2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

ED-SLAM: Event-Depth Gaussian Splatting SLAM

Jian Huang[1,2*], Haotian Shen[2*], Xinhao Lou[2], Peidong Liu[2†]

***Abstract*— Event-based Gaussian splatting (GS) reconstruction have recently attracted considerable attention. Existing methods usually assume the camera poses are known as a prior, or struggle to process long event streams due to the robustness of the method while poses are not known. In this work, we present ED-SLAM, an Event-Depth Gaussian Splatting-based simultaneous localization and mapping (SLAM) pipeline, which is robust to long event streams and does not require ground-truth camera poses. The pipeline achieves high-accuracy pose estimation and high-fidelity 3D reconstruction thanks to the impressive 3D representation capability of Gaussian splatting. In particular, we propose a novel bidirectional patch-based event-depth tracking algorithm and seamlessly integrate it into the Gaussian splatting mapping pipeline. Extensive experiments on both synthetic and real-world datasets demonstrate that our method significantly improves tracking accuracy and robustness, and also delivers improved reconstruction performance.**

I. Introduction

Visual Simultaneous Localization and Mapping (V-SLAM) is a foundational technology for robotics and AR/VR, enabling autonomous navigation, immersive interaction, and scene understanding in dynamic environments. Traditional V-SLAM systems [1, 2], which estimate camera motion by matching sparse features or dense patches, achieve robust localization but typically generate maps—such as point clouds or meshes—that capture only geometric skeletons. These maps often lack fine details, texture, or color information, limiting their usefulness for high-fidelity applications.

Recently, 3D Gaussian Splatting (3DGS) [3] has emerged as a powerful representation for dense, photorealistic scene reconstruction with real-time rendering. Despite this breakthrough, 3DGS-based SLAM pipelines [4] remain heavily dependent on high-quality input from conventional cameras, which often fail under challenging conditions such as low illumination, rapid motion, or textureless surfaces.

Event cameras provide a novel sensing paradigm. Operating asynchronously and responding to per-pixel brightness changes with microsecond precision, they offer high dynamic range, minimal motion blur, and low power consumption. These characteristics allow event cameras to capture fine-grained motion and intensity changes that are often lost in conventional frame-based sensors, making them particularly suitable for robust SLAM in fast-motion or low-light scenarios.

*Equal Contribution.
†Corresponding author.
[1]Jian Huang is with Zhejiang University, Hangzhou, Zhejiang, China. E-mail: huangjian2022@zju.edu.cn
[2]Jian Huang, Haotian Shen, Xinhao Liu, and Peidong Liu are with Westlake University, Hangzhou, Zhejiang, China. E-mails:{huangjian39, shenhaotian, louxinhao, liupeidong}westlake.edu.cn

While RGB-D GS-SLAM methods have demonstrated impressive dense reconstruction capabilities, their performance is fundamentally constrained by the limitations of conventional RGB-D sensors. Specifically, standard RGB-D cameras often struggle under challenging conditions due to limited dynamic range, motion blur, and relatively low temporal resolution. This motivates the use of event-based tracking, which can robustly estimate pose under such conditions and provide reliable initialization for 3DGS-based mapping.

Recent works have explored event-based SLAM with 3DGS. Methods such as *InceEvent-GS* [5] demonstrate the feasibility of combining event data with 3DGS-based SLAM. However, their tracking modules, often based on global photometric alignment, remain fragile under fast camera motion or long-term tracking. This fragility often propagates errors into the mapping pipeline, degrading reconstruction quality and system robustness.

To address these limitations, we propose **ED-SLAM**, a framework that tightly integrates a patch-based event-depth tracker with a 3DGS mapping pipeline. Inspired by DEVO [6] and DPVO [7], our tracker leverages local spatio-temporal patches to estimate dense, depth-aware motion, effectively exploiting the high temporal resolution of event data. This enables stable and accurate pose estimation even under poor illumination or high-speed motion. The tracking output directly initializes and constrains the 3DGS optimization, yielding a faster, more robust event-based SLAM system that outperforms existing methods.

We validate our approach on both synthetic and real-world event datasets. Extensive experiments show that our method significantly improves tracking stability and reconstruction performance compared to state-of-the-art event-based SLAM and 3DGS-based methods.

In summary, our key contributions are:

- A novel patch-based event-depth tracking algorithm with a bidirectional formulation for robust pose estimation.
- The first integration of event-depth tracking with 3DGS-based SLAM, enabling robust and reliable 3D reconstruction from event data over long trajectories.
- Extensive evaluation on synthetic and real-world datasets, demonstrating consistently better performance than prior methods.

II. Related Works

A. Event-based 3D Reconstruction

Recent years have witnessed remarkable progress in reconstructing 3D environments from event camera data. The evolution from implicit NeRF-based methods (e.g., E-NeRF [8],

EventNeRF [9], Ev-NeRF [10]) to explicit 3D Gaussian Splatting (e.g., Event3DGS [11], Event-3DGS [12]) has significantly improved the speed and quality of event-based reconstruction. Recent pose-free methods like EvGGS [13] and IncEventGS [5] further enable online operation by jointly optimizing poses and Gaussians. However, these approaches remain fundamentally limited: feed-forward strategies (e.g., EvGGS) suffer from poor generalization, while incremental frameworks (e.g., IncEventGS) rely on tracking modules adapted from frame-based paradigms that lack robustness under challenging conditions such as high-speed motion, low texture, or poor lighting. This inherent fragility in online tracking prevents existing systems from achieving reliable SLAM performance in the scenarios event cameras are designed for.

B. Gaussian Splatting SLAM

The advent of 3D Gaussian Splatting [3] has revolutionized dense visual SLAM by enabling real-time photorealistic mapping through explicit scene representation. While this breakthrough has inspired various SLAM systems, their performance characteristics differ significantly based on sensor modality. Monocular approaches face inherent limitations due to their dependence on RGB input quality, struggling with challenging conditions including motion blur, and poor illumination. For instance, MonoGS [14] integrates camera tracking and mapping within a unified 3DGS framework, while Splat-SLAM [15] enhances its pose and depth estimation through a monocular prediction network. Overall, these methods remain constrained by the inherent limitations of monocular vision, such as scale ambiguity and unreliable depth estimation. RGB-D variants like SplaTAM [4], GS-SLAM [16] and Photo-SLAM [17] mitigate these issues by leveraging direct depth sensing, thereby achieving more accurate geometry and improved robustness. However, their reliance on standard RGB frames for tracking leaves them vulnerable to motion blur and adverse lighting—the very failure modes of conventional cameras.

Recent work such as EGS-SLAM [18] combines event cameras with RGB-D data to address motion blur, demonstrating the potential of hybrid systems. Nevertheless, these methods still treat event data as a supplementary signal rather than establishing a tightly-coupled event-depth frontend specifically designed for challenging scenarios. Critically, under poor illumination conditions where RGB data quality severely degrades, these frame-centric approaches suffer from unreliable tracking and mapping performance, as the corrupted RGB signals directly undermine the system's perceptual capabilities. This persistent gap necessitates a paradigm shift beyond conventional frame-based sensing and motivates our work toward a fundamentally more robust solution through dedicated event-depth fusion.

III. Method

A. Overview

As illustrated in Fig. 1, *ED-SLAM* jointly estimates the camera pose and reconstructs the 3D scene from synchronized event streams and depth images. During tracking, incoming events are transformed into a time-surface map (TSM) to enable efficient and accurate edge extraction and alignment. The TSM provides a compact and temporally smoothed representation that facilitates stable geometric alignment under fast motion. Leveraging both the time-surface maps derived from events and the corresponding depth information, the tracking process incrementally recovers the 6-DoF camera pose through efficient patch-based TSM alignment, inspired by DEVO [6] and DSO [2]. For mapping, the raw event stream is directly utilized to exploit its high temporal resolution for fine-grained reconstruction. Unlike TSM, which aggregates recent events for robust pose estimation, the raw event stream preserves precise temporal ordering and event density. This detailed temporal information is essential for continuous-time 3D reconstruction and Gaussian optimization. We adopt 3DGS as the scene representation and model the camera trajectory as a continuous and differentiable function, following the approach of IncEventGS [5].

B. Patch-based Event-Depth Tracker

During the tracking stage, *ED-SLAM* employs a TSM to represent the event stream over a given time interval. A time surface (TS) is a 2D map in which each pixel stores a single temporal value — typically the timestamp of the most recent event at that pixel [6, 19]. Given a set of N events $\mathcal{E} = \{e_k\}_{k=1}^{N}$ occurring within a specified time interval, let $t_{\text{last}}(x)$ denote the timestamp of the last event at pixel coordinate $x = (u, v)$. The time surface at time $t \geq t_{\text{last}}(x)$ is then defined as:

$$\mathcal{T}(\mathbf{x}, t) = \exp\left(-\frac{t - t_{\text{last}}\,(\mathbf{x})}{\tau}\right), \tag{1}$$

where η, the decay rate, is a small constant. Because events are triggered by intensity changes, the TSM naturally concentrates information along edges and thus provides a sharp and low-latency representation of scene structure, which benefits edge alignment.

The tracking module operates independently of the mapping process, relying solely on the TSM and depth data. While inspired by DEVO, our tracker differs in its objective formulation and optimization strategy. DEVO minimizes a potential field constructed from the negated current TSM by sampling values at reprojected semi-dense 3D points using a forward compositional Lucas–Kanade scheme. In contrast, we directly minimize bidirectional photometric residuals between paired TSMs in a patch-based manner. Instead of sampling a potential field over 3D map points, we enforce symmetric TSM-to-TSM alignment in the image domain, leading to a distinct optimization objective without relying on potential-field assumptions. Given two depth maps and their corresponding TSMs at two different timestamps, the objective of the tracking module is to estimate the relative camera pose between these two timestamps.

As illustrated in Fig. 2, the core idea of our tracker is to perform bidirectional warping between the target and source

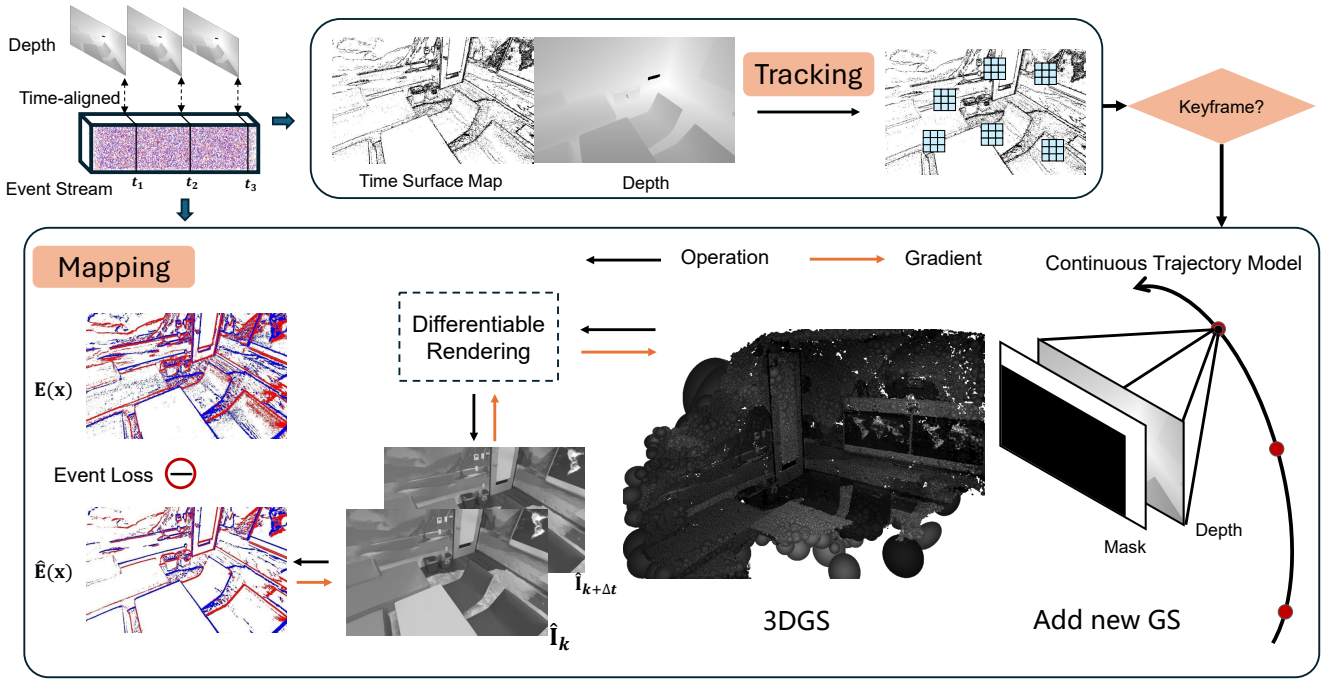

Fig. 1: **The pipeline of *ED-SLAM*.** *ED-SLAM* takes synchronized event streams and depth images as input to jointly estimate camera motion and reconstruct the scene. Incoming events are converted into time-surface maps (TSMs) and, together with the provided depth, used for patch-based alignment to incrementally recover the 6-DoF camera pose. Based on these poses, the raw high-temporal-resolution event stream is directly integrated for fine-grained 3D reconstruction using 3DGS representation, while a continuous differentiable trajectory model ensures smooth interpolation of camera poses between boundary timestamps.

time-surface maps, i.e., warping $\mathrm{TSM}_{\mathrm{tar}}$ onto $\mathrm{TSM}_{\mathrm{src}}$ and vice versa, to improve robustness. The optimal relative camera pose is then estimated by minimizing a photometric loss between the two warped maps. To enhance computational efficiency, robustness, and accuracy, this optimization is carried out in a patch-based manner rather than at the pixel level. We randomly select N patches on $\mathrm{TSM}_{\mathrm{tar}}$ from regions where the TS values exceed the threshold τ. Using ρ to represent inverse depth and (x, y) to represent pixel coordinates, we represent each patch as a $4 \times p^2$ homogeneous array:

$$\mathbf{P}_i = \begin{pmatrix} \mathbf{x} \\ \mathbf{y} \\ \mathbf{1} \\ \boldsymbol{\rho} \end{pmatrix} \quad \mathbf{x}, \mathbf{y}, \boldsymbol{\rho} \in \mathbb{R}^{1 \times p^2} \tag{2}$$

where p is the width of the patch. The patch $\mathbf{P}_i$ from $\mathrm{TSM}_{\mathrm{tar}}$ can be back-projected onto $\mathrm{TSM}_{\mathrm{src}}$.

$$\mathbf{P}'_i = \mathbf{K}\mathbf{T}\mathbf{K}^{-1}\mathbf{P}_i. \tag{3}$$

where $\mathbf{K}$ is the 4×4 camera intrinsic matrix, and $\mathbf{T}$ denotes the relative camera pose that maps 3D points from the camera frame of $\mathrm{TSM}_{\mathrm{tar}}$ to that of $\mathrm{TSM}_{\mathrm{src}}$.

$$\hat{\mathbf{t}}_{i\_tar} = \mathcal{S}_{\mathrm{bilinear}}(\mathrm{TSM}_{\mathrm{src}}, \mathbf{P}'_i), \tag{4}$$

where $\mathcal{S}_{\mathrm{bilinear}}(\mathrm{TSM}, \mathbf{P})$ denotes bilinear interpolation of the map TSM at the locations specified by $\mathbf{P}$.

Using the same procedure, the corresponding patch from the source map can be obtained as $\hat{\mathbf{t}}_{j\_src}$.

$$\mathbf{T}^* = \arg\min_{\mathbf{T}} \sum_i \left\| \mathbf{t}_{i\_tar} - \hat{\mathbf{t}}_{i\_tar} \right\|_2 + \sum_j \left\| \mathbf{t}_{j\_src} - \hat{\mathbf{t}}_{j\_src} \right\|_2 \tag{5}$$

where $\mathbf{t}_{i\_tar}$ and $\mathbf{t}_{j\_src}$ denote the ground-truth TS values of patch $\mathbf{P}_i$ in $\mathrm{TSM}_{\mathrm{tar}}$ and patch $\mathbf{P}_j$ in $\mathrm{TSM}_{\mathrm{src}}$, respectively.

To ensure accuracy and robustness, a patch is discarded if $\left| \hat{\mathbf{d}}_{src} - \mathbf{d}_{src} \right|$ or $\left| \hat{\mathbf{d}}_{tar} - \mathbf{d}_{tar} \right|$ exceeds a predefined threshold τ_d.

C. 3D Scene Representation

The scene is represented by a set of 3D Gaussians, where each is parameterized by its mean position $\boldsymbol{\mu} \in \mathbb{R}^3$, 3D covariance $\boldsymbol{\Sigma} \in \mathbb{R}^{3\times3}$, opacity $o \in \mathbb{R}$ and color $\mathbf{c} \in \mathbb{R}^3$. The distribution of each scaled Gaussian is defined as:

$$\mathbf{G}(\mathbf{x}) = e^{-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})}. \tag{6}$$

To ensure the semi-definite physical property of 3D covariance $\boldsymbol{\Sigma}$ and enable differentiable rasterization, 3DGS [3] represents the 3D covariance $\boldsymbol{\Sigma}$ and 2D covariance $\boldsymbol{\Sigma}' \in \mathbb{R}^{2\times2}$ as follows:

$$\boldsymbol{\Sigma} = \mathbf{R}\mathbf{S}\mathbf{S}^T\mathbf{R}^T, \quad \boldsymbol{\Sigma}' = \mathbf{J}\mathbf{R}_c\boldsymbol{\Sigma}\mathbf{R}_c^T\mathbf{J}^T, \tag{7}$$

where $\mathbf{S} \in \mathbb{R}^3$ is the scale, $\mathbf{R} \in \mathbb{R}^{3\times3}$ is the rotation matrix stored by a quaternion $\mathbf{q} \in \mathbb{R}^4$, $\mathbf{J} \in \mathbb{R}^{2\times3}$ is the Jacobian

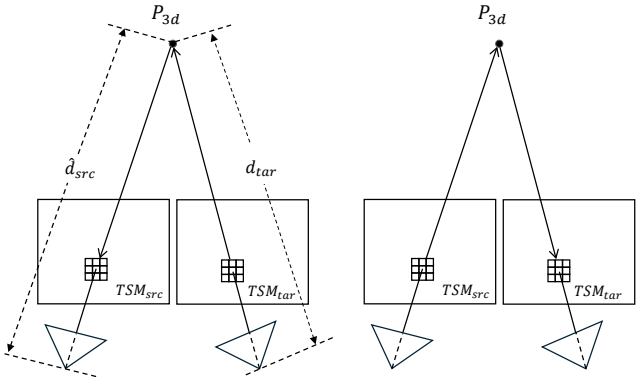

Fig. 2: Illustration of patch-based bidirectional warping between target and source time-surface maps, with pose $\mathbf{T}^*$ estimated by minimizing photometric differences. Patches are sampled from high-TS regions, warped, and their TS values are obtained via bilinear interpolation.

of the affine approximation of the projective transformation and $\mathbf{R}_c$ is the rotation part of the rendering camera pose $\mathbf{T}_c = \{\mathbf{R}_c \in \mathbb{R}^{3\times3}, \mathbf{t}_c \in \mathbb{R}^3\}$.

Afterward, each pixel color is rendered by rasterizing these N sorted 2D Gaussians based on their depths, following the formulation:

$$\mathbf{I}(\mathbf{x}) = \sum_{i=1}^{N} \mathbf{c}_i \alpha_i T_i, \quad \mathbf{D}(\mathbf{x}) = \sum_{i=1}^{N} d_i \alpha_i T_i, \tag{8}$$

where $\mathbf{c}_i$ and d_i are the color and depth of each Gaussian, $T_i = \prod_{j=1}^{i-1}(1-\alpha_j)$ and $\alpha_i = o_i \cdot \exp\left(-\frac{1}{2}\Delta_i^\top (\mathbf{\Sigma}')^{-1}\Delta_i\right)$.

D. Camera Motion Trajectory Modeling

Following prior works on event-based reconstruction [5], we parameterize the camera trajectory using a continuous, differentiable model. In particular, the pose at any timestamp is obtained via geodesic interpolation between two key poses: the initial pose $\mathbf{T}_{\text{start}} \in \text{SE}(3)$ and the final pose $\mathbf{T}_{\text{end}} \in \text{SE}(3)$ at the boundaries of each chunk. For an arbitrary time t_k between t_{start} and t_{end}, the pose is computed as

$$\mathbf{T}(t_k) = \mathbf{T}_{\text{start}} \cdot \exp\left(\frac{t_k - t_{\text{start}}}{t_{\text{end}} - t_{\text{start}}} \cdot \log\left(\mathbf{T}_{\text{start}}^{-1}\mathbf{T}_{\text{end}}\right)\right) \tag{9}$$

This formulation keeps the interpolated pose $\mathbf{T}(t_k)$ on the $\text{SE}(3)$ manifold and makes it differentiable with respect to both boundary poses $\mathbf{T}_{\text{start}}$ and $\mathbf{T}_{\text{end}}$.

E. Mapping

In the mapping process, we make direct use of the raw event stream to exploit its high temporal resolution for high-fidelity reconstruction. We sample a pair of consecutive timestamps with a small interval, denoted as t_k and $t_{k+\Delta t}$. The corresponding camera poses $\mathbf{T}_k$ and $\mathbf{T}_{k+\Delta t}$ are interpolated from the continuous camera motion trajectory. Using these poses, we render the corresponding brightness images $\hat{I}_k$ and $\hat{I}_{k+\Delta t}$ from the 3DGS. The synthesized brightness change is computed as:

$$\hat{\mathbf{E}}(\mathbf{x}) = \log\left(\hat{\mathbf{I}}_{k+\Delta t}(\mathbf{x})\right) - \log\left(\hat{\mathbf{I}}_k(\mathbf{x})\right) \tag{10}$$

Meanwhile, the measured brightness change derived from the event stream is:

$$\mathbf{E}(\mathbf{x}) = C\left\{e_i\left(\mathbf{x}, t_i, p_i\right)\right\}_{t_k < t_i < t_k + \Delta t} \tag{11}$$

where $e(\mathbf{x}_i, t_i, p_i)$ represents the i-th event occurring at pixel location $\mathbf{x}_i$ and time t_i within the specified time interval, $p_i \in \{-1, +1\}$ denotes the event polarity, and C is the fixed contrast threshold parameter. The summation operator accumulates event contributions over the temporal window Δt.

The event loss is defined as:

$$\mathcal{L}_{\text{event}} = \left\| \mathbf{E}(\mathbf{x}) - \hat{\mathbf{E}}(\mathbf{x}) \right\|_2, \tag{12}$$

For each incoming online frame, new Gaussians are introduced into the map. After tracking, the camera pose of the current frame is reliably estimated, and the accompanying depth image provides a reasonable estimate of the spatial locations where new Gaussians could be placed in the scene. However, to avoid redundancy, new Gaussians are not introduced in regions already well represented by the existing Gaussians. Following [4], a mask is constructed to identify the pixels where new Gaussians should be added:

$$\mathbf{M}(p) \;=\; \Big(\mathbf{V}(p) < \lambda_V\Big) \vee \Big(e_D(p) > \lambda_{MDE} \cdot \text{MDE}\Big) \tag{13}$$

Here $\mathbf{M}(p) \in \{0, 1\}$ is a binary mask that marks pixels where new Gaussians should be added. $\mathbf{V}(p)$ denotes the rendered alpha at pixel p and λ_V is a visibility threshold. We define the per-pixel depth error $e_D(p) = |\hat{D}(p) - D(p)|$, where $\hat{D}(p)$ is the rendered depth and $D(p)$ the observed depth. The median depth error (MDE) is computed over valid pixels in the frame. A pixel is selected if its rendered alpha is below the threshold or its depth error exceeds λ_{MDE} times the MDE; i.e., $\mathbf{M}(p) = 1$ if $\mathbf{V}(p) < \lambda_V$ or $e_D(p) > \lambda_{MDE} \cdot \text{MDE}$. Pixels with $\mathbf{M}(p) = 1$ are used to initialize new Gaussians.

During tracking and mapping, the system continuously evaluates whether the current frame should be designated as a keyframe. Following the procedure of GS-SLAM, each tracked event chunk is assessed for keyframe registration according to a covisibility criterion. A Gaussian is considered visible from a given view if it contributes to the rasterization and the accumulated ray opacity (α) has not yet reached 0.5. An event chunk i is promoted to a keyframe if its covisibility with the previous keyframe drops below a predefined threshold or if the relative translation t_{ij} becomes large compared with the median scene depth. Whenever a new keyframe is created, its pose, timestamp, depth image, and the adjacent N event streams are stored for subsequent mapping; in our implementation, N is empirically set to 5.

To prevent the 3D Gaussian kernels from becoming overly skinny, we apply the scale regularization loss [14] to a batch of Gaussians G:

$$\mathcal{L}_{\text{reg}} = \frac{1}{|G|} \sum_{\mathbf{g} \in G} \max(\max(\mathbf{S}_g)/\min(\mathbf{S}_g),\, r) - r, \tag{14}$$

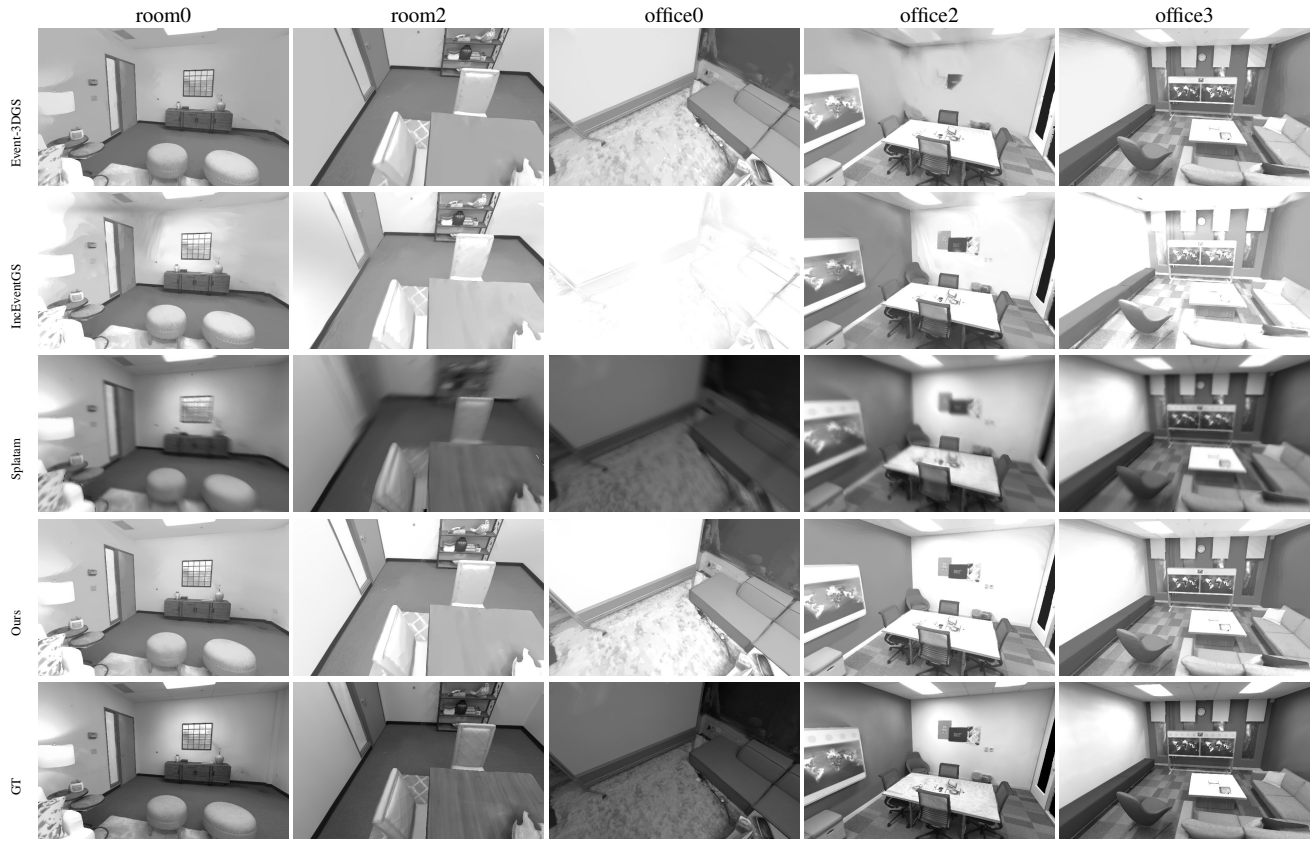

Fig. 3: Qualitative evaluation of novel view synthesis on the Replica dataset. Our method produces higher-quality images with fewer artifacts compared to the baselines. Notably, IncEventGS performs poorly on the *office0* scene due to significant pose estimation errors at this location.

where $\mathbf{S}_g$ denotes the scales of the 3D Gaussians, and r is a predefined minimum ratio that constrains the allowed anisotropy of each Gaussian. Specifically, this regularization loss penalizes Gaussians whose maximum-to-minor axis ratio exceeds r, preventing extreme elongation and ensuring stable optimization.

To further constrain the geometry of the 3D Gaussian Splatting representation, we incorporate a depth loss term defined as follows:

$$\mathcal{L}_d = \frac{1}{|R|} \sum_{\mathbf{x} \in R} \left\| D(\mathbf{x}) - D^{gt}(\mathbf{x}) \right\|_1, \tag{15}$$

where R denotes the set of rendered pixels, $D(\mathbf{x})$ represents the predicted depth at pixel $\mathbf{x}$, $D^{gt}(\mathbf{x})$ is the corresponding ground truth depth value, and $\| \cdot \|_1$ indicates the L1 norm.

The overall objective function of *ED-SLAM* is defined as a weighted combination of three loss terms:

$$\mathcal{L} = \lambda_{\text{event}} \mathcal{L}_{\text{event}} + \lambda_{\text{reg}} \mathcal{L}_{\text{reg}} + \lambda_d \mathcal{L}_{\text{d}}, \tag{16}$$

where $\mathcal{L}_{\text{event}}$ enforces consistency between the predicted and observed event data, ensuring that the reconstructed 3D scene aligns with the event stream. The regularization term $\mathcal{L}_{\text{reg}}$ encourages spatial smoothness and prevents degenerate Gaussian parameters, such as extreme scales or orientations, during optimization. Finally, the depth loss $\mathcal{L}_d$ supervises the rendered depth against the ground-truth depth, promoting accurate 3D geometry reconstruction. The weighting coefficients λ_{event}, λ_{reg}, and λ_d are empirically selected to balance the contributions of each term.

IV. Experiments

A. Experimental Setup

Datasets. We evaluate *ED-SLAM* on two widely used datasets: EventReplica [5] and VECtor [20]. EventReplica is a synthetic event dataset that provides ground-truth brightness images, enabling quantitative comparison. In contrast, VECtor is a more challenging real-world dataset.

Metrics. We evaluate novel view synthesis (NVS) using the standard metrics PSNR, SSIM, and LPIPS. To ensure fair comparisons, these metrics are computed with the evaluation code provided by EventNeRF [9], which applies a linear color transformation between the predicted and ground-truth images. Motion trajectories are assessed using the Absolute Trajectory Error (ATE), calculated with the publicly available EVO toolbox [21].

Baselines. To the best of our knowledge, *ED-SLAM* is the first dense SLAM method that jointly leverages event and depth modalities to estimate camera motion and reconstruct a dense 3D scene, which makes direct comparisons with prior work unavailable. For a meaningful evaluation, we select two representative event-based methods as references: IncEventGS [5] (monocular, pose-free) and Event3DGS

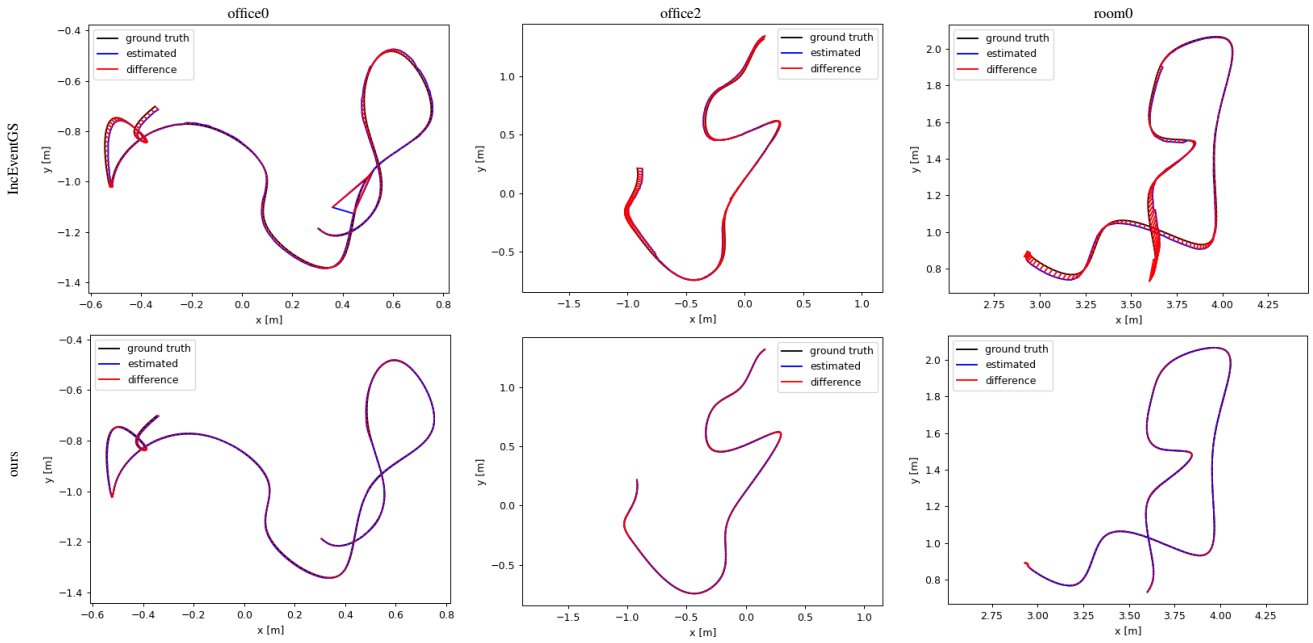

Fig. 4: Representative visualization of trajectories on the EventReplica dataset, illustrating that our method achieves more robust pose estimation over long event streams.

Patch Size	PSNR↑	SSIM↑	LPIPS↓	ATE (cm)
5 × 5	20.12	0.51	0.44	0.58
7 × 7	23.45	0.55	0.38	0.42
9 × 9	**27.85**	**0.59**	**0.31**	**0.35**
11 × 11	27.10	0.57	0.33	0.37
13 × 13	25.30	0.54	0.36	0.44

TABLE I: Ablation study on patch size in office0.

Patch Num	PSNR↑	SSIM↑	LPIPS↓	ATE (cm)
200	22.50	0.52	0.39	0.48
400	**27.85**	**0.59**	**0.31**	**0.35**
600	27.50	0.58	0.32	0.36
800	27.20	0.57	0.33	0.38

TABLE II: Ablation study on patch number in office0.

[11] (monocular, pose-required), the latter of which we re-implemented for comparison. In addition, we compare our method against state-of-the-art RGB-D Gaussian Splatting SLAM and event-depth visual odometry methods, including Splatam [4].

Implementation details. All evaluations are conducted on a PC equipped with an Intel Core i7-11700 CPU (16 cores at 2.50 GHz), 32 GB of RAM, and a single NVIDIA GeForce RTX 4090 GPU. During tracking, the patch size is set to 9×9 and the number of patches to 200. The contrast threshold C of the event camera is empirically set to 0.1 for synthetic datasets and 0.2 for real datasets. The visibility threshold λ_V is set to 0.8. The median depth error coefficient λ_{MDE} is set to 50. τ_d is set to 0.1. The weighting coefficients for the final loss, λ_{event}, λ_{reg}, and λ_d, are set to 0.7, 0.25, and 0.05, respectively. Other configurations are as default in gsplat [22].

B. Tracking Evaluation

The quantitative comparison of pose estimation is presented in Table IV. As shown, our method outperforms IncEventGS on both synthetic and real-world datasets, which can be attributed to our improved tracking strategy and the enhancements in mapping that make the overall system more robust. We further visualize the trajectory differences on the Replica dataset in Fig. 4, where it is evident that our approach achieves significantly more accurate pose estimation than IncEventGS. Compared with SplaTAM, our method also demonstrates improved pose accuracy, primarily due to its ability to fully leverage the high temporal resolution of event data.

C. Rendering Evaluation

We evaluate the NVS performance on the EventReplica dataset, which simulates fast camera motion by providing blurred brightness images and event streams. Each scene contains approximately 50 s of event data (around one billion events) along with corresponding depth images, and the results are summarized in Table III. Our method surpasses IncEventGS owing to its more robust pose tracking over long event streams. Moreover, even though Event 3D-GS is trained with ground-truth camera poses, our approach still achieves superior performance thanks to a more accurate distribution of Gaussian means and the incorporation of bundle adjustment. Because event-based methods cannot recover absolute brightness values, we restrict our quantitative comparisons to other event-based approaches.

We further perform qualitative evaluations on the EventReplica and VECtor datasets (Figs. 3 and 5, respectively). As shown in these comparisons, our method demonstrates a clear advantage over SplaTAM under fast camera motion

	room0			*room2*			*office0*			*office2*			*office3*		
	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓	PSNR↑	SSIM↑	LPIPS↓
IncEventGS	17.38	0.63	0.52	18.98	0.53	0.57	18.89	0.32	0.85	13.49	0.55	0.56	14.61	0.58	0.40
Event 3DGS	15.05	0.60	0.53	16.17	0.57	0.59	19.69	0.45	0.59	12.40	0.47	0.57	13.21	0.49	0.48
Ours	**22.37**	**0.87**	**0.18**	**24.27**	**0.83**	**0.20**	**27.85**	**0.59**	**0.31**	**22.21**	**0.81**	**0.16**	**20.43**	**0.85**	**0.15**

TABLE III: NVS performance comparison on EventReplica dataset. All results are calculated using the same code and ground-truth images. The results demonstrate that our method outperforms other baselines.

	room0	room2	office0	office2	office3	desk_normal	robot_normal	sofa_normal	hdr_normal
IncEventGS	3.75	0.84	5.48	2.38	1.58	2.16	5.62	8.78	2.15
Splatam	2.67	1.54	1.48	1.71	1.58	**0.75**	3.47	5.69	1.49
Ours	**0.18**	**0.45**	**0.35**	**0.26**	**0.66**	0.91	**0.94**	**3.00**	**1.02**

TABLE IV: Absolute trajectory error (ATE, in cm) for camera motion estimation on the Replica and VECtor datasets. Overall, our method achieves lower errors than the baselines in most scenes, demonstrating its strong pose estimation performance.

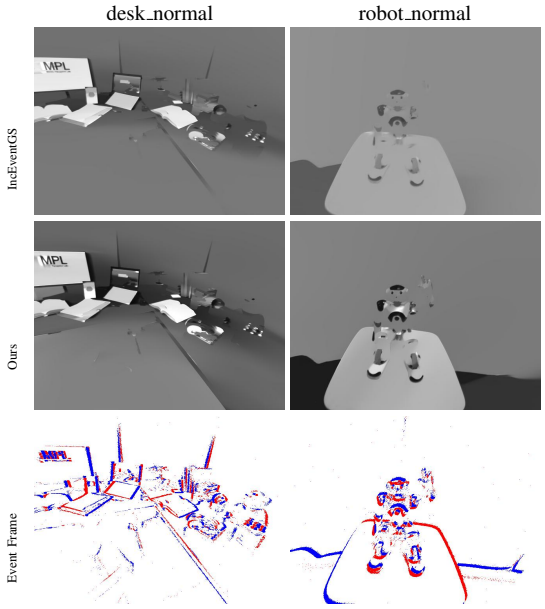

Fig. 5: Qualitative results on real-world VECtor scenes, showing that our method produces cleaner reconstructions than IncEventGS.

scenarios, primarily because it leverages the high temporal resolution of event cameras. Due to degraded pose estimation performance on long trajectories, IncEventGS produces lower-quality renderings than ours. Event 3D-GS generates images closest to ours, but it relies on ground-truth poses for training, whereas our method incrementally estimates camera poses.

D. Ablation Study

We conduct ablation studies to validate our design choices. In particular, we investigate the effects of patch size and patch number on both pose estimation accuracy and NVS metrics. All experiments are performed on the EventReplica dataset. From Tables I and II, we observe that a patch size of 9×9 and 400 patches achieve the best balance for both NVS quality and pose estimation accuracy.

V. Conclusion

In this work, we have presented *ED-SLAM*, an Event-Depth Gaussian Splatting SLAM system that jointly estimates 6-DoF camera motion and reconstructs dense 3D scenes from synchronized event streams and depth images. By leveraging patch-based alignment of time-surface maps for robust tracking and a 3D Gaussian Splatting representation for high-fidelity mapping, our method effectively handles long-duration event streams and fast camera motion. Extensive experiments on both synthetic (EventReplica) and real-world (VECtor) datasets demonstrate that *ED-SLAM* consistently outperforms prior event-based approaches such as IncEventGS and Event 3D-GS in terms of pose accuracy and novel view synthesis quality. Overall, *ED-SLAM* highlights the potential of combining event-based sensing with depth information and differentiable 3D Gaussian representations for accurate and robust SLAM in challenging scenarios.

References

[1] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos, "Orb-slam: a versatile and accurate monocular slam system," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[2] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 3, pp. 611–625, 2017.

[3] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3d gaussian splatting for real-time radiance field rendering," *ACM Transactions on Graphics*, vol. 42, no. 4, July 2023. [Online]. Available: https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/

[4] N. Keetha, J. Karhade, K. M. Jatavallabhula, G. Yang, S. Scherer, D. Ramanan, and J. Luiten, "Splatam: Splat, track & map 3d gaussians for dense rgb-d slam," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[5] J. Huang, C. Dong, X. Chen, and P. Liu, "Inceventgs: Pose-free gaussian splatting from a single event camera," in *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025, pp. 26 933–26 942.

[6] Y.-F. Zuo, J. Yang, J. Chen, X. Wang, Y. Wang, and L. Kneip, "Devo: Depth-event camera visual odometry in challenging conditions," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE, 2022, pp. 2179–2185.

[7] Z. Teed, L. Lipson, and J. Deng, "Deep patch visual odometry," *Advances in Neural Information Processing Systems*, vol. 36, pp. 39 033–39 051, 2023.

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

[9] V. Rudnev, M. Elgharib, C. Theobalt, and V. Golyanik, "Eventnerf: Neural radiance fields from a single colour event camera," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 4992–5002.

[10] I. Hwang, J. Kim, and Y. M. Kim, "Ev-nerf: Event based neural radiance field," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, 2023, pp. 837–847.

[11] T. Xiong, J. Wu, B. He, C. Fermuller, Y. Aloimonos, H. Huang, and C. A. Metzler, "Event3dgs: Event-based 3d gaussian splatting for high-speed robot egomotion," *arXiv preprint arXiv:2406.02972*, 2024.

[12] H. Han, J. Li, H. Wei, and X. Ji, "Event-3dgs: Event-based 3d reconstruction using 3d gaussian splatting," *Advances in Neural Information Processing Systems*, vol. 37, pp. 128 139–128 159, 2024.

[13] J. Wang, J. He, Z. Zhang, M. Sun, J. Sun, and R. Xu, "Evggs: A collaborative learning framework for event-based generalizable gaussian splatting," *arXiv preprint arXiv:2405.14959*, 2024.

[14] H. Matsuki, R. Murai, P. H. Kelly, and A. J. Davison, "Gaussian splatting slam," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 18 039–18 048.

[15] E. Sandström, G. Zhang, K. Tateno, M. Oechsle, M. Niemeyer, Y. Zhang, M. Patel, L. Van Gool, M. Oswald, and F. Tombari, "Splat-slam: Globally optimized rgb-only slam with 3d gaussians," in *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025, pp. 1680–1691.

[16] C. Yan, D. Qu, D. Wang, D. Xu, Z. Wang, B. Zhao, and X. Li, "Gs-slam: Dense visual slam with 3d gaussian splatting," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[17] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 21 584–21 593.

[18] S. Chen, S. Yuan, T.-M. Nguyen, Z. Huang, C. Shi, J. Jing, and L. Xie, "Egs-slam: Rgb-d gaussian splatting slam with events," *IEEE Robotics and Automation Letters*, 2025.

[19] Y. Zhou, G. Gallego, and S. Shen, "Event-based stereo visual odometry," *IEEE Transactions on Robotics*, vol. 37, no. 5, pp. 1433–1450, 2021.

[20] L. Gao, Y. Liang, J. Yang, S. Wu, C. Wang, J. Chen, and L. Kneip, "Vector: A versatile event-centric benchmark for multi-sensor slam," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 8217–8224, 2022.

[21] M. Grupp, "evo: Python package for the evaluation of odometry and slam." https://github.com/MichaelGrupp/evo, 2017.

[22] V. Ye, R. Li, J. Kerr, M. Turkulainen, B. Yi, Z. Pan, O. Seiskari, J. Ye, J. Hu, M. Tancik, and A. Kanazawa, "gsplat: An open-source library for gaussian splatting," *Journal of Machine Learning Research*, vol. 26, no. 34, pp. 1–17, 2025.