OGS-SLAM: Hybrid ORB-Gaussian Splatting SLAM

Xiaohan Li
University of Science and Technology of China
Hefei, China
li2xh@mail.ustc.edu.cn

Wenxiang Shen
Tongji University
Shanghai, China
2010495@tongji.edu.cn

Dong Liu
University of Science and Technology of China
Hefei, China
dongeliu@ustc.edu.cn

Jun Wu
Fudan University
Shanghai, China
wujun@fudan.edu.cn

ABSTRACT

Traditional visual SLAM systems (dense and sparse) focus on building metric maps, but the internal representations are misaligned with human vision, making it insufficient for assisting robots in scene perception and interpretation. Conversely, aligning robot scene representation with human vision enables more intuitive human-to-robot commands and improves the generalization capability of deployed neural networks trained on natural images. Neural scene representation-based visual SLAM system, with its consistent and high-fidelity mapping, provides a novel way to assist robots in detailed scene depiction and comprehensive perception. However, end-to-end methods suffer from low accuracy in robot localization, which inevitably degrades mapping quality and limits their practical applications. In this paper, we propose a robust hybrid SLAM system, named OGS-SLAM, which integrates traditional visual SLAM with 3D Gaussian Splatting (3D GS) mapping. This system inherits the high localization accuracy of traditional SLAM while providing a scene model that aligns with human cognition, thereby offering a reliable foundation for downstream human-robot interaction tasks. Experiments demonstrate that our method outperforms state-of-the-art (SOTA) end-to-end SLAM systems in localization, mapping, and map semantic segmentation. Code will be available at: https://github.com/realXiaohan/OGS-SLAM.

KEYWORDS

SLAM, 3D Gaussian Splatting, Semantic Segmentation

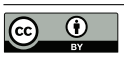
ACM Reference Format:
Xiaohan Li, Wenxiang Shen, Dong Liu, and Jun Wu. 2025. OGS-SLAM: Hybrid ORB-Gaussian Splatting SLAM. In *Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Detroit, Michigan, USA, May 19 – 23, 2025*, IFAAMAS, 9 pages.

1 INTRODUCTION

With the advancement of robotics, accurate mapping is crucial for tasks such as navigation, path planning, and scene exploration. Simultaneous Localization and Mapping (SLAM) collects data from on-board sensors and offers real-time scene representation while simultaneously estimating ego-motion within the map. The constructed internal representations store the geometric features of a scene, typically in the form of discrete point clouds. Although the point cloud maps assist robots in ego-motion estimation, they fail to support robots in performing high-level tasks. For instance, consider the task of instructing robots to make a cup of coffee. Traditional topological maps fail to provide sufficient support for such tasks because the significant disparity between the metric maps and human perception of the environment. Thus, there is an urgent need for a high-fidelity, anthropomorphic scene representation to better support high-level tasks in human-robot interaction. Scenes typically consist of various semantic objects, such as the 'cup' in the example. As a result, extracting semantic information from maps is fundamental to effective human-robot interaction. Currently, mainstream SLAM systems for scene representation are categorized into two types: metric maps and neural scene representations.

Based on the sparsity of scene representation, metric maps can be divided into sparse mapping and dense mapping. Sparse mapping [17, 22, 34] typically extracts feature points from video streams, storing optimized 3D keypoints and camera poses within the map. Feature points extraction is the core of pose estimation and optimization; however, it only utilizes a small portion of pixels, primarily concentrated in high-gradient areas, resulting in underutilization of rich scene information and low tolerance to textureless scenes. Dense mapping methods [13, 33] select pixels that surpass a certain threshold within image blocks to estimate the camera pose by minimizing photometric error. The selected 3D points are then stored on the map. Compared to sparse mapping, dense mapping involves more pixels, making it more effective in handling weak textures, repetitive areas, and edges. However, dense mapping heavily depends on the photometric consistency assumption, which may not always hold in practical applications. Although both feature-based and direct visual SLAM have been proven capable of effective mapping and localization in specific scenarios, they still produce discrete maps with relatively low quality. This limitation significantly reduces the effectiveness of SLAM in supporting high-level robot tasks in a human compatible manner.

Due to the excellent noise resistance and high-fidelity mapping, neural representation methods have been extensively researched in recent years [18, 25, 36], demonstrating significant potential for scene modeling in SLAM. Compared to metric maps, radiance-field-based mapping methods generate comprehensive and consistent scene representation. These methods directly backpropagate gradients inside scene representations, which is certainly beneficial



Proc. of the 24th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2025), Y. Vorobeychik, S. Das, A. Nowé (eds.), May 19 – 23, 2025, Detroit, Michigan, USA. © 2025 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

for high-level anthropomorphic tasks, such as semantic segmentation and scene editing [8, 9]. Neural mapping significantly outperforms metric maps in generating high-fidelity scene representations and 3D GS-based methods, with their advantage of fast rendering, perfectly meet the demands of SLAM, making them the top performance. However, mapping quality also heavily relies on the accuracy of ego-motion estimation. Given the real-time running requirements of SLAM and the limitations of the neural networks, current end-to-end SLAM [11, 18] systems simply rely on per-pixel RGB and depth loss for supervision and mistakenly optimize the camera ego-motion in vector space rather than in tangent space [30]. The rotation matrix is non-additive; therefore, directly performing gradients backpropagation in vector space inevitably introduces noise and reduce the robustness of the optimization process, which leads to a lower precision in camera pose estimation and consequently limits the accuracy of scene modeling. In contrast, sparse SLAM employs a well-designed and robust optimization framework for camera pose estimation. Inspired by this, we propose a hybrid SLAM system, named OGS-SLAM, which combines the strengths of localization in sparse SLAM with the high-fidelity mapping capabilities of 3D Gaussian Splatting (3D GS).

The proposed OGS-SLAM is designed to bridge the gap between robotic scene modeling and human visual perception. Our method enhances the quality of neural scene representation while maintaining high localization accuracy through the integration of sparse active SLAM and 3D Gaussian Splatting. In this work, we first leverage the initialized point cloud map from sparse SLAM to initiate the training of the 3D Gaussian Splatting map, thereby ensuring that the constructed map is both detailed and geometrically accurate. Second, our system fuses the keyframe camera poses generated from various Lie group-based bundle adjustment optimization frameworks with poses from 3D GS to refine the local Gaussian Splatting map for high-fidelity scene representation. Finally, we conducted semantic-level testing of the constructed maps in both real and synthetic indoor scenarios, demonstrating the potential of our method for practical applications.

2 RELATED WORK

2.1 Localization

Classic SLAM systems, whether sparse [4, 16, 22] or dense [6, 23, 32], primarily focus on improving the accuracy of localization through well-designed optimization algorithms. Ego-motion estimation can be viewed as a state estimation problem in mathematics [1]. Thus, early SLAM methods typically formulated an explicit motion model and predicted the robot's current state using filtering algorithms (e.g., Particle Filter [19], Extended Kalman Filter [24]). Then, the predictions of current camera pose are optimized with observation from sensors. However, the filter-based SLAM methods require an explicit distinction between the motion and observation models, which leads to low efficiency and robustness in the optimization process. Subsequent researchers formulated SLAM as a maximum a posteriori estimation (MAP) problem [3], treating both motion and observation models as factors and seamlessly incorporating them into the estimation process. Leveraging the sparsity of the Hessian matrix, the camera poses and landmark points are jointly optimized within a bundle adjustment (BA) framework [2] to provide accurate camera pose for the mapping process, which has become the mainstream approach in traditional SLAM systems.

Recently, numerous studies have focused on training methods for particular subproblems or modules within SLAM. ∇SLAM [10] implements several existing SLAM modules as differentiable computation graphs, enabling the backpropagation of reconstruction errors to sensor measurements. DeepFactors [5] jointly optimizes both pose and depth, adjusting the parameters of a learned depth basis during inference. DROID-SLAM [29] introduces a dense bundle adjustment layer that iteratively updates the residuals of inverse depths and camera poses using estimated optical flow and weights, leading to improved localization accuracy and making it the most practical end-to-end SLAM system. However, all end-to-end methods still fall short of classical sparse methods in robot localization.

2.2 Mapping

Classical SLAM systems initially model scene as a segmentation problem involving a grid of occupied space. These methods [7, 20] describe the scene as a lightweight graph that captures its topological structure. While the graph-based representation facilitates graph theory to assist robots in some tasks, merely storing the scene's topological structure results in significant information loss. Subsequent sparse metric maps [17, 22] constructed sparse point cloud map, which are particularly useful in optimal control and belief-space planning approaches. In contrast, dense metric maps [6, 23, 32], based on the assumption of grayscale invariance, divide images into blocks and extract points with significant gradients in each block as candidates for scene reconstruction. These methods jointly optimize photometric loss with geometric parameters to improve the accuracy of camera poses and scene representation. Although dense metric maps provide more detailed representations of the scene, including meshes [21] and occupancy grids [31], they remain fundamentally discrete representations of the environment.

Neural methods have garnered extensive attention due to their consistent and high-fidelity scene representation. iMAP [28] is the first work to integrate neural representation into SLAM. The mapping thread of iMAP optimizes camera poses and scene model of selected frames under the supervision of per-pixel error between the rendered and real images. NICE-SLAM [36] encodes the scene as multi-dimensional vectors within voxel feature grids and decodes the trained features into spatial occupancy and color. However, the aforementioned methods are based on NeRF which consumes long training times that cannot satisfy the real-time requirements of SLAM. Additionally, they require prior allocation of spatial grids for scenes, which limits the scalability of maps. In 2023, Kerbl et al. proposed 3D Gaussian splatting [12], which offers efficient rendering and high-fidelity scene representation. SplaTAM [11] represents as the first work introducing 3D GS into SLAM. SplaTAM generates the scene by utilizing per-pixel RGB-D loss with multiple frames insides a sliding window. Unlike SplaTAM, MonoGS [18] incorporates efficient online optimization and keyframe management techniques, keeping keyframes organized within a sliding window based on the inter-frame co-visibility. Furthermore, MonoGS performs resource allocation and employs pruning methods to eliminate unstable Gaussians, thereby reducing artifacts in the scene model.

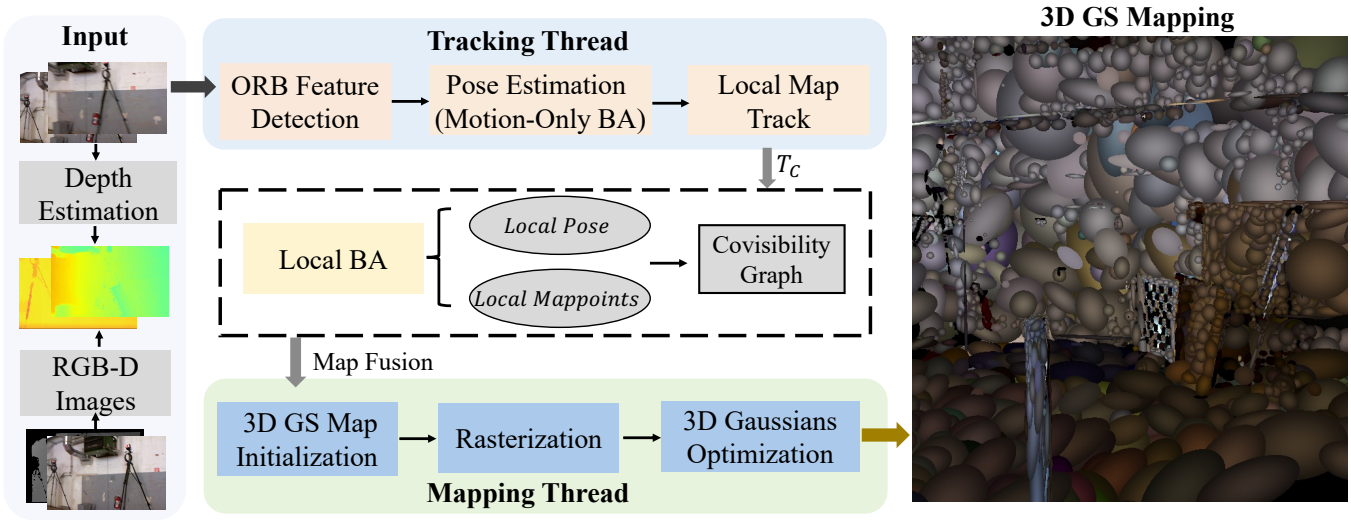


Figure 1: Overview of the Proposed OGS-SLAM. OGS-SLAM takes RGB-D images as input, the images will be firstly processed with depth estimation network if in monocular setting. In the tracking thread, ORB feature points are extracted from the input image, and a motion-only BA is employed to estimate the camera pose for current frame. A local map is constructed based on frame covisibility, and a local BA is performed to jointly optimize keyframe camera poses and 3D mappoints. In the mapping thread, the system is first initialized with the ORB-based point cloud map. Then, under the supervision of per-pixel RGB-D loss, the observed 3D Gaussians in the local map are further optimized to generate a high-quality scene representation.

3 METHOD

OGS-SLAM inherits the localization algorithms from ORB-SLAM2 while refining the tracking thread to align seamlessly with 3D GS. In the tracking thread, we primarily leverage the pose estimation and backend optimization strategies from ORB-SLAM2 while discarding the relocalization and loop closure modules to ensure system efficiency. In the initialization module, our method achieves high-fidelity scene mapping by fusing the local metric map from sparse SLAM with 3D Gaussian splatting scene representation. Additionally, OGS-SLAM improves the accuracy of camera pose estimation through the multi-view geometry pose optimization framework, combined with per-pixel loss supervision. A comprehensive illustration of OGS-SLAM is in Fig. 1.

3.1 Localization

The proposed OGS-SLAM supports both monocular video streams and RGB-D video streams as input. Notably, for monocular setting, we preprocess the RGB input by applying a monocular depth estimation network [15] to generate depthmaps. Similar to traditional sparse SLAM, OGS-SLAM is primarily divided into two parallel threads: tracking and mapping. The tracking thread extracts ORB feature points from input and output the optimized ego-motion estimation, which is robot localization. The mapping thread further optimizes the camera poses and generates a high-fidelity scene representation. In the tracking thread, ORB features are first extracted from each frame and establish a set of correspondences with ORB descriptors between consecutive frames. These correspondences are then used to estimate the camera pose $\mathbf{T}$ (Rotation $\mathbf{R} \in SO(3)$ and Translation $t \in \mathbb{R}^3$) by performing a motion-only bundle adjustment (BA). Motion-only BA simultaneously optimizes both camera pose $\mathbf{T}$ of current frame and 3D positions of feature points by minimizing the reprojection error between matched 3D points $\mathrm{X}^i \in \mathbb{R}^3$ and the correspondences $x^i \in \mathbb{R}^2$, where $i \in \mathcal{X}$ the collection of matches. The projected $\tilde{x}^i \in\in \mathbb{R}^2$ in pixel plane can be generated with the pinhole camera model:

$$\tilde{x}^i = \pi(\mathbf{T}_{cw_k}\mathrm{X}^i) = K(\mathbf{R}\mathrm{X}^i + \mathbf{t}), \tag{1}$$

where $\mathbf{T}_{cw_k} \in SE(3)$ is the camera pose projecting 3D world points to camera plane and k stands for kth frame, $\pi(\cdot)$ is the projection function and K is camera intrinsic. The projection error measures the distance of projected $\tilde{x}^i$ (marked as **measurement**) and correspondence x^i (marked as **observation**) can be expressed as:

$$e_i(\mathbf{T}_{cw_k}) = x^i - \tilde{x}^i. \tag{2}$$

By combining all $e_i \in \mathcal{X}$, the motion-only BA is:

$$\mathbf{T}_{cw_k} = \arg\min_{\mathbf{T}_{cw_k}} \sum_{i \in \mathcal{X}} \mathrm{H}(\cdot)\|x^i - \pi(\mathbf{R}\mathrm{X}^i + \mathbf{t})\|^2_\Sigma, \tag{3}$$

where $\mathrm{H}(\cdot)$ is Huber cost function and Σ the covariance matrix associated to the scale of keypoints. Parameterizing $\xi_C \in se(3)$ with Lie-algebra, the camera pose is optimized with the Levenberg-Marquardt method in tangent space. To maintain spatiotemporal consistency, a sliding window is applied to manage a set of covisible keyframes $\mathcal{K}_C$ and optimize the camera poses $\mathbf{T}_C$ along with all points $\mathcal{P}_C$ observed in those keyframes through local bundle adjustment. Keyframes within of the sliding window $\mathcal{K}_L$, which observes points in $\mathcal{K}_C$ but lack direct co-visibility with the active frames, also contribute to the loss function but remain fixed during the optimization process. This ensures that the optimization

focuses on local consistency while maintaining global constraints for long-term robust tracking. The local bundle adjustment is:

$$\{\mathrm{X}^i, \mathbf{R}_l, \mathbf{t}_l | i \in \mathcal{P}_C, l \in \mathcal{K}_C\} = \arg\min_{\mathrm{X}^i, \mathbf{R}_l, \mathbf{t}_l} \sum_{k \in \mathcal{K}_C \cup \mathcal{K}_L} \sum_{i \in \mathcal{X}_k} \mathrm{H}(\cdot) e(\mathbf{T}_{cw_k}, \mathrm{X}^i), \tag{4}$$

where $\mathcal{X}_k$ is the set of matches between points in $\mathcal{P}_C$ and keypoints in kth keyframe, $e(\mathbf{T}_{cw_k}, \mathrm{X}^i)$ can be calculated with equation 1 and equation 2. Similarly, by parameterizing $\xi_C \in se(3)$ with Lie algebra and leveraging the sparsity of the Hessian matrix, the local BA can be modeled as a factor graph to jointly optimize the camera pose and map points. The local keyframes and the observed 3D map points (marked as **landmarks**) are represented as vertices in the factor graph, with projection errors set up the corresponding edges. This facilitates efficient optimization of both camera poses and map landmarks within the sliding window.

3.2 3D Gaussian Mapping

3.2.1 ***3D Gaussian Representation***. OGS-SLAM utilizes a set of 3D Gaussian ellipsoids as the exclusive representation of the scene, denoted as $\mathcal{G}$. Each 3D Gaussian ellipsoid $\mathcal{G}^i$ contains both optical parameters (color c_i and opacity α_i) and pose (rotation $\Sigma^i_{\mathcal{W}} \in SO(3)$ and spatial position $\mu^i_{\mathcal{W}}$) parameters. During training, the 3D Gaussian ellipsoids are initially projected onto the 2D camera plane. Then, the optical and pose parameters are iteratively optimized under the supervision of per-pixel loss between the groundtruth and rendered RGB-D images generated through rasterization to achieve high-fidelity scene modeling. The projection from world space to the camera plane is expressed as:

$$f(x) = \alpha \exp(-\frac{\|x-\mu\|^2}{2r^2}), \tag{5}$$

where r is the radius. Rasterization is a core module of 3D Gaussian Splatting, making the entire rendering process fast and efficient. The rasterization process can be represented as follows:

$$\mu_I = \pi(\mathbf{T}_{cw} \cdot \mu_{\mathcal{W}}), \Sigma_I = \mathbf{J}\mathbf{W}\Sigma_{\mathcal{W}}\mathbf{W}^T\mathbf{J}^T, \tag{6}$$

where $\mathbb{G}(\mu_I, \Sigma_I)$ is the 2D Gaussians, $\mathbf{W}$ is the rotation matrix in $SO(3)$ and $\mathbf{J}$ is the Jacobian matrix which performs linear approximation of the projective transformation. With alpha blending, the color and depth can be renderrd which are directly compared with groundtruth to calculate the per-pixel loss. The loss is then back-propagated to iteratively optimize the parameters of all observed 3D Gaussians in current view. For more details, please refer to [12].

3.2.2 ***Map Fusion***. Most 3D Gaussian Splatting-based end-to-end SLAM systems rely on the depthmap from the first frame to initialize the 3D Gaussian ellipsoids. However, SLAM systems require sufficient translational movement to generate a robust initial scene representation. Relying solely on depthmaps from a few frames inevitably introduces significant optimization errors, leading to geometrically inaccurate scene representations. Thus, in the initialization module, our approach begins by utilizing a 3D map generated from ORB point cloud and RGB images to initiate the 3D Gaussian scene representation. This ensures that our initial representation is geometrically accurate and detailed. However, since ORB features are primarily extracted from high-gradient areas, such as regions with high-frequency details, and subsequent filtering methods marginalize incorrect matches, the point cloud generated from ORB features tends to be overly sparse, failing to adequately cover the scene structure. To ensure that the sparse point cloud used for the initialization process provides a relatively complete and detailed description of the scene, OGS-SLAM first divides the image into several blocks and evenly extracts ORB features within each block. Moreover, our method utilizes the densification scheme of 3D Gaussian Splatting, which increases the number of 3D Gaussians to enhance the scene representation during the training phase.

3.2.3 ***Mapping***. The mapping thread takes the camera pose from the tracking thread and further optimizes it with 3D Gaussian Splatting while generating a high-fidelity scene representation. First, based on the projective transformation described in equation 6 and the camera pose $\mathbf{T}_{cw}$ output by the tracking thread, the relationship between the observed 3D Gaussians in world coordinates and the 2D Gaussians in image plane is established. Then, by splatting and blending the observed 3D Gaussians, the synthesized color and depth of a pixel are computed as follows:

$$\begin{aligned} C_p &= \sum_{i=1}^{n} c_i \alpha_i \prod_{j=1}^{i-1}(1-\alpha_j), \\ D_p &= \sum_{i=1}^{n} d_i \alpha_i \prod_{j=1}^{i-1}(1-\alpha_j). \end{aligned} \tag{7}$$

Under the supervision of groundtruth RGB-D images, the mapping thread computes a per-pixel photometric loss L_{pho} and geometric loss L_{geo} between the rendered RGB-D images and the groundtruth.

$$\begin{aligned} L_{pho} &= \|I(\mathcal{G}, T_{CW}) - \tilde{I}\|_1, \\ L_{geo} &= \|D(\mathcal{G}, T_{CW}) - \tilde{D}\|_1, \end{aligned} \tag{8}$$

where $I(\mathcal{G}, T_{CW})$ is RGB rasterization, $D(\mathcal{G}, T_{CW})$ is depth rasterization, $\tilde{I}$ and $\tilde{D}$ are groundtruth RGB and depthmap. The total cost function in mapping thread is:

$$L_{mapping} = w_1 L_{pho} + w_2 L_{geo}, \tag{9}$$

where w_1 and w_2 are weights. During training, by minimizing the per-pixel error and backpropagating the gradient flow to all the observed 3D Gaussians, our method achieves precise camera pose estimation and high-fidelity scene reconstruction.

4 EXPERIMENTS

We conduct a comprehensive evaluation of OGS-SLAM in both real and synthetic datasets. The qualitative and quantitative results are provided to demonstrate the performance of OGS-SLAM. Finally, we conduct the experiments of semantic segmentation both on the rendered images and scene modeling, demonstrating the potential of assisting robots in executing tasks in a human-cognitive manner.

4.1 Datasets and Baselines

The proposed OGS-SLAM is verified on TUM-RGBD [27] and Replica [26] datasets. The TUM-RGBD dataset is a real-world indoor dataset captured with handheld cameras. The RGB images in the TUM-RGBD dataset suffer from significant lighting variations and motion blurs, and the depthmap often contains lots of missing holes. Additionally, TUM-RGBD features high-dynamic and large-scale scene

Table 1: Localization Accuracy ATE RMSE [cm]↓ on the TUM RGB-D dataset. Note that *fr2/large* is significant challenging for end-to-end SLAM systems. We conduct experiments on *fr2/large* to evaluate the robustness and scalability of different methods. Bold numbers indicate the best result.

Methods	fr1/desk	fr1/desk2	fr1/room	fr2/xyz	fr3/office	Avg.-5	fr2/large	Avg.-6
ORB-SLAM2	1.60	**2.20**	**4.70**	0.40	**1.00**	**1.98**	24.20	5.68
Vox-Fusion	3.52	6.00	19.53	1.49	26.01	11.31	196.34	42.15
NICE-SLAM	4.26	4.99	34.49	31.73	3.87	15.87	171.81	41.86
Point-SLAM	4.34	4.54	30.92	1.31	3.48	8.92	164.59	34.86
SplaTAM	3.35	6.54	11.13	1.24	5.16	5.48	204.40	38.64
MonoGS	**1.52**	5.10	6.30	1.58	1.65	3.23	165.14	30.22
Ours	1.71	2.80	5.82	**0.31**	1.62	2.45	**18.04**	**5.05**

Table 2: Quantitative Evaluation of Mapping on the TUM RGB-D dataset. Bold numbers indicate the best result.

Methods	Metrics	fr1/desk	fr1/desk2	fr1/room	fr2/xyz	fr3/office	fr2/large	Average
SplaTAM	PSNR ↑	22.00	**20.20**	20.24	24.50	21.90	16.62	20.91
	SSIM ↑	**0.86**	**0.79**	0.82	**0.95**	0.88	0.63	0.82
	LPIPS ↓	**0.23**	**0.27**	0.26	0.10	**0.20**	0.37	0.24
MonoGS	PSNR ↑	19.67	19.16	18.41	16.17	20.63	19.48	18.92
	SSIM ↑	0.73	0.66	0.64	0.72	0.77	0.72	0.71
	LPIPS ↓	0.33	0.48	0.51	0.31	0.34	0.38	0.39
Ours	PSNR ↑	**22.03**	20.07	**22.20**	**26.42**	**22.71**	**24.38**	**22.96**
	SSIM ↑	0.83	0.77	**0.87**	**0.95**	**0.91**	**0.85**	**0.86**
	LPIPS ↓	0.28	0.31	**0.23**	**0.09**	**0.20**	**0.26**	**0.23**

Table 3: Quantitative Evaluation of Mapping on the Replica dataset. Bold numbers indicate the best result.

Methods	Metrics	room0	room1	room2	office0	office1	office2	office3	office4	Average
NICE-SLAM	PSNR ↑	22.12	22.47	24.52	29.07	30.34	19.66	22.23	24.94	24.42
	SSIM ↑	0.69	0.76	0.81	0.87	0.89	0.80	0.80	0.86	0.81
	LPIPS ↓	0.33	0.27	0.21	0.23	0.18	0.23	0.21	0.20	0.23
SplaTAM	PSNR ↑	32.86	33.89	35.25	38.26	39.17	31.97	29.70	31.81	34.11
	SSIM ↑	**0.98**	**0.97**	**0.98**	**0.98**	**0.98**	**0.97**	**0.95**	**0.95**	**0.97**
	LPIPS ↓	0.07	0.10	0.08	0.09	**0.09**	0.10	**0.12**	0.15	0.10
Ours	PSNR ↑	**33.82**	**34.04**	**36.11**	**39.90**	**40.14**	**32.25**	**30.48**	**32.96**	**34.96**
	SSIM ↑	**0.98**	**0.97**	**0.98**	**0.98**	**0.98**	0.96	**0.95**	**0.95**	**0.97**
	LPIPS ↓	**0.06**	**0.09**	**0.06**	**0.06**	**0.09**	**0.09**	**0.12**	**0.14**	**0.09**

changes, making it challenging for end-to-end SLAM methods. In contrast, the Replica dataset is a synthetic indoor dataset that provides comprehensive RGB images and depthmaps. Since the Replica dataset is generated with dense meshes, the sequences tend to be smoother compared to those real captured. While the Replica dataset may not fully showcase the strengths of our method in long-term, large-scale scenarios, it remains an important benchmark, as most end-to-end methods are validated on this dataset. Therefore, we also provide both quantitative and qualitative results on the Replica dataset for reference. We compare the proposed OGS-SLAM against SOTA neural representation-based SLAM methods, primarily including NeRF-based methods [25, 35, 36] and 3D Gaussian Splatting-based methods [11, 18]. For baselines that provide specific results, we directly use the reported performance for comparison. For those that do not include results, we conduct each experiment three times and take the average as the final result.

4.2 Metrics

For robot localization evaluation, following [27], we calculate the Root Mean Square Error (RMSE) of the Absolute Trajectory Error (ATE) between camera poses estimation and trajectory groundtruth. For mapping quality evaluation, similar to SplaTAM [11] and MonoGS [18], we report PSNR, SSIM, and LPIPS for photometric rendering.

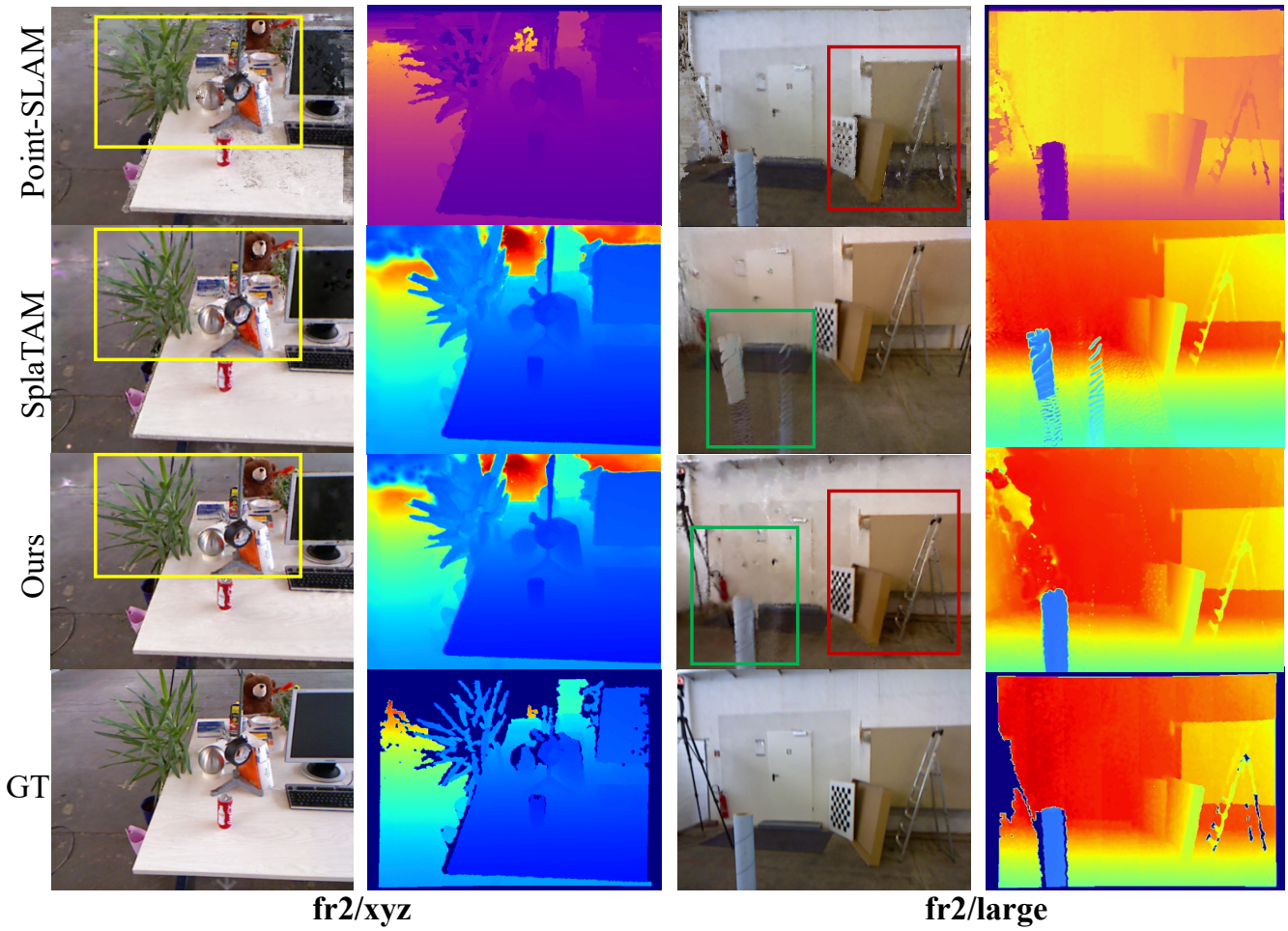


Figure 2: Visualization of Rendering Quality on the TUM RGB-D datasets. The yellow box in column 1 shows the rendering distortion of Point-SLAM and the erroneous splatting of SplaTAM, while our method achieves accurate rendering. The red and green boxes in column 3 highlight blurs and distortions from baselines, which our OGS-SLAM recovers more fine details.

4.3 Quantitative and Qualitative Evaluation

4.3.1 ***Localization Accuracy.*** We compared the performance of OGS-SLAM with SOTA neural representation-based SLAM and sparse SLAM in terms of localization accuracy on the TUM-RGBD dataset. The results are shown in Table. 1, where the methods between two dashed lines are NeRF-based SLAM, and those below the second dashed line are 3D Gaussian Splatting (3D GS)-based methods. As indicated in Table. 1, current NeRF-based SLAM methods perform worse in visual localization compared to 3D GS-based methods, even Point-SLAM uses the groundtruth depthmap as input. Our proposed OGS-SLAM demonstrates a **24%** improvement in visual localization (Avg.-5) compared to the SOTA end-to-end MonoGS. The *fr2/large* sequence, which spans two distinct scenes and involves significant lighting changes and intense camera movements, leads to substantial trajectory drift, making it particularly challenging for SLAM systems. Referring to Avg.-6, the proposed OGS-SLAM utilizes the the bundle adjustment framework and Lie group optimization to provide robust and accurate camera pose estimation, significantly improving visual localization accuracy in high-dynamic, large-scale scenarios compared to other SOTA approaches. Notably, due to the benefit of loop closure (global BA) and visual relocalization, ORB-SLAM2 outperforms our method in localization on certain sequences. However, this also introduces additional computation cost. In contrast, our method, aided by the supervision of per-pixel RGB-D loss in the mapping thread, performs better in localization on large-scale scenarios *fr2/large*.

4.3.2 ***Mapping Quality.*** Table. 2 and Table. 3 present the quantitative evaluation of mapping on TUM-RGBD and Replica datasets for the baselines and our OGS-SLAM. Regarding the TUM-RGBD dataset, Table. 2 demonstrates that our method achieves comparable rendering performance to SplaTAM on the *fr1/desk2* sequence, while on other sequences, it outperforms the SOTA end-to-end SLAM methods in rendering. Specifically, in high-dynamic and large-scale scene *fr2/large*, thanks to the accurate visual localization, our method shows almost **5dB** improvement in PSNR compared

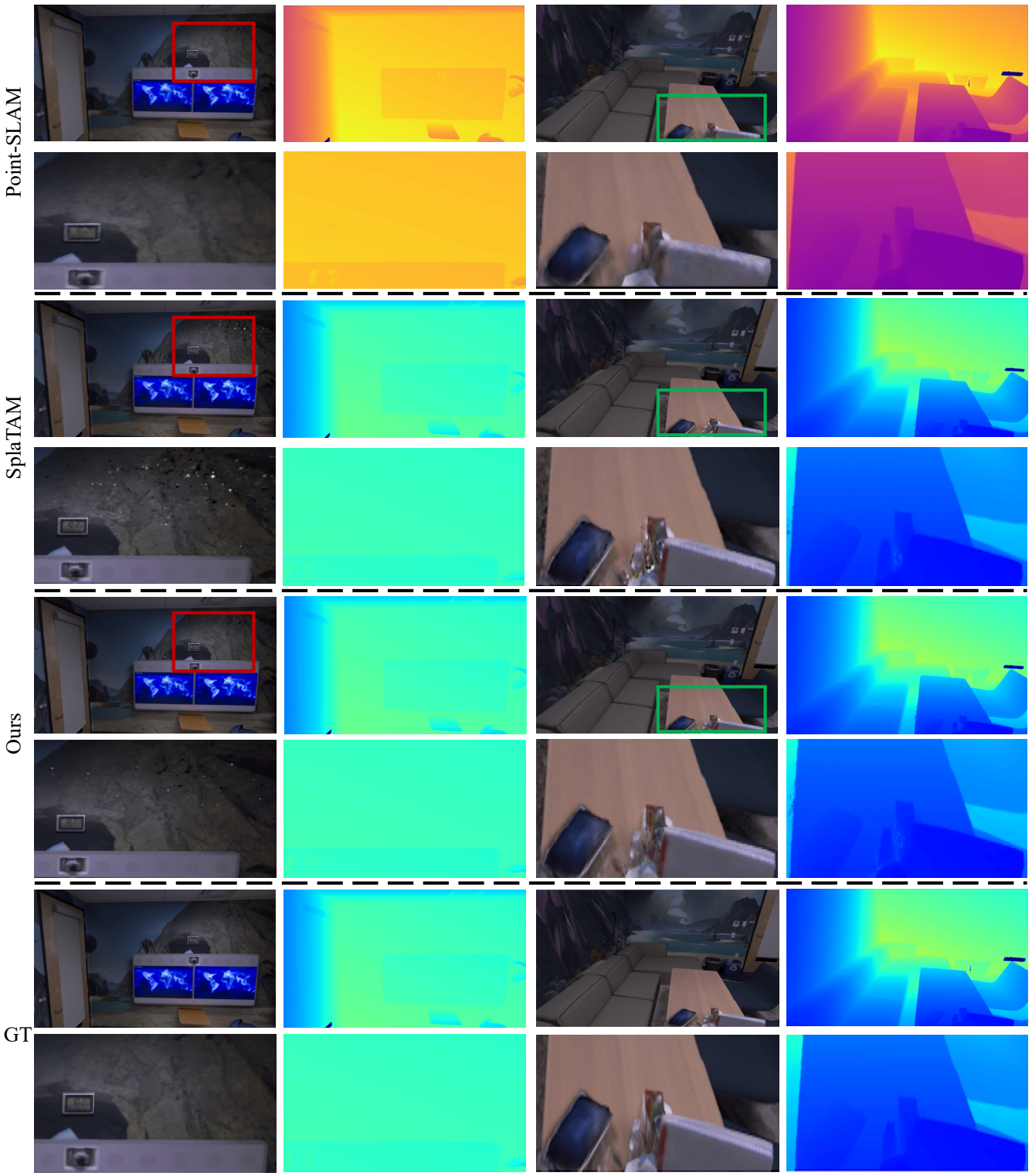


Figure 3: Visualization of Rendering Quality on the Replica dataset. The red and green boxes highlight that our method excels in recovering high-frequency details and exhibits fewer erroneous splats compared to Point-SLAM and SplaTAM.

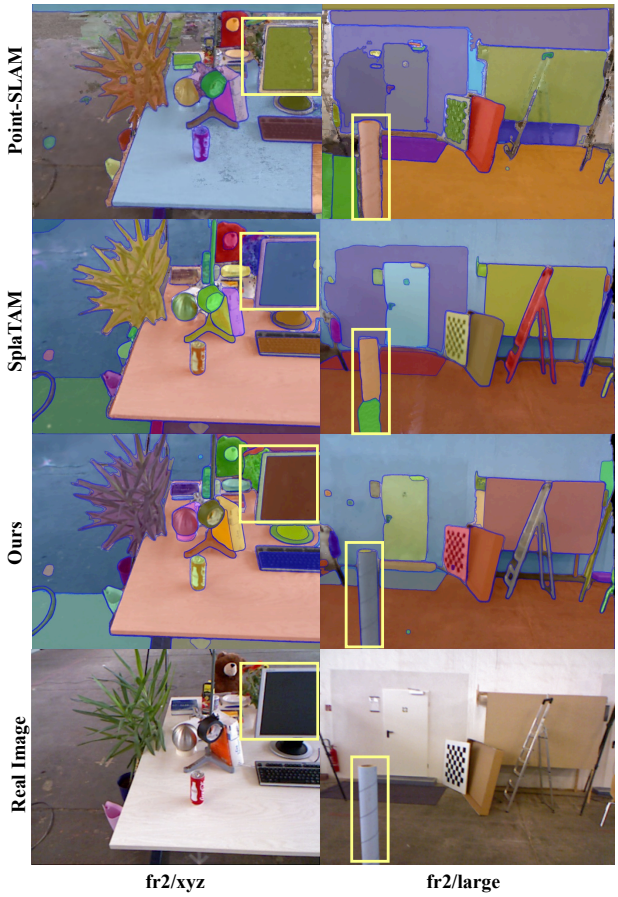


Figure 4: Semantic segmentation on Rendering Images.

to MonoGS. In summary, on the TUM-RGBD dataset, our method shows a PSNR improvement over **2dB** compared to SplaTAM.

Table. 3 provides the comparison of rendering performance between neural representation-based SLAM and our OGS-SLAM on the Replica dataset. Since the Replica dataset is synthetic and provides comprehensive and accurate RGB images and depthmaps, all neural representation-based methods achieve promising rendering performance. However, our method still shows a **0.85dB** improvement in PSNR compared to the SOTA method. Fig. 2 and Fig. 3 show the qualitative evaluation of mapping performance on both TUM-RGBD and Replica datasets. Fig. 2 shows the results of our OGS-SLAM and baselines on TUM-RGBD sequences with different scales. In the *fr2/xyz* sequence, Point-SLAM exhibits significant rendering errors in high-frequency details, particularly along the edges. SplatTAM, due to its lower localization accuracy, results in erroneous splatting and noticeable blur, such as the floaters around the plants highlighted in the yellow box. In contrast, the proposed OGS-SLAM, benefiting from accurate visual localization, produces high-fidelity scene details. In the challenging large-scale *fr2/large* sequence, although all methods suffer from performance degradation, our method still surpasses the baselines in scene details recovering. Fig. 3 presents the visualization of rendering quality on the Replica dataset, due to the high quality of this dataset, both the baselines and our method achieves promising results. Nonetheless, our method exhibits fewer erroneous splats and superior scene detail recovery compared to baselines.

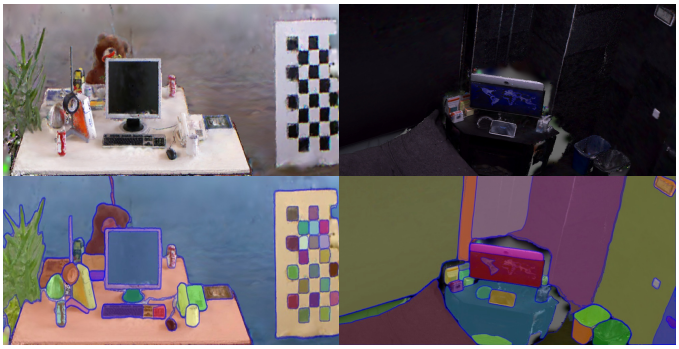


Figure 5: Visualization of Semantic Segmentation on the Mapping of OGS-SLAM.

4.3.3 ***Semantic Segmentation.*** Fig. 4 presents the results of semantic segmentation using the SAM model[14] on rendered real-world images generated by different methods. It is evident that Point-SLAM and SplaTAM renderings suffer from distortion, leading to inaccurate semantic segmentation, such as the computer screen and the white paper stick highlight in the yellow boxes. In contrast, our method produces more realistic renderings that yield accurate semantic segmentation. This not only reduces data storage cost for robots but also effectively mitigates the challenges associated with deploying neural networks trained on natural images. Fig. 5 shows the semantic segmentation of the map representation modeled by OGS-SLAM using SAM. The scene representation constructed by OGS-SLAM provides rich semantic information, which is able to enhance the robots scene perception and interpretation.

5 CONCLUSION

This paper presents a hybrid SLAM that combines traditional sparse SLAM with 3D Gaussian Splatting to bridges the gap between robot and human vision. The proposed OGS-SLAM inherits the accurate localization of sparse SLAM while achieving high-fidelity scene representation. The map fusion algorithm ensures a geometrically accurate scene representation during initialization. By utilizing motion-only and local bundle adjustment optimization frameworks, OGS-SLAM significantly improves visual localization accuracy compared to SOTA end-to-end SLAM systems and ORB-SLAM2. This robust visual localization further enhances the capability of 3D Gaussian Splatting to accurately and meticulously model large-scale scenes. The semantic segmentation results of the 3D GS scene representation demonstrate that the system's mapping aligns with human visual perception, presenting potential for robots to perform tasks in human understandable manner.

ACKNOWLEDGMENTS

This work was supported by National Key R&D Program of China under Grant 2020YFA0711400, National Natural Science Foundation of China (NSFC) under Grants U21A20452 and the scholarship supported by China Scholarship Council (CSC).

REFERENCES

[1] Josep Aulinas, Yvan Petillot, Joaquim Salvi, and Xavier Lladó. 2008. The SLAM problem: a survey. *Artificial Intelligence Research and Development* (2008), 363–371.

[2] Alvaro Parra Bustos, Tat-Jun Chin, Anders Eriksson, and Ian Reid. 2019. Visual SLAM: Why bundle adjust?. In *2019 international conference on robotics and automation (ICRA)*. IEEE, 2385–2391.

[3] Cesar Cadena, Luca Carlone, Henry Carrillo, Yasir Latif, Davide Scaramuzza, José Neira, Ian Reid, and John J Leonard. 2016. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on robotics* 32, 6 (2016), 1309–1332.

[4] Carlos Campos, Richard Elvira, Juan J Gómez Rodríguez, José MM Montiel, and Juan D Tardós. 2021. Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam. *IEEE Transactions on Robotics* 37, 6 (2021), 1874–1890.

[5] Jan Czarnowski, Tristan Laidlow, Ronald Clark, and Andrew J Davison. 2020. Deepfactors: Real-time probabilistic dense monocular slam. *IEEE Robotics and Automation Letters* 5, 2 (2020), 721–728.

[6] Jakob Engel, Thomas Schöps, and Daniel Cremers. 2014. LSD-SLAM: Large-scale direct monocular SLAM. In *European conference on computer vision*. Springer, 834–849.

[7] Leonardo Fermin-Leon, José Neira, and José A Castellanos. 2017. TIGRE: Topological graph based robotic exploration. In *2017 European Conference on Mobile Robots (ECMR)*. IEEE, 1–6.

[8] Lin Gao, Jie Yang, Bo-Tao Zhang, Jia-Mu Sun, Yu-Jie Yuan, Hongbo Fu, and Yu-Kun Lai. 2024. Mesh-based gaussian splatting for real-time large-scale deformation. *arXiv preprint arXiv:2402.04796* (2024).

[9] Xu Hu, Yuxi Wang, Lue Fan, Junsong Fan, Junran Peng, Zhen Lei, Qing Li, and Zhaoxiang Zhang. 2024. Semantic anything in 3d gaussians. *arXiv preprint arXiv:2401.17857* (2024).

[10] Krishna Murthy Jatavallabhula, Soroush Saryazdi, Ganesh Iyer, and Liam Paull. 2019. gradSLAM: Automagically differentiable SLAM. *arXiv preprint arXiv:1910.10672* (2019).

[11] Nikhil Keetha, Jay Karhade, Krishna Murthy Jatavallabhula, Gengshan Yang, Sebastian Scherer, Deva Ramanan, and Jonathon Luiten. 2024. SplaTAM: Splat Track & Map 3D Gaussians for Dense RGB-D SLAM. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 21357–21366.

[12] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 2023. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics* 42, 4 (2023), 1–14.

[13] Christian Kerl, Jürgen Sturm, and Daniel Cremers. 2013. Dense visual SLAM for RGB-D cameras. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2100–2106.

[14] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C Berg, Wan-Yen Lo, et al. 2023. Segment anything. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 4015–4026.

[15] Mykola Lavreniuk. 2024. SPIdepth: Strengthened Pose Information for Self-supervised Monocular Depth Estimation. *arXiv preprint arXiv:2404.12501* (2024).

[16] Xiaohan Li, Shiqi Lin, Meng Xu, Deyun Dai, and Jikai Wang. 2021. Po-SLAM: A novel monocular visual SLAM with points and objects. In *2021 4th International conference on artificial intelligence and big data (ICAIBD)*. IEEE, 454–458.

[17] Xiaohan Li, Dong Liu, and Jun Wu. 2024. CTO-SLAM: Contour Tracking for Object-Level Robust 4D SLAM. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 38. 10323–10331.

[18] Hidenobu Matsuki, Riku Murai, Paul HJ Kelly, and Andrew J Davison. 2024. Gaussian splatting slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 18039–18048.

[19] Michael Montemerlo, Sebastian Thrun, Daphne Koller, Ben Wegbreit, et al. 2003. FastSLAM 2.0: An improved particle filtering algorithm for simultaneous localization and mapping that provably converges. In *IJCAI*, Vol. 3. Citeseer, 1151–1156.

[20] Beipeng Mu, Matthew Giamou, Liam Paull, Ali-akbar Agha-mohammadi, John Leonard, and Jonathan How. 2016. Information-based active SLAM via topological feature graphs. In *2016 IEEE 55th Conference on decision and control (Cdc)*. IEEE, 5583–5590.

[21] Manasi Muglikar, Zichao Zhang, and Davide Scaramuzza. 2020. Voxel map for visual SLAM. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 4181–4187.

[22] Raul Mur-Artal and Juan D Tardós. 2017. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE transactions on robotics* 33, 5 (2017), 1255–1262.

[23] Richard A Newcombe, Steven J Lovegrove, and Andrew J Davison. 2011. DTAM: Dense tracking and mapping in real-time. In *2011 international conference on computer vision*. IEEE, 2320–2327.

[24] Juan Nieto, Tim Bailey, and Eduardo Nebot. 2006. Scan-SLAM: Combining EKF-SLAM and scan correlation. In *Field and Service Robotics: Results of the 5th International Conference*. Springer, 167–178.

[25] Erik Sandström, Yue Li, Luc Van Gool, and Martin R Oswald. 2023. Point-slam: Dense neural point cloud-based slam. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 18433–18444.

[26] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J Engel, Raul Mur-Artal, Carl Ren, Shobhit Verma, et al. 2019. The Replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797* (2019).

[27] Jürgen Sturm, Nikolas Engelhard, Felix Endres, Wolfram Burgard, and Daniel Cremers. 2012. A benchmark for the evaluation of RGB-D SLAM systems. In *2012 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 573–580.

[28] Edgar Sucar, Shikun Liu, Joseph Ortiz, and Andrew J Davison. 2021. imap: Implicit mapping and positioning in real-time. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 6229–6238.

[29] Zachary Teed and Jia Deng. 2021. Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras. *Advances in neural information processing systems* 34 (2021), 16558–16569.

[30] Zachary Teed and Jia Deng. 2021. Tangent space backpropagation for 3d transformation groups. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 10338–10347.

[31] Emanuele Vespa, Nikolay Nikolov, Marius Grimm, Luigi Nardi, Paul HJ Kelly, and Stefan Leutenegger. 2018. Efficient octree-based volumetric SLAM supporting signed-distance and occupancy mapping. *IEEE Robotics and Automation Letters* 3, 2 (2018), 1144–1151.

[32] Rui Wang, Martin Schworer, and Daniel Cremers. 2017. Stereo DSO: Large-scale direct sparse visual odometry with stereo cameras. In *Proceedings of the IEEE international conference on computer vision*. 3903–3911.

[33] Thomas Whelan, Stefan Leutenegger, Renato F Salas-Moreno, Ben Glocker, and Andrew J Davison. 2015. ElasticFusion: Dense SLAM without a pose graph.. In *Robotics: science and systems*, Vol. 11. Rome, Italy, 3.

[34] Shichao Yang and Sebastian Scherer. 2019. Cubeslam: Monocular 3-d object slam. *IEEE Transactions on Robotics* 35, 4 (2019), 925–938.

[35] Xingrui Yang, Hai Li, Hongjia Zhai, Yuhang Ming, Yuqian Liu, and Guofeng Zhang. 2022. Vox-fusion: Dense tracking and mapping with voxel-based neural implicit representation. In *2022 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, 499–507.

[36] Zihan Zhu, Songyou Peng, Viktor Larsson, Weiwei Xu, Hujun Bao, Zhaopeng Cui, Martin R Oswald, and Marc Pollefeys. 2022. Nice-slam: Neural implicit scalable encoding for slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 12786–12796.