EdgeSplats: Robust 3D Edge Reconstruction for In-The-Wild Data

Anonymous CVPR submission

Paper ID *****

Abstract

Robust 3D edge reconstruction from in-the-wild data remains a key challenge in robotics and SLAM, where scenes are often noisy, unstructured, and captured under imprecise conditions. Recent methods based on neural implicit representations offer compact reconstructions but suffer from high computational costs and poor localization of fine edge detail due to reliance on NeRF-style volumetric rendering. While Gaussian Splatting offers a fast and accurate alternative, its use in edge extraction has been limited to clean or synthetic datasets with precise initialization.

We present EdgeSplats, a method for 3D edge reconstruction that operates effectively on real-world data without requiring clean geometric priors. Our approach leverages the 3DGS-MCMC pipeline to train edge-aligned Gaussians directly from posed images. A graph-based clustering stage then extracts edge-consistent splats by exploiting spatial and directional coherence.

We demonstrate that EdgeSplats improves both reconstruction quality and training speed over existing methods, demonstrating strong performance on noisy, real-world inputs.

1. Introduction

Man-made environments are rich in 3D curves and lines, commonly referred to as edges. These edges delineate the structural layout of a scene and encapsulate salient information, enabling a compact and illumination-invariant representation of the environment. As a result, robust 3D edge reconstruction methods are powerful tools in applications such as robotics and SLAM.

Early 3D edge reconstruction techniques relied on matching 2D-detected line segments across multiple views. Methods like LIMAP [14] have achieved notable success using this approach, although they inherently depend on the presence of overlapping line correspondences across images.

More recent methods leverage the rise of neural implicit representations, such as Neural Radiance Fields (NeRF) [16]. Techniques including EMAP [13], NEF [31], and NEAT [28] utilize these representations to learn a dense 3D structure of a scene and then extract curves and lines from the learned geometry. However, these approaches inherit key limitations from the volumetric rendering frameworks they build upon—most notably, high computational cost and slow inference times. In addition, accuracy suffers when sampling edge points due to discretization and resolution limits [4].

Just as Gaussian Splatting has emerged as an alternative to NeRF for general 3D reconstruction [11], it has also inspired edge-focused techniques. EdgeGaussians [4], learns a set of edge-aligned Gaussian primitives to reconstruct curves and lines with both speed and accuracy. However, EdgeGaussians relies on a graph traversal-based line fitting approach, which suffers from substantial slowdowns when Gaussians are poorly aligned or when the underlying point cloud is dense or noisy.

Despite these advancements, fast and robust 3D edge reconstruction from in-the-wild data remains a open challenge. Most existing methods are evaluated under controlled conditions, and their performance degrades in real-world scenarios. Despite its strong performance in ideal settings, EdgeGaussians exhibits reduced accuracy when faced with noisy camera poses or imperfect initializations. Given that robotics and SLAM applications frequently operate in unstructured, cluttered, and sensor-imprecise environments, current methods remain insufficient for reliable deployment in practical settings.

To address these limitations, we introduce **EdgeSplats**, a method for robust 3D edge reconstruction explicitly designed for in-the-wild data. Our approach accounts for common real-world challenges, including pose uncertainty, sparse observations, and complex scene geometry.

EdgeSplats follows the general paradigm of EdgeGaussians, with key modifications to improve robustness and scalability. First, we learn a set of edge-aligned 3D Gaussians by training on edge-mapped 2D images using a minimally modified Gaussian training pipeline. In the second stage, we treat the learned Gaussians as a graph and apply a two-stage clustering process, creating 'mega-Gaussians'

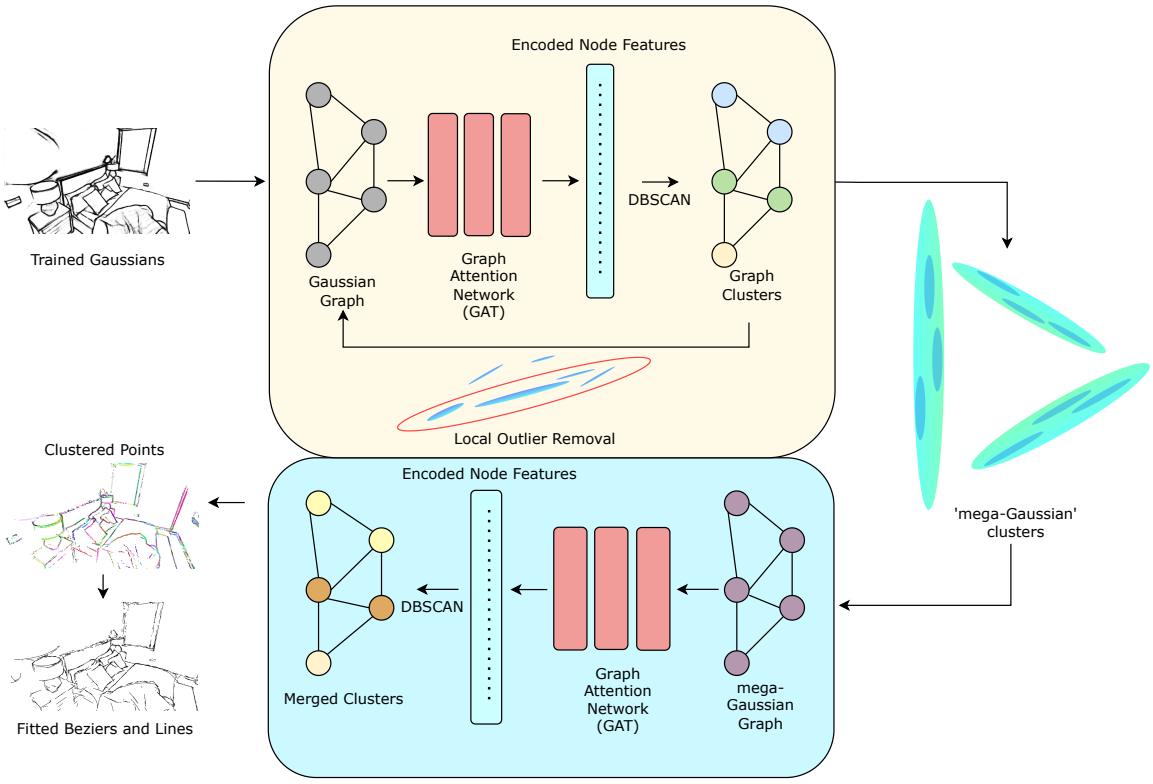

Figure 1. Given a set of edge aligned Gaussians, we first cluster the gaussian graph by encoding using a graph-attention-transformer (GAT) and DBSCAN. We prune local outliers from individual clusters, and repeat the previous clustering step. This yields a set of 'mega-Gaussians', which can be treated analogously to the constituent gaussians. A third clustering step is performed, to merge fragmented and colinear structures, to yield final gaussian clusters for line fitting.

which are further merged to form line segments. A Graph Attention Network (GAT) [25] encodes local Gaussian features into a higher-dimensional feature space, facilitating more effective clustering into meaningful curves and lines. We adopt the name EdgeSplatting (EdgeSplats) to emphasize that the resulting structures are not continuous fitted curves, but rather collections of larger Gaussian primitives whose splatted support captures the dominant edges of the scene.

Our method is trained on edge maps generated by PidiNet [23] and uses COLMAP [19] for initial pose estimation and structure-from-motion. Notably, we demonstrate that with COLMAP-derived poses, EdgeSplats can reliably reconstruct coherent 3D edges from a noisy point initialisation, exhibiting minimal performance degradation.

In summary we make the following contributions:

- We propose a robust and efficient pipeline for extracting 3D lines and curves from in-the-wild image datasets.
- By combining a modified Gaussian training pipeline with a hierarchical graph neural network encoding and clustering, our method reduces reliance on sensitive hyperparameter tuning.
- We show that EdgeSplats outperforms prior methods under increasing noise levels. Successfully reconstructing 3D edges in real-world conditions where existing approaches fail, and extracts edges **150 times** faster than existing Gaussian methods under the same conditions.

In the subsequent sections we first discuss related literature in more detail, before providing the central methodology behind ESplats. We then provide both quantitative and qualitative results comparing the approach against existing methods on benchmark datasets, before closing with discussions and future work.

2. Related Works

NeRF and Neural Implicit Methods: Neural Radiance Fields (NeRF) [16] established a foundation for view-consistent 3D scene reconstruction from posed 2D images by learning volumetric density and color fields. Numerous variants have since emerged. Methods such as [8, 33, 34] improve rendering speed and memory efficiency, while others aim to enhance representational quality [2, 3, 30]. Neu-

ral implicit models have also been adopted in SLAM, with systems like iMAP, NICE-SLAM, GoSurf, and Co-SLAM [24, 26, 27, 38] pushing toward scalable and accurate real-time mapping. However, all such approaches remain fundamentally limited by the computational cost of volumetric rendering via ray tracing.

Neural Edge Fitting: Recent methods have explored learning compact edge and curve representations directly from neural fields. EMAP, NEAT, and NEF [13, 28, 31] all predict 3D edges from an underlying neural implicit volume, producing sparse and interpretable reconstructions. Yet, like their dense counterparts, these methods are bottlenecked by the slow and resource-intensive nature of neural field training and rendering.

Gaussian Splatting: Gaussian Splatting [11] has emerged as a compelling counterpart to NeRF, offering a fast, rasterization-based alternative to traditional volumetric rendering. By avoiding the computational overhead of ray marching, it enables significantly faster training and inference while preserving high visual fidelity. Since its introduction, several extensions have improved its geometry and application scope—such as enhanced surface reconstruction [6, 7, 9] and integration into SLAM systems [1, 10, 15, 35]. The most relevant development to our work is 3DGS-Markov Chain Monte Carlo (3DGS-MCMC) [12], which reinterprets the Gaussian optimization process through a Markov Chain Monte Carlo framework. This eliminates the need for manual hyperparameter tuning and improves robustness when training from sparse or noisy inputs.

Gaussian Edge Methods: Following the trajectory of Gaussian-based rendering, recent work has explored edge reconstruction using Gaussian primitives. EdgeGaussians [4] demonstrates that edge-aligned Gaussians can be used to produce accurate and efficient 3D edge reconstructions, outperforming neural methods in both speed and geometric quality. However, its local line-fitting strategy is sensitive to noisy or imperfect initialisations, which can lead to performance degradation. SketchSplat [32] builds on the foundation of EdgeGaussians by continuing to optimize the parameters of edges after initial fitting, however its scope remains largely limited to CAD level datasets.

A related effort, 3Doodle [5], proposes a lightweight representation of objects optimizing Bézier curves, rather than Gaussians. While promising in compact abstraction, its application is currently limited to object-scale scenes and has not yet been extended to full environments.

3. Methodology

Figure 1 illustrates the **EdgeSplats** pipeline. Starting from edge-aligned Gaussians, we iteratively cluster them into geometrically consistent groups, which are then fitted with simple parametric curves.

Learning Edge-Aligned Gaussians At the core of our method is a set of view-dependent 3D Gaussian primitives, each has learnable parameters: mean $\mu \in \mathbb{R}^3$, covariance $\Sigma \in \mathbb{R}^{3\times3}$, opacity $\alpha \in [0,1]$, and color $c \in \mathbb{R}^3$. The spatial density of a Gaussian at location x is defined as:

$$G(x) = e^{-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)} \tag{1}$$

To enforce geometric fidelity over appearance matching, we fix all Gaussian colors to white and disable colour optimization. This biases optimization toward spatial alignment with 3D edges rather than background colour reproduction. A scale regularizer is also applied to prevent Gaussians from collapsing to infinitesimal volume.

Training is performed by projecting the Gaussians into the image plane through a differentiable rasteriser, and minimizing a masked $\mathcal{L}_1$ loss against binary edge maps derived from PiDiNet [23]. To counter the imbalance between edge and background pixels, we define a weight map $w(x)$ such that both regions contribute equally:

$$\mathcal{L}_{\text{edge-aware}} = \sum_{x\in\Omega} w(x)\cdot\left|\hat{I}(x) - I(x)\right| \tag{2}$$

where

$$w(x) = \begin{cases} \frac{|\bar{M}|}{|M|+|\bar{M}|}, & \text{if } M(x)=1 \\ \frac{|M|}{|M|+|\bar{M}|}, & \text{if } M(x)=0 \end{cases}$$

where $M(x) \in \{0,1\}$ is the binary edge mask.

To promote spatial consistency without hand-tuned directional losses, we adopt the resampling-based optimization from 3DGS-MCMC [12], wherein underperforming Gaussians are periodically relocated near well-performing neighbors. This encourages local structure while avoiding oversegmentation.

Graph-Based Clustering of Gaussians We propose a three-stage hierarchical clustering procedure for edge fitting. First, we obtain initial clusters by grouping Gaussians using Euclidean proximity and directional alignment. Second, we prune local outliers within each cluster and recluster the pruned set to refine groupings. Third, we aggregate each refined cluster into a "mega-Gaussian" (center from the mean of members, covariance from PCA) and construct a new graph over these mega-Gaussians; a final clustering pass over Euclidean center distances merges larger structures and removes duplicates.

We represent the set of optimized 3D Gaussians as a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where each node $v_i \in \mathcal{V}$ corresponds to a Gaussian parameterized by its center μ_i, covariance Σ_i,

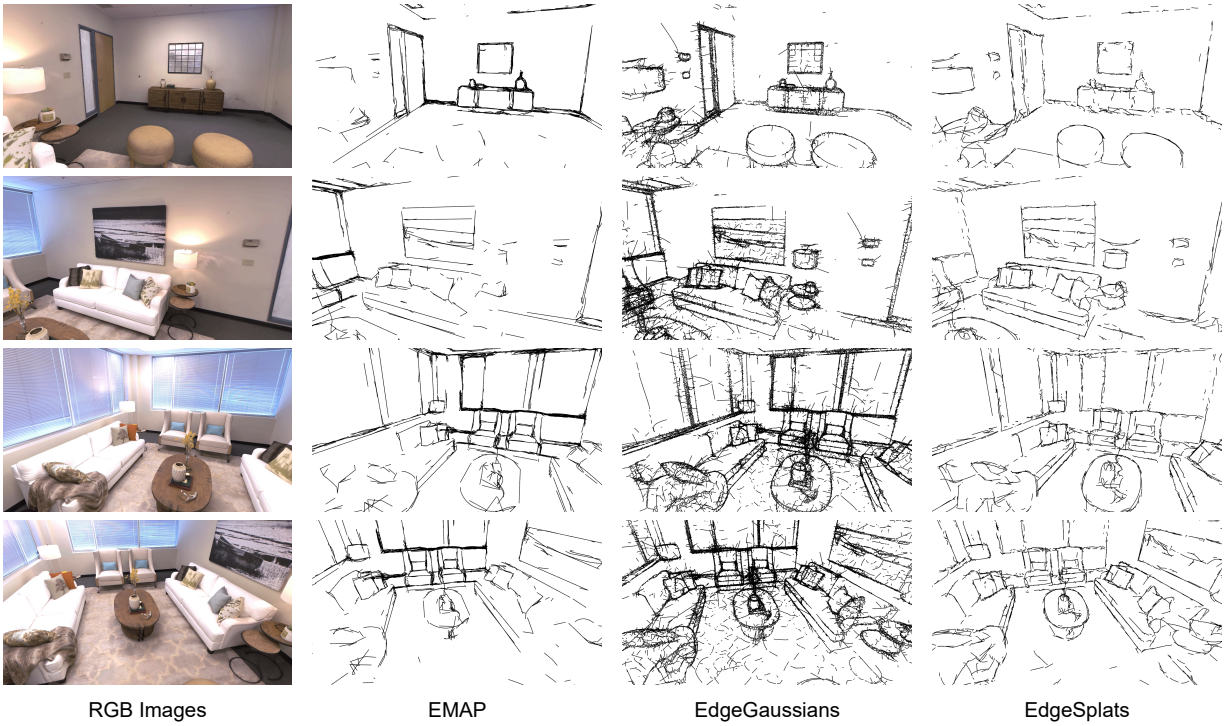

Figure 2. Qualitative results on Room0 of the Replica dataset. Comparing the outputs of EMAP, EdgeGaussians, and EdgeSplats against the input PidiNet edge maps. All methods were initialised with COLMAP estimated poses and seed points. It can be seen that EdgeSplats is less noisy than EdgeGaussians, whilst being more complete than EMAP.

and opacity α_i. Edges are constructed by connecting each Gaussian to its k-nearest neighbours in Euclidean space. For each edge $e_{ij} \in \mathcal{E}$, we define the weight as the directional affinity between Gaussian i's principal axis and the unit vector from μ_i to μ_j.

$$w_{ij} = \left| d_i^\top \frac{\mu_j - \mu_i}{\|\mu_j - \mu_i\|} \right| \tag{3}$$

where d_i is the unit principal direction (first eigenvector of Σ_i).

We then feed the graph into an untrained Graph Attention Network (GAT) acting as an encoder, which maps the node features into a higher-dimensional embedding space for clustering. We apply DBSCAN to these embeddings to obtain initial clusters $\{\mathcal{C}_1, \ldots, \mathcal{C}_N\}$. To improve cluster quality, we run RANSAC on each initial cluster to remove local geometric outliers from the graph. Following this pruning, graph edges and weights are recalculated, and a second pass through the same GAT encoder is performed. DBSCAN is reapplied to the encoded, pruned graph, for refined clusters.

To improve representation and reduce fragmentation, we treat each cluster as a "mega-Gaussian," defining its center as the mean of the constituent Gaussian centers and computing its covariance via principal component analysis (PCA). Mega-Gaussians with approximately isotropic covariance are discarded, as such clusters typically correspond to poorly structured or noisy regions. The remaining mega-Gaussians are then processed analogously to the initial Gaussians, with a new graph constructed by connecting nodes based on Euclidean distance between mega-Gaussian centers.

We then construct a new graph over the remaining mega-Gaussians by once more connecting each node to its k nearest neighbours in Euclidean space, this time with edge weights given by the Euclidean distance between centers:

$$w_{ij} = \|\mu_i - \mu_j\|_2. \tag{4}$$

We choose Euclidean distance over alignment between mega-Gaussian centers. At this stage, clusters are already spatially well-localized by earlier pruning and alignment. A simple distance metric reliably merges (i) nearby collinear segments and (ii) slightly misaligned clusters that lie along the same 3D edge, while avoiding shape/scale biases introduced by covariance-based distances. It is also cheaper and more stable numerically.

Parametric Fitting For each cluster we collect Gaussian centers and directions. Small clusters are fitted with a line (via direction-weighted PCA), while larger clusters are fitted with cubic Bézier curves obtained from ordered quantiles along the dominant PCA axis. Since a line is a degenerate Bézier, the two models are unified within the same parametric framework.

4. Evaluation

We benchmark EdgeSplats against prior edge reconstruction approaches, EdgeGaussians[4] and EMAP [13], on the synthetic *Replica* dataset [22]. For all Replica scenes, we obtain camera poses and an initial sparse point cloud using COLMAP applied to the raw RGB images. Furthermore, to demonstrate EdgeSplats' capability in real-world scenarios we evaluate against EdgeGaussians on the ETH3D [20] and Living Lab dataset [17], using poses and initial points, provided by the dataset authors.

4.1. Evaluation on Replica

We evaluate the performance on Rooms 0-2 of the Replica dataset, and report performance across several metrics.

Due to the absence of 3D ground truth for edge structures, we assess performance based on 2D reprojection accuracy. Specifically, we evaluate how well the reconstructed lines and curves reproject onto the PiDiNet edge mapped images [23]. We use a suite of metrics: (i) Chamfer distance, which captures geometric accuracy by computing the average distance between reprojected segments and image edges; (ii) SSIM, which quantifies structural similarity and sensitivity to misalignment or blurring; and (iii) a perceptual similarity score following [5], which combines CLIP [18] and LPIPS [36] to jointly capture semantic and low-level correspondence. We also report training times for the Gaussians, and edge fitting stages of EdgeSplats and EdgeGaussians (in the case of EMAP we include the training time of the unsigned distance function). All metrics are computed per image and averaged over the training set of a given scene.

Beyond image-space metrics, we quantify structural completeness by reporting the number and average length of reconstructed edge segments. While a higher raw count of short segments may indicate over-fragmentation, longer and fewer segments are generally indicative of coherent edge reconstruction.

Finally, to assess robustness, we perturb the initial COLMAP point cloud of Room0 with increasing levels of Gaussian noise and compare the performance of EdgeSplats and EdgeGaussians under identical conditions.

Replica Results We report our quantitative performance in Tab.1 and a full breakdown of training times in Tab.4.

Method	Scene	Chamfer↓	SSIM↑	CLIP↓	Time↓	N Segments	Ave. Length↑
EGS [4]	Room0	7.54	0.640	5.39	132:00	34289	0.0886
	Room1	21.7	0.746	5.11	78:30	21879	0.0828
	Room2	**5.98**	0.733	4.96	150:33	40405	0.0666
EMAP [13]	Room0	6.67	0.670	5.04	821:12	3369	0.0612
	Room1	11.71	0.767	4.80	774:27	2655	0.0583
	Room2	10.77	0.746	**4.72**	764:19	2629	0.0574
EdgeSplats	Room0	**4.69**	**0.678**	**5.02**	**25:34**	2511	**0.495**
	Room1	**7.83**	**0.78**	**4.72**	**25:02**	2147	**0.515**
	Room2	6.28	**0.761**	4.74	**23:09**	1658	**0.528**

Table 1. Performance metrics across scenes for each method. Values are averaged across all images in each dataset. Our method consistently demonstrates improved Chamfer and SSIM scores, competitive CLIP results, and significantly faster fitting with longer edge segments.

Under standard initialization, EdgeSplats achieves performance comparable to EdgeGaussians and EMAP, while remaining faster overall. Figure 2 shows that our method is more complete then EMAP, and less noisy than EdgeGaussians, with a small trade off in duplicate edge fragments. The elevated Chamfer score of EdgeGaussians on Room2 can be attributed to the large number of short line segments it produces. While this increases raw coverage and can create the impression of structural completeness (see Fig. 3 and supplementary material), the result lacks semantic and geometric consistency.

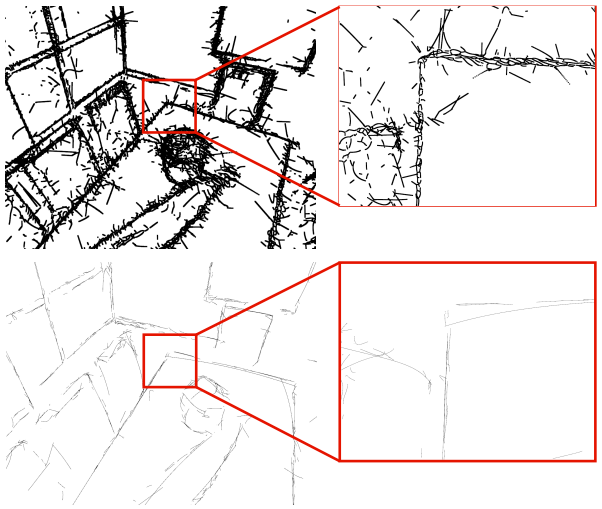

Figure 3. Close up results on the Room2 scene of the Replica dataset. Showing that when trained from COLMAP initialised poses, EdgeGaussians reconstructs the scene as many line fragments. Where as EdgeSplats, while still having some duplicates produces fewer, cleaner segments

The advantage of EdgeSplats becomes most evident under noisy initialization. Further results on Room0 with added noise are present in Tab. 2 and shown qualitatively in Fig. 4. When we gradually introduce noise into the ini-

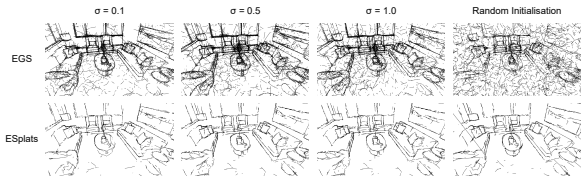

Figure 4. Visualisation of results in Tab.2. We show that the reconstruction provided by EdgeSplats remains consistent with increasing noise level on the initial points.

tialisation we show that our method maintains the same performance, whilst the performance of EdgeGaussians begins to degrade. We show that our method remains unaffected, even when the initialisation is pure noise, and continues to perform on par with the unperturbed results of EdgeGaussians. To simulate noise, we perturb the positions of the initial COLMAP point cloud with Gaussian noise of increasing magnitudes of standard deviation.

Scene	EdgeGaussians [4]			EdgeSplats		
	Chamfer↓	SSIM↑	CLIP↓	Chamfer↓	SSIM↑	CLIP↓
σ 0.1	7.23	0.648	5.40	**5.05**	**0.675**	**5.02**
σ 0.5	9.03	0.603	5.64	**4.65**	**0.679**	**5.01**
σ 1.0	10.3	0.584	5.79	**4.64**	**0.679**	**5.03**
Random init	20.2	0.400	6.30	**5.67**	**0.672**	**5.08**

Table 2. Performance metrics when adding noise to the initialization. Our method suffers minimal performance loss across all levels of perturbation, while EdgeGaussians struggles to maintain performance.

For our graph clustering heuristics on Replica scenes, we set the DBSCAN parameter, $\epsilon = 0.09$ for the first two stages of Gaussian clustering, and $\epsilon = 0.13$ for the line merging stage.

4.2. Evaluation on ETH3D

Building on the robustness analysis, we introduce a new benchmark using several scenes from the ETH3D dataset[20]. To the best of our knowledge, this constitutes the first *quantitative* evaluation of edge-based reconstruction methods on such real-world data. The dataset comprises in-the-wild scenes with diverse geometry, illumination, and texture. Training views range from 10–100 images per scene, posing challenging conditions for 3D edge reconstruction.

On many ETH3D scenes, EdgeGaussians exhibits substantial degradation, whereas EdgeSplats retains competitive performance despite increased scene complexity. Given the absence of ground-truth edge annotations, we adopt the same perceptual and structural measures as used in our Replica study - comparing to the PidiNet edge images used in training of the edges. This range of metrics provides baselines for perceptual similiarity, structural completeness, and accuracy.

Scene	EdgeGaussians [4]			EdgeSplats		
	Chamfer↓	SSIM↑	CLIP↓	Chamfer↓	SSIM↑	CLIP↓
Bridge	**31.7**	0.50	4.88	35.7	**0.62**	**4.22**
Living Room	65.2	0.69	4.39	**26.0**	**0.73**	**2.99**
Office	85.4	0.71	3.88	**40.3**	**0.80**	**2.20**
Pipes	47.5	0.62	4.24	**14.0**	**0.69**	**2.83**
Terrace	**15.9**	0.63	**3.29**	20.0	**0.67**	3.43
Terrace 2	9.65	**0.77**	**2.24**	**8.53**	0.74	2.32
Lounge	57.5	0.59	4.40	**13.1**	**0.67**	**3.11**
Door	26.2	0.50	4.71	**5.2**	**0.64**	**2.81**
Terrains	**9.19**	**0.73**	**2.44**	16.6	0.71	3.02
Average	38.7	0.64	3.83	**19.9**	**0.7**	**2.99**

Table 3. Performance metrics on a variety of scenes from the ETH3D dataset. Showing that on average EdgeSplats performs more consistently than EdgeGaussians.

Figure 5. Comparison of EdgeGaussians and EdgeSplats on the *'living room'* scene of ETH3D.

For ETH3D, we adjust the clustering thresholds to $\epsilon = 0.065$ for the two Gaussian clustering stages and $\epsilon = 0.1$ for line merging, while keeping the remainder of the pipeline unchanged. While these values could be hand tuned for a more optimal result on a per scene basis, we opt to keep them constant, to keep in line with our claims of robustness.

For EdgeGaussians, no guidelines exist for ETH3D or similar datasets; we therefore apply the same training protocol as used on Replica. EMAP is omitted, as the publicly released implementation does not support evaluation on custom datasets.

ETH3D Results While effective on some scenes, EdgeGaussians reconstructions are prone to visually collapsing into noise. This highlights the difficulty of transferring from synthetic to real-world data, whilst underscoring the robustness of EdgeSplats under such conditions.

We report quantitative results across a variety of scenes in Tab.3 and show qualitative results on the *'bridge'*, *'living room'* and *'office'* scenes (we refer the reader to the supplementary material for further results). While our

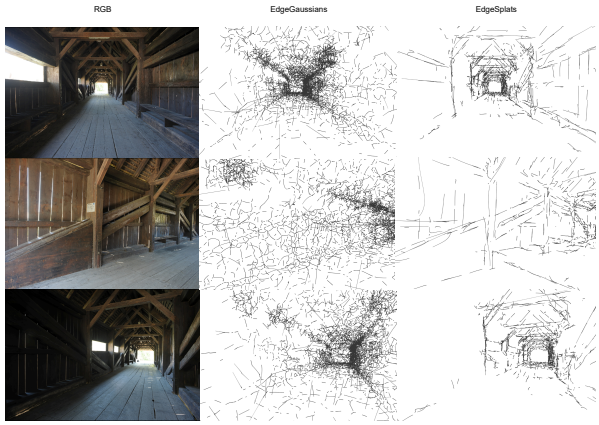

Figure 6. Comparison of EdgeGaussians and EdgeSplats on the *'bridge'* scene of ETH3D.

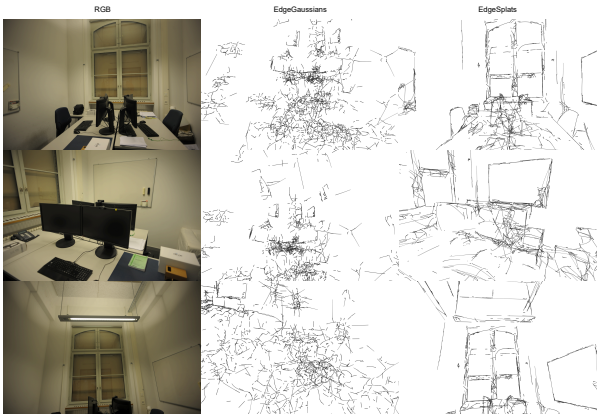

Figure 7. Comparison of EdgeGaussians and EdgeSplats on the *'office'* scene of ETH3D.

ETH3D reconstructions show higher levels of noise and fragmentation than on the Replica scenes, EdgeSplats on the whole achieves higher performance than EdgeGaussians. These results demonstrate that EdgeSplats narrows the gap between synthetic evaluation and real-world deployment, delivering reliable performance where earlier edge-based Gaussian methods tend to fragment or collapse

4.3. Evaluation on Living Lab

We provide further qualitative results on the *Living Lab* dataset [17], an additional example of an in-the-wild dataset captured using a standard mobile phone. COLMAP is used to estimate camera poses and generate initial points. As shown in Fig.8, EdgeSplats is able to reconstruct coherent edge structures in the scene, whereas EdgeGaussians suffers from the 'tangled' appearance present in the ETH3D scenes.

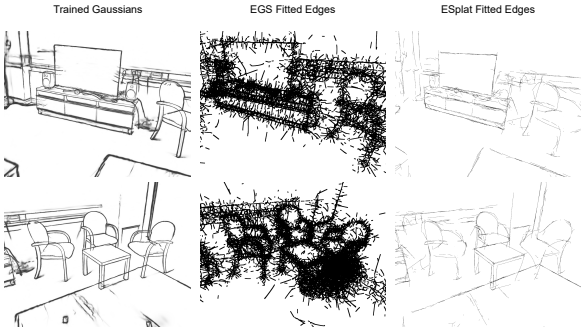

Figure 8. Preliminary results on the Living Lab dataset [17]. A dataset captured on an uncalibrated, handheld device, with estimated points and poses from COLMAP. Highlighting the contrast between EGS and EdgeSplats abilities to adapt to custom data.

Scene	EdgeGaussians [4]			EMAP [13]			Ours		
	T_{gauss}	T_{edges}	Total	T_{train}	T_{edges}	Total	T_{gauss}	T_{edges}	Total
Room0	17:57	114:04	132:00	804:43	16:29	821:12	25:48	**00:46**	**26:34**
Room1	12:25	66:05	78:30	762:54	11:33	774:27	24:24	**00:38**	**25:02**
Room2	24:57	125:37	150:33	764:19	18:26	782:45	22:31	**00:38**	**23:09**
Mean	18:26	101:55	120:21	771:10	15:26	792:48	24:14	**00:41**	**24:55**

Table 4. Timing comparison across Replica scenes (in minutes:seconds). While EdgeGaussians training time is slightly faster than ours, our edge fitting is dramatically faster, resulting in lower overall runtime.

4.4. Ablations

We conduct a series of ablation studies to validate the contributions of each stage in our pipeline. We first provide a breakdown of times of each stage of our pipeline, showing that our largest increase in speed comes from our updated clustering and edge fitting stage. Secondly, we investigate the effect of the number training iterations on edge-aligned Gaussians, and how this affects final clusters. Finally, we verify the use of the line merging stage against simply clustering intiail Gaussians.

Pipeline Times We show a breakdown of the overall training times in Tab.4. For EdgeSplats and EdgeGaussians, we break this into Gaussian training time and edge fitting time. For EMAP we include the time taken to train the UDF. While EdgeGaussians training time may be several minutes faster than ours, our line fitting method is on average **150 times faster**. This makes our overall pipeline **4-5 times** faster than EdgeGaussians, and **30 times** faster than the neural counterpart, EMAP.

Training iterations vs. Gaussians As a further ablation study, we demonstrate the effect of the number of Gaussian training iterations on the output (Fig.9). We show that while we train for 30k iterations, key structures form at around 7-

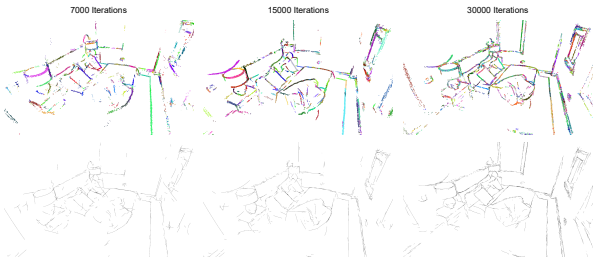

Figure 9. We show the effects on produced clusters when training the underlying gaussians for varying numbers of iterations. Lower iterations produce cleaner, but less complete reconstructions, while more iterations produce more complete, but noisier results.

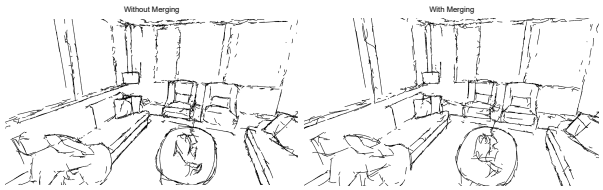

Figure 10. Effects of line merging stage. It can be seen that there are fewer duplicate edges, most notable around the table and window frames.

15k iterations. Clustering at these earlier stages produces more appealing segments, at the cost of less complete representations. We find that further training causes too much local variation in Gaussians, and thus find that 30k is a suitable compromise.

Line merging We compare line segments before and after merging "mega-Gaussians." The merging stage reduces fragmentation and consolidates collinear structures, improving the readability of the final topology. Figure 10 shows the changes are subtle, but noticeable, as it can be seen that there are fewer duplicate fragments along the 3D edges. While acceptable results can be obtained without merging, the operation consistently cleans noisy outputs.

4.5. Discussion and Future Work

We have shown that EdgeSplats performs as well as the two leading approaches, EdgeGaussians [4] and EMAP [13], in 3D edge reconstruction. We show that our method performs consistently against EdgeGaussians on in-the-wild data, and establish a new quantitative baseline for these types of scene. We acknowledge that in scenarios where the output of EdgeGaussians does not degrade into noise, results are still impressive. Where EdgeSplats truly excels is in its robustness to noise, faster runtime, and reduced reliance on carefully tuned hyperparameters. We believe EdgeSplats provides a practical and complementary option for 3D edge reconstruction, helping to advance a research area that remains comparatively underexplored.

Despite these strengths, there is still room for improvement. A common challenge across all methods—including ours—is the tendency for reconstructed edges to appear fragmented or over-segmented. This is largely due to the heavy reliance on, and noisy nature of the Gaussians produced during training on real-world data - a problem partially tackled in [29]. Although our training pipeline makes use of minimal modifications, and removes the need for directional losses between Gaussians, this design choice does not preclude future improvements. For instance, integrating a dynamic graph that evolves during training may help enforce global spatial or directional consistency and further reduce edge segmentation artifacts. Recent advances in neural SDFs [37] also suggest promising directions for retrieving more accurate edge Gaussians through hybrid neural–splatting approaches.

The fitted primitives are also not beyond further optimisation, as seen in [32] which continues to improve the lines and curves once initial estimates have been made.

5. Conclusion

In this work, we have presented EdgeSplats, a robust and efficient method for 3D edge reconstruction from in-the-wild 2D image data. By leveraging improvements made in [12], our approach is capable of producing meaningful reconstructions even under extreme noise conditions. A graph-based clustering framework enables rapid edge fitting, making EdgeSplats significantly faster and more resilient than existing state-of-the-art methods.

Quantitative evaluations demonstrate that EdgeSplats maintains high performance across a range of noisy initialisations, and varied datasets, while matching or exceeding the accuracy of baseline methods under standard conditions.

The robustness and efficiency of our approach make it particularly well-suited for real-world applications in robotics and SLAM, and we hope this work serves as a foundation for further advancements in Gaussian-based 3D edge reconstruction.

References

[1] Lizhi Bai, Chunqi Tian, Jun Yang, Siyu Zhang, Masanori Suganuma, and Takayuki Okatani. Rp-slam: Real-time photorealistic slam with efficient 3d gaussian splatting, 2024. 3

[2] Jonathan T. Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P. Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. *ICCV*, 2021. 2

[3] Jonathan T. Barron, Ben Mildenhall, Dor Verbin, Pratul P. Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. *CVPR*, 2022. 2

[4] Kunal Chelani, Assia Benbihi, Torsten Sattler, and Fredrik Kahl. Edgegaussians – 3d edge mapping via gaussian splatting, 2025. 1, 3, 5, 6, 7, 8, 11

[5] Changwoon Choi, Jaeah Lee, Jaesik Park, and Young Min Kim. 3doodle: Compact abstraction of objects with 3d strokes. *ACM Transactions on Graphics*, 43(4):1–13, 2024. 3, 5

[6] Pinxuan Dai, Jiamin Xu, Wenxiang Xie, Xinguo Liu, Huamin Wang, and Weiwei Xu. High-quality surface reconstruction using gaussian surfels, 2024. 3

[7] Antoine Guédon and Vincent Lepetit. Sugar: Surface-aligned gaussian splatting for efficient 3d mesh reconstruction and high-quality mesh rendering. *CVPR*, 2024. 3

[8] Peter Hedman, Pratul P. Srinivasan, Ben Mildenhall, Jonathan T. Barron, and Paul Debevec. Baking neural radiance fields for real-time view synthesis, 2021. 2

[9] Binbin Huang, Zehao Yu, Anpei Chen, Andreas Geiger, and Shenghua Gao. 2d gaussian splatting for geometrically accurate radiance fields. In *SIGGRAPH 2024 Conference Papers*. Association for Computing Machinery, 2024. 3

[10] Nikhil Keetha, Jay Karhade, Krishna Murthy Jatavallabhula, Gengshan Yang, Sebastian Scherer, Deva Ramanan, and Jonathon 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. 3

[11] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42 (4), 2023. 1, 3

[12] Shakiba Kheradmand, Daniel Rebain, Gopal Sharma, Weiwei Sun, Jeff Tseng, Hossam Isack, Abhishek Kar, Andrea Tagliasacchi, and Kwang Moo Yi. 3d gaussian splatting as markov chain monte carlo, 2025. 3, 8, 11

[13] Lei Li, Songyou Peng, Zehao Yu, Shaohui Liu, Rémi Pautrat, Xiaochuan Yin, and Marc Pollefeys. 3d neural edge reconstruction, 2024. 1, 3, 5, 7, 8, 11

[14] Shaohui Liu, Yifan Yu, Rémi Pautrat, Marc Pollefeys, and Viktor Larsson. 3d line mapping revisited, 2023. 1

[15] Hidenobu Matsuki, Riku Murai, Paul H. J. Kelly, and Andrew J. Davison. Gaussian splatting slam, 2024. 3

[16] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis, 2020. 1, 2

[17] Georgina Nuthall, Richard Bowden, and Oscar Mendez. The radiance of neural fields: Democratizing photorealistic and dynamic robotic simulation, 2024. 5, 7, 11

[18] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021. 5

[19] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. 2

[20] Thomas Schöps, Johannes L. Schönberger, Silvano Galliani, Torsten Sattler, Konrad Schindler, Marc Pollefeys, and Andreas Geiger. A multi-view stereo benchmark with high-resolution images and multi-camera videos. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 5, 6, 11

[21] X. Soria, E. Riba, and A. Sappa. Dense extreme inception network: Towards a robust cnn model for edge detection. In *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1912–1921, Los Alamitos, CA, USA, 2020. IEEE Computer Society. 11

[22] 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, 2019. 5, 11

[23] Zhuo Su, Wenzhe Liu, Zitong Yu, Dewen Hu, Qing Liao, Qi Tian, Matti Pietikäinen, and Li Liu. Pixel difference networks for efficient edge detection, 2021. 2, 3, 5, 11

[24] Edgar Sucar, Shikun Liu, Joseph Ortiz, and Andrew J. Davison. imap: Implicit mapping and positioning in real-time, 2021. 3

[25] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks, 2018. 2

[26] Hengyi Wang, Jingwen Wang, and Lourdes Agapito. Co-slam: Joint coordinate and sparse parametric encodings for neural real-time slam, 2023. 3

[27] Jingwen Wang, Tymoteusz Bleja, and Lourdes Agapito. Go-surf: Neural feature grid optimization for fast, high-fidelity rgb-d surface reconstruction, 2022. 3

[28] Nan Xue, Bin Tan, Yuxi Xiao, Liang Dong, Gui-Song Xia, Tianfu Wu, and Yujun Shen. Volumetric wireframe parsing from neural attraction fields. *CoRR*, abs/2307.10206, 2023. 1, 3

[29] Chenggang Yang and Yuang Shi. Linegs : 3d line segment representation on 3d gaussian splatting, 2024. 8

[30] Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman. Volume rendering of neural implicit surfaces. In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021. 2

[31] Yunfan Ye, Renjiao Yi, Zhirui Gao, Chenyang Zhu, Zhiping Cai, and Kai Xu. Nef: Neural edge fields for 3d parametric curve reconstruction from multi-view images, 2023. 1, 3

[32] Haiyang Ying and Matthias Zwicker. Sketchsplat: 3d edge reconstruction via differentiable multi-view sketch splatting, 2025. 3, 8

[33] Alex Yu, Sara Fridovich-Keil, Matthew Tancik, Qinhong Chen, Benjamin Recht, and Angjoo Kanazawa. Plenoxels: Radiance fields without neural networks, 2021. 2

[34] Alex Yu, Ruilong Li, Matthew Tancik, Hao Li, Ren Ng, and Angjoo Kanazawa. Plenoctrees for real-time rendering of neural radiance fields, 2021. 2

[35] Vladimir Yugay, Yue Li, Theo Gevers, and Martin R. Oswald. Gaussian-slam: Photo-realistic dense slam with gaussian splatting, 2024. 3

[36] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 5

[37] Wenyuan Zhang, Yu-Shen Liu, and Zhizhong Han. Neural signed distance function inference through splatting 3d gaussians pulled on zero-level set, 2024. 8

[38] Zihan Zhu, Songyou Peng, Viktor Larsson, Weiwei Xu, Hujun Bao, Zhaopeng Cui, Martin R. Oswald, and Marc Pollefeys. Nice-slam: Neural implicit scalable encoding for slam, 2022. 3

EdgeSplats: Robust 3D Edge Reconstruction for In-The-Wild Data

Supplementary Material

A. Implementation Details

Our Gaussian training follows the 3DGS-MCMC protocol [12], with scale and opacity regularization coefficients set to 0.01 and positional noise scale 5×10^5. For all scenes we cap the number of Gaussians at 5×10^5 and train for $30{,}000$ iterations. We supervise edge-aligned Gaussians with PiDiNet [23] edge maps (other detectors such as DexiNed [21] are also compatible).

Our GAT encoder uses three hidden layers of width 128 and an output dimension of 64. On Replica [22], DBSCAN uses $\varepsilon = 0.09$ for the first two passes (constituent Gaussians) and $\varepsilon = 0.13$ for the third pass (mega-Gaussian clustering). On ETH3D [20] and Living Lab [17], we use $\varepsilon = 0.065$ and $\varepsilon = 0.01$ for these stages, respectively.[1]

For Replica and Living Lab we initialize poses and points with COLMAP estimates to mimic in-the-wild capture; for ETH3D we use the poses and points provided by the dataset authors.

B. Additional Results

To provide full comparisons with EMAP [13] and EdgeGaussians [4], we include additional results on *room1* and *room2* of Replica (Figs. 11, 12). Our method achieves comparable completeness with noticeably lower noise, while running substantially faster. Under the described setup, EdgeGaussians often closely follows the supervised PiDiNet maps, yielding multiple fragmented segments rather than longer semantic structures, whereas EMAP shows incomplete regions and the longest training times.

We also present qualitative results on *terrace* and *door* from ETH3D [20]. In Fig. 13, EdgeGaussians reconstructs the general layout but exhibits spurious floating artifacts; Fig. 14 shows a failure case where EdgeGaussians collapses to noise. In contrast, EdgeSplats remains robust across both scenes without method-specific tuning.

[1] Reported ε values correspond to (passes 1–2, pass 3) in order.

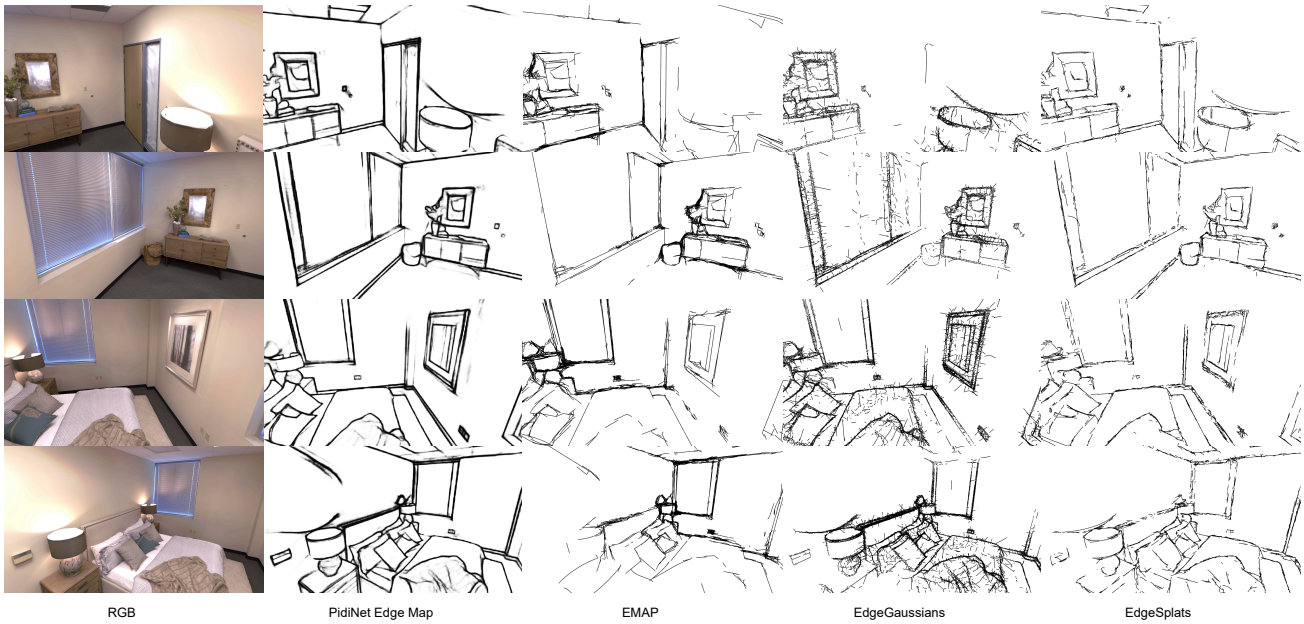

Figure 11. Qualitative results on *room 1* from the Replica dataset. EdgeSplats produces more complete renders than EMAP, whilst being less noisy than EdgeGaussians.

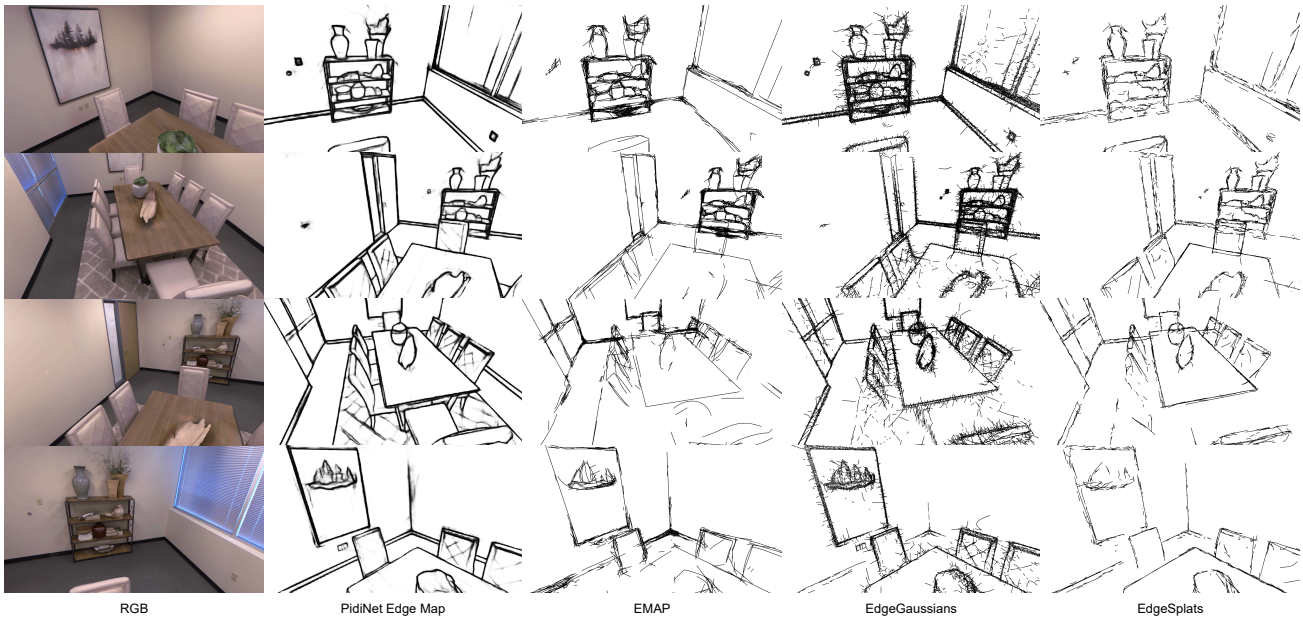

Figure 12. Qualitative results on *room 2* from the Replica dataset. EdgeGaussians may look more 'accurate' to the training images, however with close inspection this effect is from many fragmented edge sections.

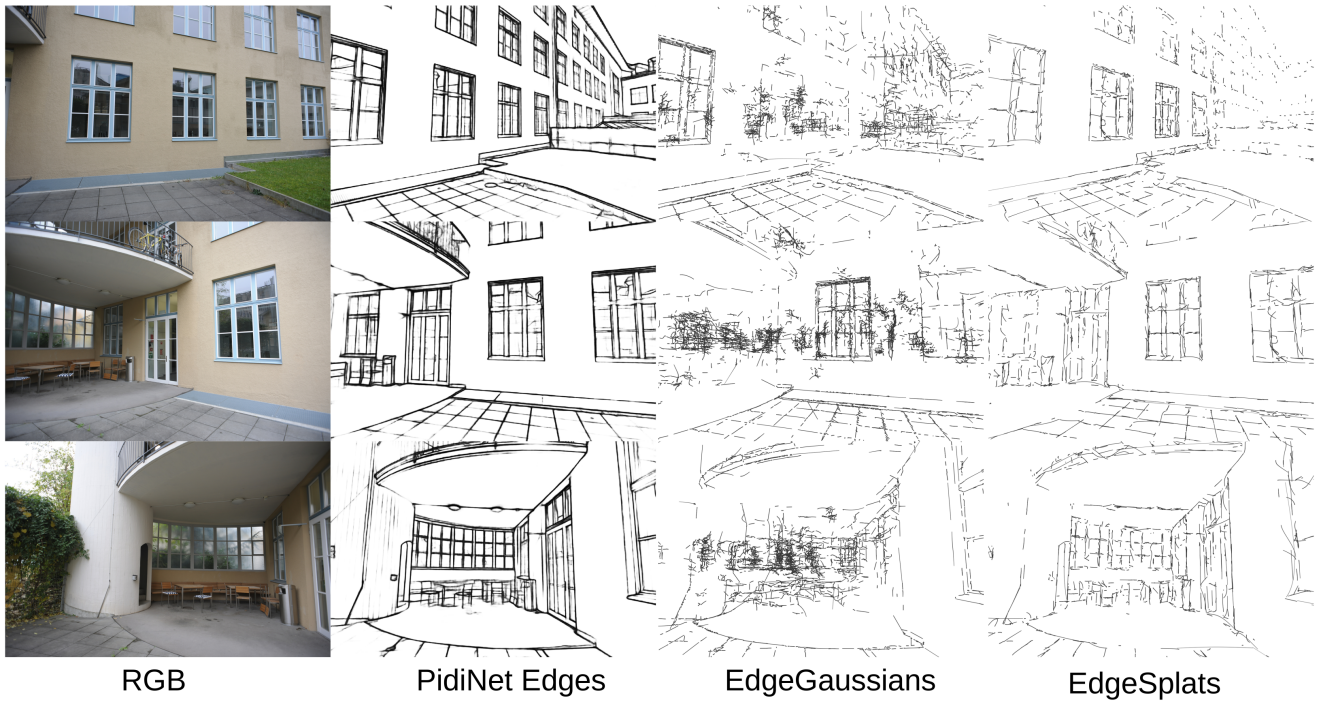

Figure 13. Results on the *terrace* scene from ETH3D, a more challenging scene than those provided in Replica. Both options produce semi-complete reconstructions, however EdgeGaussians suffers from less noise.

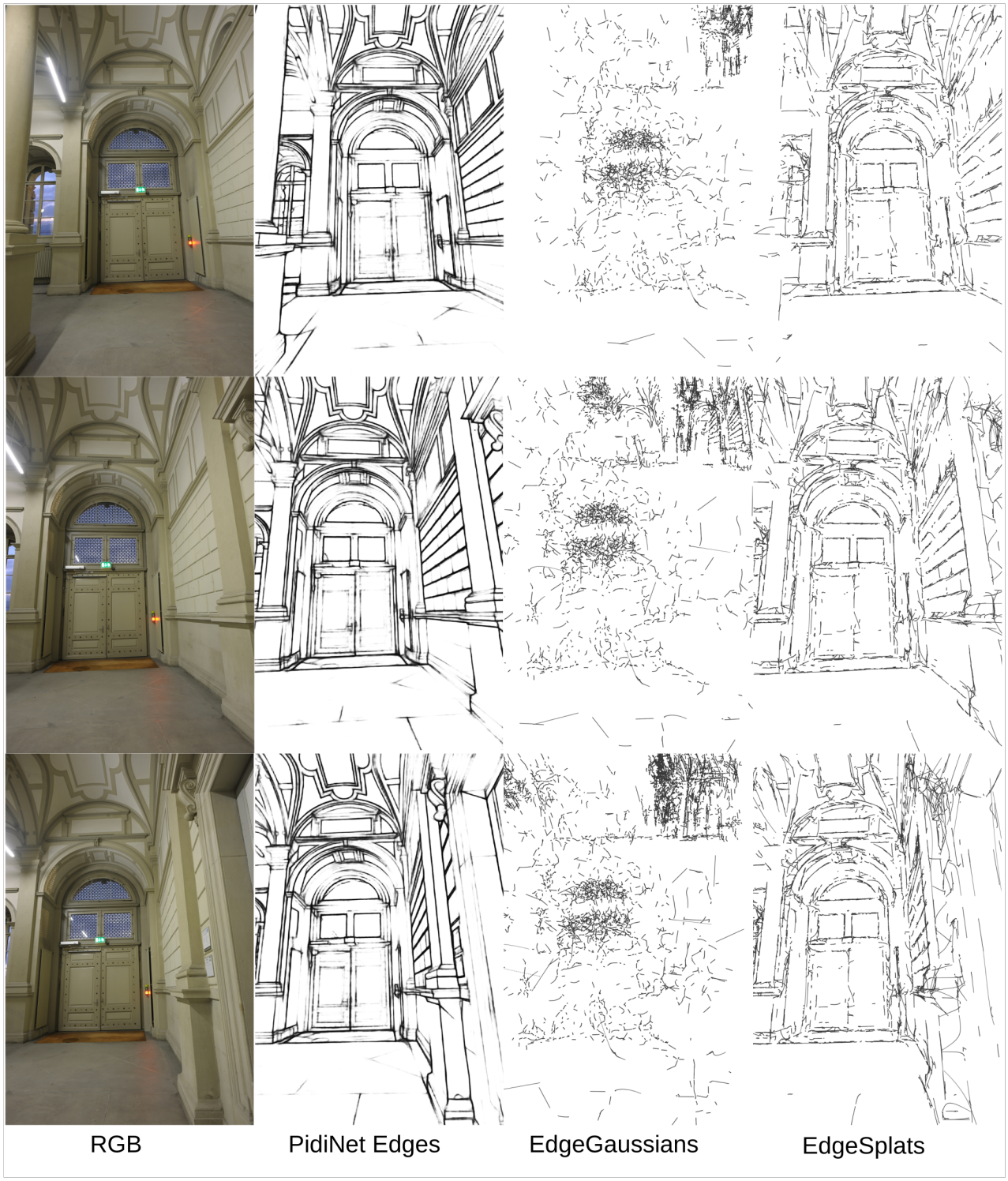

Figure 14. Results on the *door* scene from ETH3D, a scene composed of only 8 training views. EdgeGaussians falls into a degenerate configuration, whilst EdgeSplats remains robust.