E2EGS: Event-to-Edge Gaussian Splatting for Pose-Free 3D Reconstruction

Supplementary Material

A. Edge-Gaussians initialization details

We provide algorithmic descriptions of our edge-guided Gaussian initialization process. Algorithm 1 presents the procedure for initializing 2D edge Gaussians from extracted edge points and normals through recursive grid-based subdivision. Algorithm 2 then lifts these 2D edge Gaussians to 3D through inverse depth sampling along viewing rays, complemented by random Gaussians for texture-less surface regions.

Algorithm 1: Initialize edge Gaussians by grid

Input: Edge points $\mathcal{P} = \{p_i\}_{i=1}^{N}$, normals $\mathcal{N} = \{n_i\}_{i=1}^{N}$, tile size s, angle threshold θ
Output: 2D Edge Gaussians $\mathcal{G}_{\text{edge}}$
$\mathcal{G}_{\text{edge}} \leftarrow \emptyset$;
for *each tile T of size $s \times s$* **do**
  $\mathcal{P}_T \leftarrow \{p_i \in \mathcal{P} : p_i \text{ in tile } T\}$;
  if $|\mathcal{P}_T| <$ *min_pts* **then**
    continue;
  $\mathcal{G}_T \leftarrow$ PROCESSTILERECURSIVE($\mathcal{P}_T, \mathcal{N}_T, T, \theta$, 0, max_depth);
  $\mathcal{G}_{\text{edge}} \leftarrow \mathcal{G}_{\text{edge}} \cup \mathcal{G}_T$;
return $\mathcal{G}_{edge}$;
Function PROCESSTILERECURSIVE(*$\mathcal{P}$, $\mathcal{N}$, tile, θ, depth, max_depth*)
  if $|\mathcal{P}| <$ *min_pts* ***or*** *depth* $\geq$ *max_depth* **then**
    return $\emptyset$;
  if *std*(*angles*($\mathcal{N}$)) $< \theta$ **then**
    return {CREATEGAUSSIAN($\mathcal{P}, \mathcal{N}$, *tile*)};
  else
    $\mathcal{G} \leftarrow \emptyset$;
    for *each sub-tile S in* SPLIT*(tile)* **do**
      $\mathcal{P}_S \leftarrow \{p \in \mathcal{P} : p \text{ in sub-tile } S\}$;
      $\mathcal{G} \leftarrow \mathcal{G} \cup$ PROCESSTILERECURSIVE($\mathcal{P}_S, \mathcal{N}_S, S, \theta$, depth+1, max_depth);
    return $\mathcal{G}$;
Function CREATEGAUSSIAN(*$\mathcal{P}$, $\mathcal{N}$, tile*)
  $\mu_{2D} \leftarrow \text{mean}(\mathcal{P})$;
  $n_{\text{avg}} \leftarrow \text{mean}(\mathcal{N})$;
  $q \leftarrow \text{QuaternionFromNormal}(n_{\text{avg}})$;
  return *2D Gaussian(μ_{2D}, q)*;

Algorithm 2: Lift 2D edge Gaussians to 3D

Input: 2D edge Gaussians $\mathcal{G}_{\text{edge}}$, total count N_{total}, edge ratio r_{edge}, depth bounds $d_{\min}, d_{\max}$, camera intrinsic K
Output: 3D Gaussians $\mathcal{G}_{3D}$
$N_g \leftarrow |\mathcal{G}_{\text{edge}}|$;
$N_{\text{edge}} \leftarrow \lfloor r_{\text{edge}} \cdot N_{\text{total}} \rfloor$;
$n_d \leftarrow \lfloor N_{\text{edge}} / N_g \rfloor$;
$\mathcal{G}_{3D} \leftarrow \emptyset$;
for *each Gaussian g_i at pixel x_i in $\mathcal{G}_{edge}$* **do**
  for $j = 1$ **to** n_d **do**
    $u \sim \mathcal{U}(0, 1)$;
    $d \leftarrow \left(\frac{1}{d_{\max}} + u\left(\frac{1}{d_{\min}} - \frac{1}{d_{\max}}\right)\right)^{-1}$;
    $X \leftarrow d \cdot K^{-1}[x_i, 1]^T$;
    $\mathcal{G}_{3D} \leftarrow \mathcal{G}_{3D} \cup \{\text{3D Gaussian}(X)\}$;
$N_{\text{random}} \leftarrow N_{\text{total}} - |\mathcal{G}_{3D}|$;
Add N_{random} randomly initialized Gaussians to $\mathcal{G}_{3D}$;
return $\mathcal{G}_{3D}$;

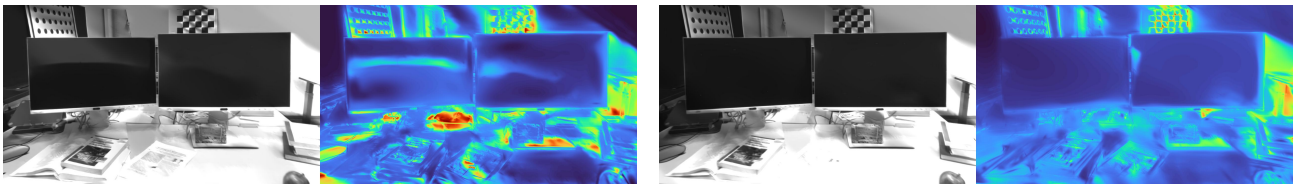
B. More details about experiments

B.1. Training details

For fair comparison with IncEventGS [1], we align the number of initialization iterations. IncEventGS performs a two-stage initialization. It first randomly initializes Gaussians, then applies depth estimation to obtain a depth map, and finally reinitializes Gaussians at the positions derived from this depth map. To ensure comparable computational settings, we adopt a similar two-stage process. We first initialize Gaussians using our edge-guided initialization method, then project their 3D positions onto the image plane, and finally reinitialize Gaussians at these projected locations. This ensures that both methods perform the same number of initialization steps.

B.2. Environment details

All experiments are conducted on a server with an AMD Ryzen Threadripper PRO 3955WX processor and NVIDIA RTX A5000 GPU. We adopt the 3DGS implementation [2] and the tracking-mapping pipeline from IncEventGS [1] with standard hyperparameter settings. For edge detection, we use $T = 3$ consecutive event maps with patch size $p = 8$ pixels and overlap ratio $\rho = 0.5$. For post-processing, we use $\sigma = 0.5$ for Gaussian smoothing, adaptive thresholding at the 70th percentile of non-zero values to only get strong edges, and morphological closing with 3×3 kernel. For



(a) $s = 16$. (b) $s = 32$.

Figure A. **Effect of tile size.** Reconstruction quality (left) and depth maps (right) for different tile sizes. With tile size $s = 16$ in (a), boundaries remain sharp and well-defined. Excessively large tile size $s = 32$ in (b) produces blurred boundaries in depth maps due to inaccurate edge Gaussian initialization.

Table A. **Ablation study on temporal window size (T).**

Window Size (T)	2	3	4	5
ATE (cm)↓	1.89	0.12	0.15	0.14

Table B. **Ablation study on patch size (hyperparameter p).**

Patch Size (p)	2	4	8	16
ATE (cm)↓	9.45	0.41	0.40	0.47
Runtime (h)↓	21.75	21.50	21.41	21.39

edge initialization, we extract edge points with $k = 5$ nearest neighbors for normal estimation. Recursive grid initialization uses tile size $s = 16$ pixels, angle threshold $\theta = 5°$, and maximum depth $D = 3$. Edge ratio is $r_{\text{edge}} = 0.3$. Inverse depth sampling uses range $[d_{\min}, d_{\max}] = [0.1, 1.1]$ meters for indoor scenes, following IncEventGS's settings. For edge-guided reconstruction, edge guidance weight is $\beta = 0.1$ and $\lambda = 0.05$.

C. Ablation study

C.1. Ablation on temporal window size

The temporal window size T determines the number of consecutive event maps used in our temporal coherence analysis. Tab. A shows that the optimal window size is $T = 3$, which achieves the best trajectory accuracy. We choose $T = 3$ to balance noise robustness and temporal consistency. Shorter sequences are more susceptible to noise, while longer sequences violate temporal coherence assumptions due to edge displacement across frames.

C.2. Ablation on patch size

The patch size p defines the spatial dimensions ($p \times p$ pixels) of local regions for computing temporal variance in our edge detection. Tab. B analyzes the impact of patch size p in our temporal coherence analysis. We use 150 frames from the TUM-VIE [3] desk2 sequence. As patch size increases, the number of windows to process decreases, resulting in reduced runtime.

Table C. **Ablation study on tile size (hyperparameter s).**

Tile Size (s)	4	8	16	32
ATE (cm)↓	0.81	0.45	0.40	9.48

When patch size becomes too small, performance dramatically degrades. At this scale, patches become nearly identical to the raw input event map, losing the spatial aggregation necessary for robust edge detection. The temporal difference analysis requires sufficient spatial context within each patch to distinguish between structured edge patterns with high variance and sparse noise with low variance. Without adequate spatial aggregation, the variance computation fails to effectively separate edges from noise, resulting in unreliable edge maps that degrade both 3D reconstruction and pose estimation.

Conversely, excessively large patches aggregate edges with diverse orientations across broader spatial regions, reducing edge localization precision. This coarse edge extraction leads to less accurate Gaussian initialization at boundaries, weakening the geometric constraints essential for precise trajectory estimation. The results demonstrate that $p = 8$ provides the optimal balance between computational efficiency and edge extraction quality for robust pose-free reconstruction.

C.3. Ablation on tile size

The tile size s determines the initial grid size ($s \times s$ pixels) for our recursive subdivision approach in edge-guided Gaussian initialization. Tab. C and Fig. A demonstrate the impact of tile size s on reconstruction quality and trajectory accuracy. As shown in Tab. C, tile sizes of $s = 16$ achieves optimal performance. When the tile size becomes too large at $s = 32$, performance dramatically degrades. This degradation occurs because large tiles aggregate edge points with significantly different orientations, and even when angular variance is high, the limited recursion depth prevents sufficient subdivision to resolve orientation conflicts. Consequently, edge Gaussians cannot be initialized at geometrically accurate positions, failing to represent the true scene

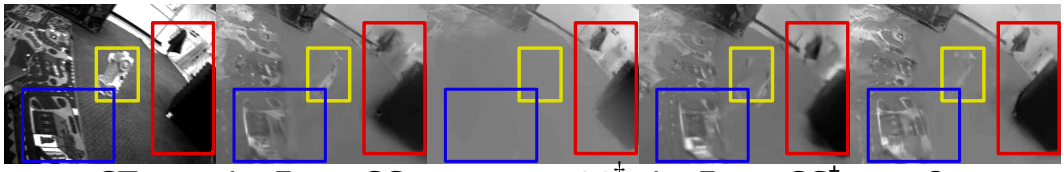


Figure B. **Dynamic scene reconstruction.** Red boxes highlight comparison regions and yellow boxes show a moving object. IncEventGS‡ employs per-frame depth estimation via Marigold model, and faces geometric distortion as shown in the blue boxes.

Table D. **Ablation study on angle threshold (hyperparameter θ).**

Angle Threshold (θ)	3	5	10	20
ATE (cm)↓	0.46	0.40	0.42	0.41

Table E. **Runtime comparison on Replica [7] (50 frames) and TUM-VIE [3] (40 frames) datasets.**

Method	**Replica (768×480)**	**TUM-VIE (1280×720)**
IncEventGS	2.11 h	5.68 h
Ours	2.12 h	5.85 h
Overhead	+0.01 h (+0.47%)	+0.17 h (+3.0%)

structure.

Fig. A visualizes this effect through depth maps. At tile size $s = 16$ shown in (a), object boundaries remain sharp and well-defined, showing clear depth discontinuities at monitor edges and desk objects. In contrast, at tile size $s = 32$ shown in (b) exhibits noticeably blurred boundaries where edges lack clear separation and depth transitions appear smooth rather than distinct. This blurring directly results from imprecise edge Gaussian initialization at large tile sizes. Because our edge-weighted optimization relies on accurate geometric constraints from edges for pose refinement, degraded edge quality directly translates to substantial trajectory errors.

C.4. Ablation on angle threshold

The angle threshold θ controls when to stop recursive subdivision based on the angular variance of edge normals within each tile. Tab. D demonstrates the stable performance across different threshold values. This stability reflects the inherent characteristics of event data. Because events naturally occur along edges with brightness changes, edge points extracted from event streams exhibit spatially coherent orientations. When applying k-nearest neighbors for normal estimation, neighboring edge points naturally share similar orientations, making the recursive subdivision process robust to variations in angular threshold.

Table F. **Dynamic scene evaluation on EVIMO dataset.**

Method	IncEventGS	IncEventGS†	IncEventGS‡	Ours
ATE (cm)	7.66	10.57	6.17	**3.19**

C.5. Runtime analysis

Tab. E reports the computational overhead of our edge-based approach compared with IncEventGS. On the Replica dataset [7] with 50 frames at 768×480 resolution, our method requires 2.12 hours, introducing only 0.01 hours of overhead, corresponding to a 0.47% increase. On the TUM-VIE dataset with 40 frames at 1280×720 resolution, the overhead increases to 0.17 hours, representing a 3.0% increase due to the higher resolution. Higher resolutions increase computational cost for edge extraction, as more patches need to be processed in our temporal coherence analysis. However, our patchwise recursive approach prevents linear scaling with resolution, keeping the overhead manageable. The additional overhead comes from edge map generation and edge Gaussian initialization, but remains negligible relative to overall training time while significantly improving trajectory accuracy and reconstruction quality.

D. Additional Results

D.1. Dynamic scene reconstruction

We evaluate our method on the EVIMO dataset [6] containing moving objects alongside camera ego-motion. As shown in Tab. F and Fig. B, our method demonstrates robust trajectory estimation even in the presence of dynamic objects, outperforming IncEventGS‡ with per-frame depth estimation. Our adaptive thresholding at the 70th percentile retains only strong edges, which implicitly filters dynamic objects. The static background exhibits sharper details with reduced blur and no geometric distortions, while IncEventGS‡ still suffers from geometric distortions (blue boxes).

D.2. Temporal consistency

Tab. G further quantifies this using Fréchet video distance (FVD) [5], where our method achieves the lowest

Table G. **Fréchet video distance comparison.**

Method	IncEventGS	IncEventGS†	Ours
FVD ↓	100.84	262.40	**57.12**

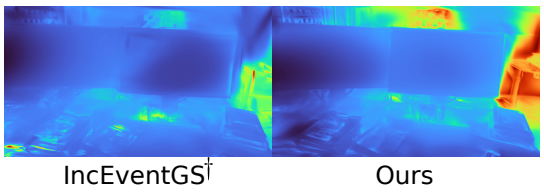


Figure C. **Depth map comparison after optimization.**

score, confirming superior temporal consistency.

D.3. Depth map after optimization

Fig. C compares depth maps after optimization. While the initialized Gaussians may exhibit depth errors in surface regions with sparse events, these are effectively refined during subsequent optimization, resulting in well-structured depth maps.

D.4. Trajectory visualization

Fig. D and Fig. E visualize estimated trajectories on the TUM-VIE sequences, with color encoding absolute pose error where blue indicates low error and red indicates high error. Our edge-guided approach maintains consistently low error across all sequences, validating the effectiveness of edge-based geometric priors for robust pose estimation in diverse scenarios. On the 1d sequence with simple translational motion, DEVO [4] demonstrates robust performance. However, on sequences with complex motion patterns including 3d, 6dof, desk, and desk2, our edge-guided approach maintains consistently low error throughout. This indicates that edge-based geometric constraints enable robust trajectory estimation across diverse motion types without external depth supervision.

D.5. Novel view synthesis

We present qualitative results on the Replica dataset in Fig. F and on the TUM-VIE dataset in Fig. G. While all methods achieve similar quality on synthetic Replica scenes, the differences become pronounced on real-world TUM-VIE sequences. This performance gap arises because synthetic data contains minimal noise and ideal lighting conditions, whereas real-world sequences exhibit significant event noise, lighting variations, and motion artifacts that challenge photometric consistency assumptions. Our edge-guided approach consistently produces accurate reconstructions, demonstrating robust performance in challenging real-world scenarios by prioritizing geometric constraints at structural boundaries over noisy photometric matching.

E. Societal Impact

Our method enables accurate 3D reconstruction and trajectory estimation from event streams without relying on pre-calibrated camera poses or auxiliary sensors. This capability benefits applications in robotics, AR/VR, and autonomous systems operating in low-light or high-speed conditions. However, improved scene reconstruction quality may increase risks related to privacy and unauthorized mapping. Careful deployment with appropriate safeguards and transparency is necessary to prevent misuse.

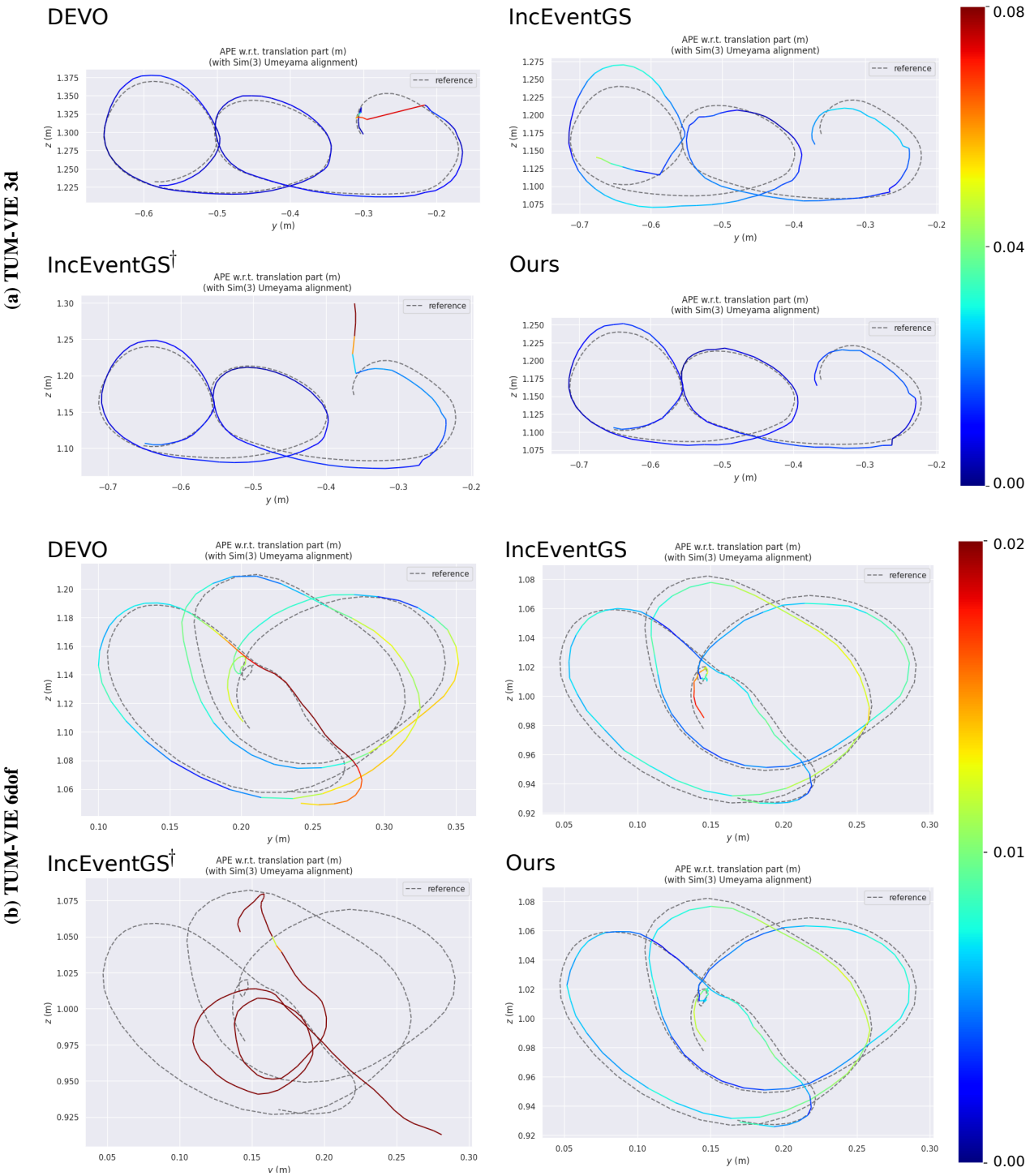


Figure D. **Trajectory visualization on TUM-VIE sequences.** Gray dotted lines indicate reference trajectories, while colored lines show estimated trajectories colored by APE (blue: low error, red: high error). Our method maintains consistently low error throughout all sequences.

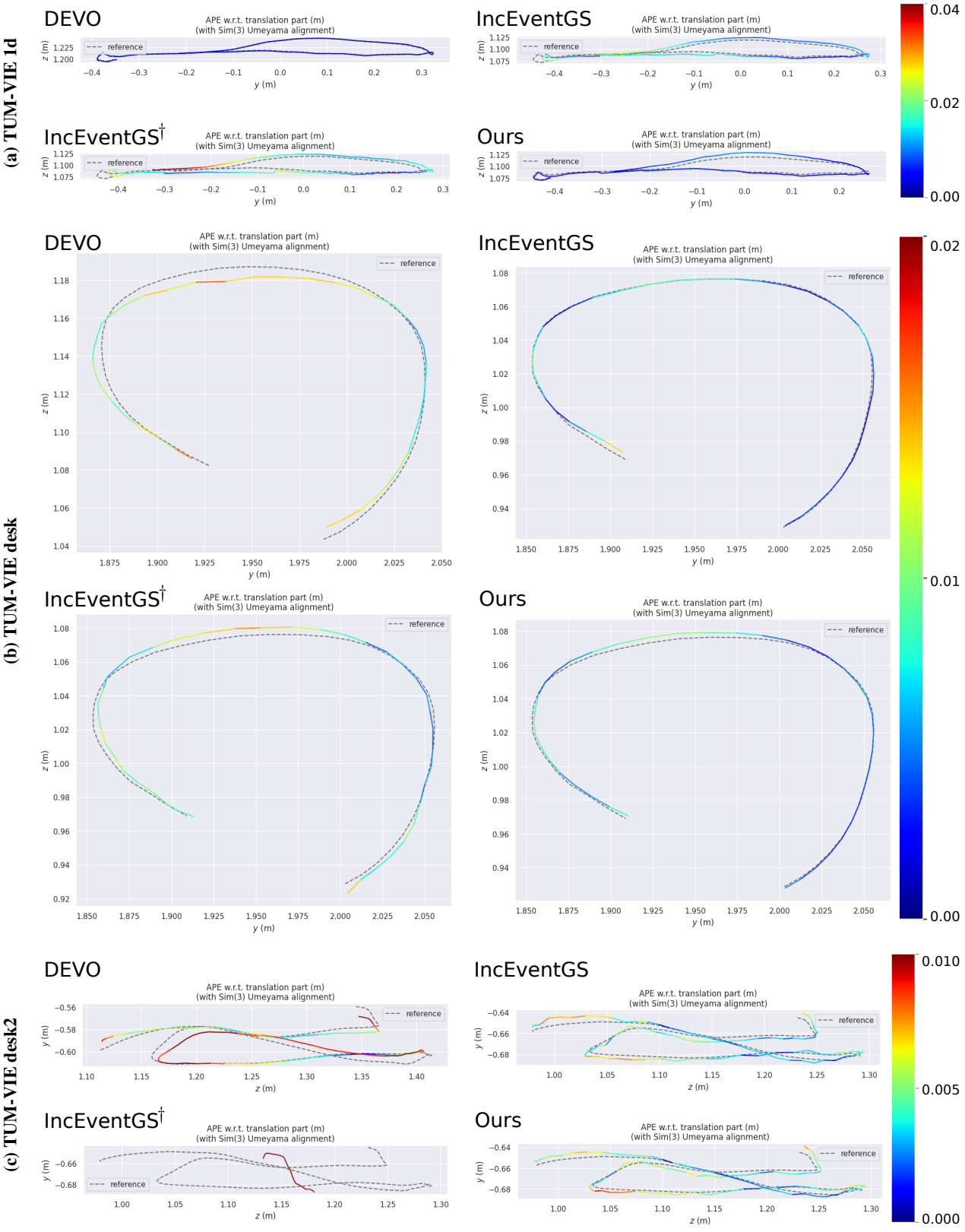


Figure E. **Trajectory visualization on TUM-VIE sequences (continued).** Gray dotted lines indicate reference trajectories, while colored lines show estimated trajectories colored by APE (blue: low error, red: high error). Our method maintains consistently low error throughout all sequences.

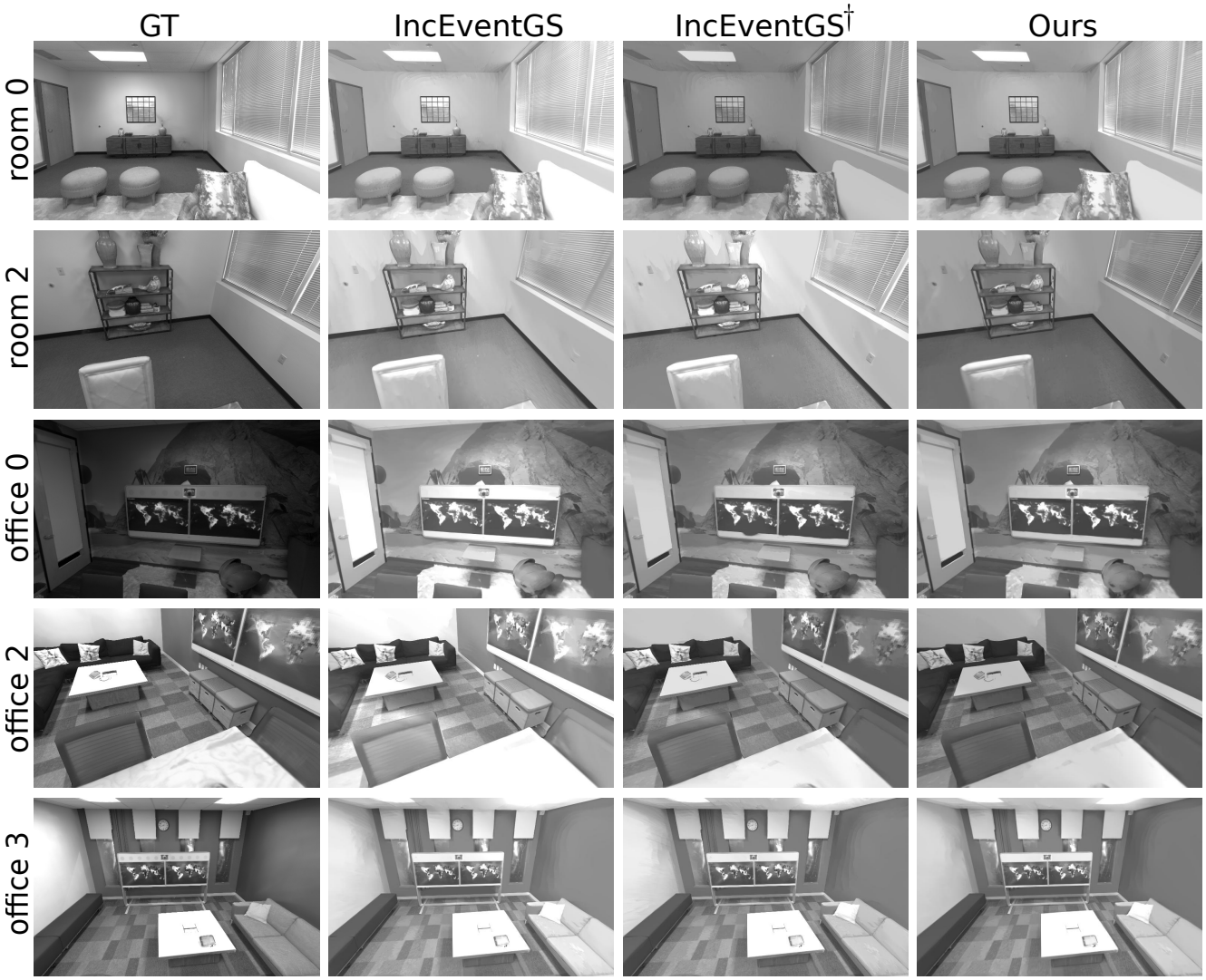


Figure F. **Qualitative results on Replica dataset.** Our method produces visually comparable results to IncEventGS with depth supervision, while significantly outperforming IncEventGS† without depth guidance across all scenes.

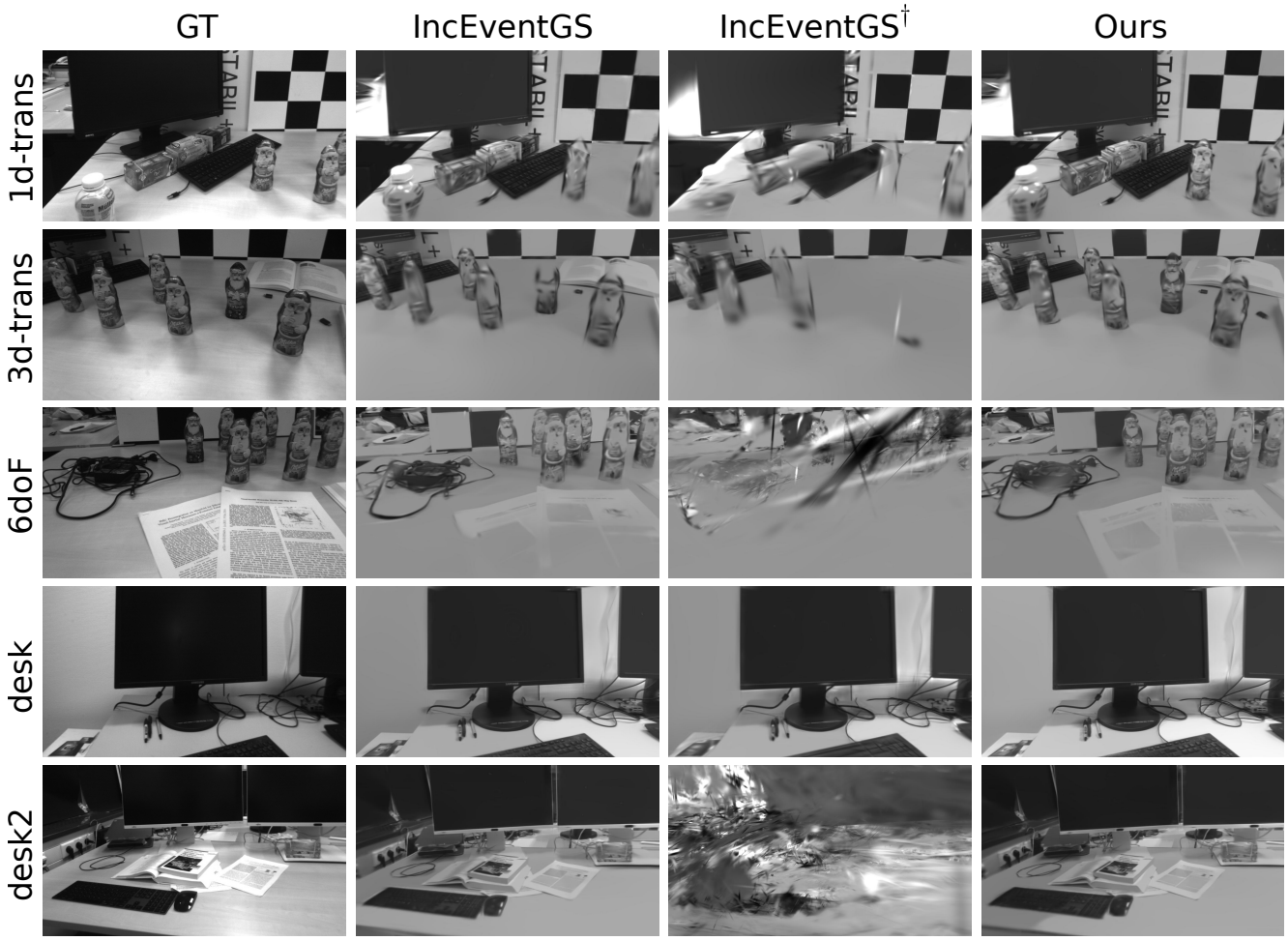


Figure G. **Qualitative results on TUM-VIE dataset.** Our method consistently produces accurate reconstructions across all sequences. IncEventGS† exhibits catastrophic failures on challenging sequences like 6-DoF and desk2 due to severe trajectory errors, while our edge-guided approach maintains sharp details and correct spatial structure.

References

[1] Jian Huang, Chengrui Dong, Xuanhua Chen, and Peidong Liu. IncEventGS: Pose-free Gaussian splatting from a single event camera. In *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, pages 26933–26942, 2025.

[2] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and Drettakis George. 3D Gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139:1–139:14, 2023.

[3] Simon Klenk, Jason Chui, Nikolaus Demmel, and Daniel Cremers. TUM-VIE: The TUM stereo visual-inertial event dataset. In *IROS*, pages 8601–8608, 2021.

[4] Simon Klenk, Marvin Motzet, Lukas Koestler, and Daniel Cremers. Deep event visual odometry. In *Proc. IEEE Int. Conf. 3D Vis.*, pages 739–749, 2024.

[5] Jiahe Liu, Youran Qu, Qi Yan, Xiaohui Zeng, Lele Wang, and Renjie Liao. Fréchet video motion distance: A metric for evaluating motion consistency in videos. *arXiv preprint arXiv:2407.16124*, 2024.

[6] A. Mitrokhin et al. EV-IMO: Motion segmentation dataset and learning pipeline for event cameras. In *IROS*, 2019.

[7] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J. Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, Anton Clarkson, Mingfei Yan, Brian Budge, Yajie Yan, Xiaqing Pan, June Yon, Yuyang Zou, Kimberly Leon, Nigel Carter, Jesus Briales, Tyler Gillingham, Elias Mueggler, Luis Pesqueira, Manolis Savva, Dhruv Batra, Hauke M. Strasdat, Renzo De Nardi, Michael Goesele, Steven Lovegrove, and Richard Newcombe. The Replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*, 2019.