *IEEE Conf. Comput. Vis. Pattern Recog.*, June 2020.

[40] L. Pan, C. Scheerlinck, X. Yu, R. Hartley, M. Liu, and Y. Dai, "Bringing a blurry frame alive at high frame-rate with an event camera," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 6820–6829.