

Article

Related Keyframe Optimization Gaussian–Simultaneous Localization and Mapping: A 3D Gaussian Splatting-Based Simultaneous Localization and Mapping with Related Keyframe Optimization

Xiasheng Ma ^{1,2}, Ci Song ^{2,3} , Yimin Ji ² and Shanlin Zhong ^{2,*} ¹ School of Mechanical Engineering, University of Science and Technology Beijing, Beijing 100083, China; b20190270@xs.ustb.edu.cn² The State Key Laboratory of Multimodal Artificial Intelligence Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China; songci@tongji.edu.cn (C.S.); jiyimin2021@ia.ac.cn (Y.J.)³ College of Electronic and Information Engineering, Tongji University, Shanghai 200092, China

* Correspondence: shanlin.zhong@ia.ac.cn

Abstract: Simultaneous localization and mapping (SLAM) is the basis for intelligent robots to explore the world. As a promising method for 3D reconstruction, 3D Gaussian splatting (3DGS) integrated with SLAM systems has shown significant potential. However, due to environmental uncertainties, errors in the tracking process with 3D Gaussians can negatively impact SLAM systems. This paper introduces a novel dense RGB-D SLAM system based on 3DGS that refines Gaussians through sub-Gaussians in the camera coordinate system. Additionally, we propose an algorithm to select keyframes closely related to the current frame, optimizing the scene map and pose of the current keyframe. This approach effectively enhances both the tracking and mapping performance. Experiments on high-quality synthetic scenes (Replica dataset) and low-quality real-world scenes (TUM-RGBD and ScanNet datasets) demonstrate that our system achieves competitive performance in tracking and mapping.



Academic Editor: Pedro Couto

Received: 6 December 2024

Revised: 23 January 2025

Accepted: 24 January 2025

Published: 27 January 2025

Citation: Ma, X.; Song, C.; Ji, Y.; Zhong, S. Related Keyframe Optimization Gaussian–Simultaneous Localization and Mapping: A 3D Gaussian Splatting-Based Simultaneous Localization and Mapping with Related Keyframe Optimization. *Appl. Sci.* **2025**, *15*, 1320. <https://doi.org/10.3390/app15031320>

Copyright: © 2025 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Keywords: dense visual SLAM; scene reconstruction; 3D Gaussian splatting; keyframe selection; RGB-D SLAM

1. Introduction

With the rapid development of mobile robots and autonomous driving, visual simultaneous localization and mapping (SLAM) has become a key technology. It allows these systems to perceive and localize themselves in unknown environments [1,2]. SLAM has made significant progress over the past decade. Classic SLAM algorithms based on point clouds [3–6], meshes [7,8], voxels [9–11], or truncated signed distance function [12–14] have been developed. Recently, 3D reconstruction technologies have advanced, particularly with methods like NeRF (Neural Radiance Field) [15–17] and 3DGS (3D Gaussian splatting) [18–20]. This has led to breakthroughs in SLAM research. These technologies allow for more accurate and realistic 3D reconstructions. They also improve localization and mapping using images or depth data. As a result, new SLAM systems based on 3D reconstruction technologies have emerged.

NeRF has been successfully applied to SLAM systems, demonstrating the profound impact of new 3D reconstruction methods on SLAM technology. NeRF excels in view synthesis, detailed 3D reconstruction, and the ability to handle complex lighting and material

effects [15]. Many NeRF-based SLAM frameworks leverage its strengths in generating fine-grained 3D scene representations [21–23], significantly improving localization and mapping accuracy in various scenes, and achieving more stable results. However, the use of ray-based volumetric rendering techniques to generate high-resolution rendered images significantly limits NeRF's rendering speed. Additionally, the implicit structure of NeRF, based on multilayer perceptron (MLP), lacks intuitiveness, which poses certain challenges for tasks such as scene understanding and editing [24].

Unlike NeRF, 3DGS uses differentiable volumetric representations that are unstructured and explicit [18]. Instead of ray-based volumetric rendering, 3DGS utilizes a projection-based approach to rasterization. This structure not only enables faster high-resolution rendering but also, due to its explicitness, makes scene editing and fine-tuning more convenient. Furthermore, 3DGS ensures a high rendering quality while achieving more efficient computation and greater flexibility, especially in applications that require rapid iteration and optimization. These advantages make 3DGS a better fit for SLAM systems, offering stable and efficient localization and mapping performance. Building on these advantages, many SLAM systems based on 3DGS have been proposed [25–28]. These systems leverage the strengths of 3DGS to design and optimize SLAM from different perspectives, achieving high-precision localization and rendering. However, in the 3DGS-based tracking process, the incompleteness of the scene map can lead to errors in the pose estimation of keyframes. Densifying the map with these erroneous poses may adversely affect its quality. To address this issue, we propose a novel SLAM system based on 3D Gaussians.

In this paper, we propose a novel SLAM system based on 3D Gaussians, which is called Related Keyframe Optimization Gaussian SLAM (RK-SLAM). In this system, the latest keyframe is not directly used to densify the existing Gaussian map. Instead, a sub-Gaussian is established and associated with the pose of the keyframe. This sub-Gaussian undergoes refinement following the keyframe pose. Only after optimization is the refined sub-Gaussian used to densify the existing scene map. Moreover, for better tracking and mapping, we design a related keyframe selection algorithm. The algorithm selects keyframes for optimization together through inter-frame co-visibility and spatial distance between frames to improve the performance of tracking and mapping. In summary, our contributions are as follows:

1. We propose an innovative SLAM system that leverages sub-Gaussian to refine the pose of keyframes and their associated Gaussian. This approach mitigates the impact of keyframe pose errors on densifying the existing map, ensuring a more accurate and robust mapping.
2. We introduce a novel keyframe selection algorithm that selects related keyframes to assist in fine-tuning, resulting in improved tracking and rendering quality.
3. The system has been evaluated on multiple RGB-D datasets, demonstrating competitive performance in both tracking and mapping.

2. Related Work

2.1. 3D Reconstruction

Three-dimensional reconstruction is a central problem in computer vision, with various methods proposed for 3D scene representation, including point clouds [29], meshes [30,31], voxels [32], and truncated signed distance functions [33]. These methods have driven significant advancements in SLAM, leading to the development of numerous SLAM systems [3–5,7–14,34]. Recently, new 3D scene representations, such as NeRF [15–17] and 3DGS [18–20], have outperformed traditional techniques. NeRF uses deep neural networks for high-quality view synthesis and detailed 3D reconstruction, while 3DGS

leverages explicit, differentiable volumetric representations with projection-based rasterization for faster rendering and easier scene editing. These innovations have enhanced 3D reconstruction accuracy and efficiency and further advanced SLAM systems, improving real-time performance, robustness, and accuracy.

2.2. NeRF-Based SLAM

Leveraging the advantages of NeRF, numerous SLAM systems have been developed. IMAP [35] is the first real-time SLAM system to use a continually trained MLP as the sole scene representation for a handheld RGB-D camera. It achieves efficient 3D mapping and tracking through keyframe and multiprocessing innovations. Its emergence has spurred the development of many NeRF-based SLAM systems. NICE-SLAM [36] leverages geometric priors to handle local information, enhancing the system's scalability and robustness for superior performance in large-scale scenes. Co-SLAM [37] combines a multi-resolution hash-grid with one-blob encoding, enabling rapid convergence and surface hole filling for a more robust SLAM performance. Point-SLAM [38] introduces neural anchor points to represent scenes, allowing for a dynamic adjustment of point density based on input image information. This approach reduces optimization time and lowers memory usage in less detailed regions. Although these methods optimize NeRF-based SLAM from various angles, they remain constrained by NeRF's implicit representation and ray-based rendering approach. This limitation hinders a more intuitive optimization of SLAM.

2.3. 3DGS-Based SLAM

Compared to NeRF-based SLAM, 3DGS-based SLAM not only benefits from training with various loss functions but also leverages 3DGS's explicit representation, allowing for more effective scene editing and optimization. SplatAM [27] utilizes a silhouette mask to capture scene density, quickly determines whether a region has been mapped previously, and expands the structured map by adding more Gaussian functions. GS-SLAM [26] employs an adaptive expansion strategy that adjusts existing regions from new perspectives by adding or removing noisy Gaussian volumes. This strategy dynamically extends 3D Gaussians to reconstruct the entire scene. MonoGS [28] derives the analytical Jacobian matrix of the camera extrinsics with respect to 3DGS, enabling optimization of the camera pose and scene geometry. CG-SLAM [39] introduces a novel uncertainty-aware 3D Gaussian with a high consistency and geometric stability for efficient computation. It also features a new depth uncertainty model to optimize the tracking process, improving both tracking efficiency and accuracy. RD-SLAM [25] employs a generalized iterative closest point for pose estimation, enhancing tracking accuracy and speed. It also introduces a dual keyframe selection strategy for dense scene processing, achieving high-quality reconstruction results.

3. Methods

This section provides a detailed explanation of our proposed SLAM system, and the overview is shown in Figure 1. The current frame first undergoes an initial pose prediction through Gaussian tracking. It is then evaluated to determine whether it qualifies as a keyframe for the scene. For keyframes, a new sub-Gaussian is constructed based on the point cloud projected from the RGB-D image and its pose. This sub-Gaussian enters the sliding sub-Gaussian window, transitioning to the fine-tuning state. To maintain a constant window size, the earliest sub-Gaussian awaiting fine-tuning is removed from the window and united with the already fine-tuned Gaussian map. Meanwhile, a related keyframe window is determined based on the pose of the current keyframe and its visible region. The keyframe is subsequently fine-tuned across this related keyframe, optimizing its pose

and the Gaussian map. Certain scene editing operations are simultaneously performed. Over time, these keyframes progressively densify the representation of the entire scene, ultimately yielding a dense 3D scene map.

Next, we provide a detailed explanation of each component of the system. Section 3.1 introduces the 3DGS-based scene representation for rendering and the differentiable splatting rasterized rendering method. Section 3.2 explains the 3DGS-based tracking process and keyframe selection strategy. In Section 3.3, we introduce the sliding sub-Gaussian window and the related keyframe selection algorithm in detail. Finally, Section 3.4 elaborates on the loss functions for mapping and some direct scene editing operations base on 3DGS.

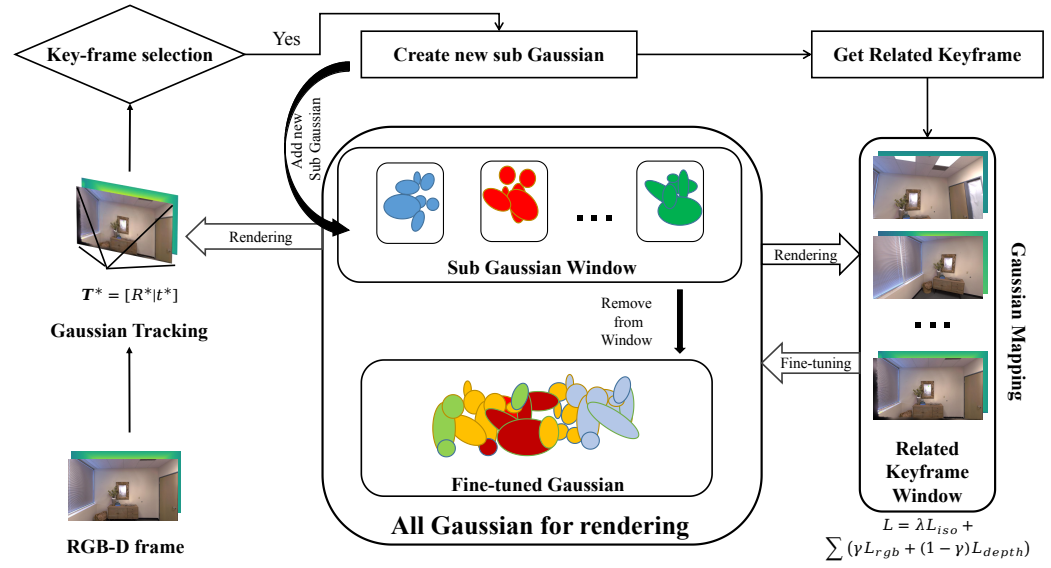


Figure 1. Overview of RK-SLAM.

3.1. 3D Gaussian Scene Representation

In RK-SLAM, for the map M representing the scene, in addition to a 3D Gaussian G_0 representing the refined scene, there is a sliding window W_s containing multiple groups of sub-Gaussians G_k awaiting fine-tuning as follows:

$$M = \left(\bigcup_{k \in W_s} G_k \right) \cup G_0. \quad (1)$$

For G_0 , there are N_0 3D Gaussians. Each 3D Gaussian contains a center $\mu \in \mathbb{R}^3$, a spatial covariance $\Sigma \in \mathbb{R}^{3 \times 3}$, an opacity $\alpha \in \mathbb{R}$, and a color c . c is represented by spherical harmonics for view-dependent radiance.

$$G_0 = \{G_i : (\mu_i, \Sigma_i, \alpha_i, c_i) \mid i = 1, \dots, N_0\}, \quad (2)$$

where $\Sigma_i = R_i S_i S_i^T R_i^T$, $S \in \mathbb{R}^3$ is a 3D scale vector, and $R \in \mathbb{R}^{3 \times 3}$ is the rotation matrix, which participates in computation as a 4D quaternion. Similarly, the 3D Gaussians in G_k also include these parameters. The key difference is that the parameters, particularly the centers μ' and rotation matrices R' , are based on the pose of the k -th keyframe. Therefore, the parameters of G_k also include the extrinsic matrix w_k of the k -th keyframe as follows:

$$\begin{aligned} \mathbf{G}_k &= \{\mathbf{w}_k, \{G_i : (\boldsymbol{\mu}'_i, \boldsymbol{\Sigma}'_i, \alpha_i, c_i) \mid i = 1, \dots, N_k\}\}, \\ \boldsymbol{\Sigma}'_i &= \mathbf{R}'_i \mathbf{S}_i \mathbf{S}_i^T \mathbf{R}'_i^T, \end{aligned} \quad (3)$$

where the extrinsic matrix $\mathbf{w}_k$ contains the a rotation matrix $\mathbf{Rot} \in \mathbb{R}^{3 \times 3}$ and a translation vector $\mathbf{Trans} \in \mathbb{R}^3$. It can be expressed as follows:

$$\mathbf{w}_k = \begin{bmatrix} \mathbf{Rot} & \mathbf{Trans}^T \\ \mathbf{0} & 1 \end{bmatrix}. \quad (4)$$

When rendering the map $\mathbf{M}$, the 3D Gaussians in $\mathbf{G}_k$ are transformed from the camera coordinate system to the world coordinate system using the extrinsic matrix $\mathbf{w}_k$. After aligning with the coordinate system of $\mathbf{G}_0$, they are united. The primary parameters involved in this transformation are the center coordinates $\boldsymbol{\mu}'_i$ and the rotation matrices $\mathbf{R}'_i$ as follows:

$$\begin{cases} \boldsymbol{\mu}_i = \mathbf{w}_k^{-1} \boldsymbol{\mu}'_i, \\ \mathbf{R}_i = \mathbf{Rot}^{-1} \mathbf{R}'_i, \end{cases} \quad i = 1, 2, \dots, N_k. \quad (5)$$

In practice, after they are united, the map $\mathbf{M}$ can be expressed as follows:

$$\begin{aligned} \mathbf{M} &= \{G_i : (\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i, \alpha_i, c_i) \mid i = 1, \dots, N\}, \\ N &= N_0 + \sum_{k \in W_s} N_k. \end{aligned} \quad (6)$$

The differentiable splatting rasterized rendering follows [18]. First, we need to project the 3D coordinates of the 3D Gaussians onto the 2D plane of the target camera using the following equations:

$$\begin{aligned} d &= (\mathbf{w}_{target} \boldsymbol{\mu})_z, \\ \boldsymbol{\mu}^{2D} &= \mathbf{K} \frac{\mathbf{w}_{target} \boldsymbol{\mu}}{d}, \\ \boldsymbol{\Sigma}^{2D} &= \mathbf{J} \mathbf{w}_{target} \boldsymbol{\Sigma} \mathbf{w}_{target}^T \mathbf{J}^T, \end{aligned} \quad (7)$$

where $\mathbf{K}$ is the intrinsic matrix of the target camera, $\mathbf{w}_{target}$ is the extrinsic matrix, $\mathbf{J}$ is the Jacobian of the affine approximation of the projective transformation, and d is the depth value on the z-axis after the 3D Gaussian center is projected onto the target camera coordinate system.

After projection, the influence of each 3D Gaussian on the camera plane can be calculated using the following Gaussian distribution equation:

$$f(\mathbf{x}) = \alpha \cdot \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{x} - \boldsymbol{\mu})\right). \quad (8)$$

The influence of all 3D Gaussians is accumulated sequentially from near to far based on their depth order after projection. By adding their colors and depths in this manner as Equation (9), the corresponding color C_p and depth D_p for each pixel in the image can be obtained.

$$\begin{aligned} C_p &= \sum_{i \in N} c_i f_i \prod_{j=1}^{i-1} (1 - f_j), \\ D_p &= \sum_{i \in N} d_i f_i \prod_{j=1}^{i-1} (1 - f_j). \end{aligned} \quad (9)$$

3.2. Gaussian Tracking

In Gaussian tracking, following [28], we adopt the proposed analytical Jacobian of the $SE(3)$ camera's extrinsic matrix with respect to the 3D Gaussians. For the loss function, we compute the losses for the RGB L_{rgb} and depth L_{depth} . Notably, similar to [28], we further optimize affine brightness parameters for exposure and penalize non-edge or low-opacity pixels. Consequently, the loss is expressed as follows:

$$\begin{aligned} L_{rgb} &= \| \exp(\alpha_{exp}) \hat{I} + \beta_{exp} - I \|_1, \\ L_{depth} &= \| \hat{D} - D \|_1, \end{aligned} \quad (10)$$

where $\hat{I}$ and $\hat{D}$ represent the rendered RGB image and depth image, respectively, while I and D denote the ground truth RGB image and depth image. α_{exp} and β_{exp} are exposure-related factors and are trainable parameters, typically initialized to 0. The final expression of the loss function is as follows:

$$L_{track} = \gamma L_{rgb} + (1 - \gamma) L_{depth}, \quad (11)$$

where γ is a hyperparameter.

After tracking, it is necessary to determine whether the current frame qualifies as a keyframe. Similar to [28], we measure the covisibility by measuring the intersection over the union of the observed Gaussians between the current frame and the last keyframe. If the covisibility drops below a threshold, or if the relative translation is large with respect to the median depth, the current frame is registered as a keyframe. The covisibility CoV_{ij} between frame i and frame j can be calculated as follows:

$$CoV_{ij} = \frac{|V_i \cap V_j|}{\min(|V_i|, |V_j|)}, \quad (12)$$

where V_i represents the observed Gaussians for frame i .

3.3. Keyframe Window

When a new keyframe is obtained, a sub-Gaussian, as described by Equation (3), is constructed. The point cloud is generated from the RGB-D image and downsampled at a certain ratio to initialize the sub-Gaussian. This sub-Gaussian is then added to the sliding keyframe window W_s . The new sub-Gaussian is based on the camera coordinate system corresponding to the current keyframe. As the pose of this keyframe changes during fine-tuning, the new sub-Gaussian adjusts accordingly, as shown in Figure 2. To maintain a constant size for W_s , the sub-Gaussian that exceeds the window limit is removed from W_s . The earliest sub-Gaussian undergoes the most fine-tuning, and its result is relatively stable (discussed in Section 4.3). Therefore, it is removed from the sliding window W_s . The removed sub-Gaussian is transformed into the world coordinate system using Equation (5) and then united with the already refined Gaussians G_0 .

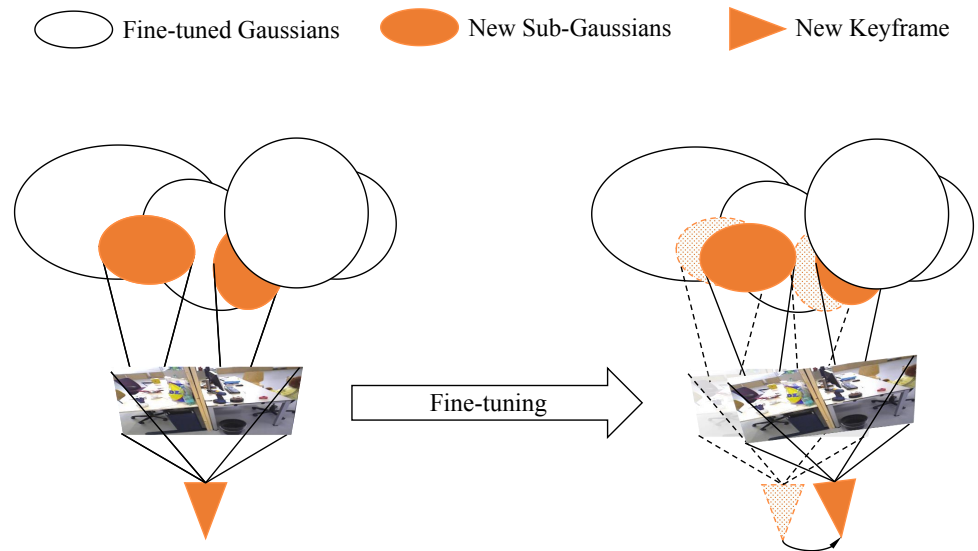


Figure 2. Sub-Gasussians.

At the same time, a related keyframe window W_r is also constructed. To ensure a strong correlation between the selected keyframes and the current keyframe, co-visibility is used as the metric for measuring inter-frame correlation. Additionally, to optimize the mapping of the current frame from a broader range of perspectives, the spatial distance between frames is also taken into consideration. Based on these factors, we design the related keyframe selection algorithm (Algorithm 1) as follows:

Algorithm 1 Related keyframe selection algorithm.

```

1: Input: keyframes  $W_{re} = [1, 2, \dots, n]$ , the observed Gaussians for all keyframes
    $V = [V_1, V_2, \dots, V_n]$ , the pose of all keyframes  $Ps = [p_1, p_2, \dots, p_n]$ , the current
   keyframe  $k$ , and the max size for the related keyframe window  $l$ 
2: Output: related keyframe window  $W_r$ 
3:  $CoV\_Scores \leftarrow [0, 0, \dots, 0]$ 
4: for  $i$  in  $W_{re}$  do
5:    $CoV\_Scores[i] \leftarrow CoV^2(V_k, V_i)$ 
6: end for
7:  $W_r \leftarrow []$ 
8:  $Dist \leftarrow [inf, inf, \dots, inf]$ 
9:  $last\_p \leftarrow k$ 
10: while  $len(W_r) < l$  do
11:   for  $i$  in  $W_{re}$  do
12:      $Dist[i] \leftarrow \min(dist(Ps[last\_p], Ps[i]), Dist[i])$ 
13:   end for
14:    $last\_p \leftarrow \arg \max_{i \in W_{re}} (Dist[i] \cdot CoV\_Scores[i])$ 
15:    $W_r \leftarrow W_r + [last\_p]$ 
16: end while
17: return  $W_r$ 

```

W_{re} contains the fine-tuned keyframes, V represents the Gaussians observed by all keyframes, and Ps denotes the 3D spatial positions of the keyframes.

In Algorithm 1, we first calculate the co-visibility between all fine-tuned keyframes and the current keyframe as Equation (12) using its square as the measure of consistency (line 5). Next, we iteratively update the nearest distances for the keyframes. This iteration prevents the selected keyframes from being overly concentrated in a specific region. By iterating, the selected keyframes are more evenly distributed across the 3D space. During each iteration, after updating the nearest distances (line 12), we compute the relation of

each keyframe as the product of its minimum distance and consistency score. Keyframes with the highest relations are added to W_r (line 14). This process continues until the number of keyframes in W_r meets the desired size, after which W_r is returned.

To provide a more intuitive demonstration of the related window, we extracted an example from the experiment, as shown in Figure 3. The figure shows the RGB images of the current frame and its related frames and their order in the SLAM sequence. From the figure, it is evident that the visible area of all related frames overlaps with the current frame. Furthermore, these related frames offer observations of the current scene from multiple views. This observation mode is beneficial for both the localization and mapping of the current frame. Notably, among these related frames, the earliest comes from the 192nd frame in the sequence, which is significantly distant from the current frame (the 1523rd) in terms of sequence order. Despite this, our algorithm is still able to capture their relation. This is of particular significance for SLAM sequences with cyclic structures.

The keyframes included in the two types of windows contribute to Gaussian mapping for optimizing the scene map. The key difference lies in how they are handled: the Gaussians in the sub-Gaussian corresponding to the keyframes in the sliding keyframe window undergo fine-tuning along with the pose of the keyframes, whereas the keyframes in the related keyframe window no longer has a corresponding sub-Gaussian.

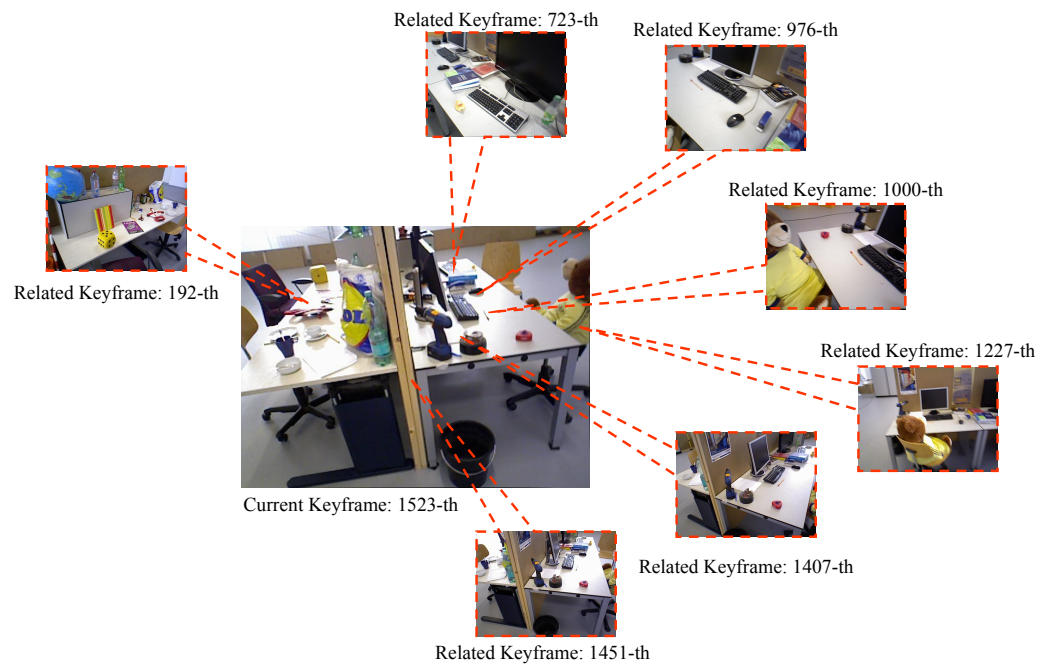


Figure 3. Example for related window.

3.4. Gaussian Mapping

In Gaussian mapping, in addition to the two windows mentioned above, we randomly select a subset W_{random} of keyframes during each iteration to participate in the computation of the mapping loss. W_{random} prevents the scene map from forgetting past observations. The mapping loss is calculated from three aspects: RGB loss L_{rgb} , depth loss L_{depth} , and isotropic regularization L_{iso} :

$$\begin{aligned}
 L_{rgb} &= (1 - \eta) \| (\exp(\alpha_{exp})\hat{I} + \beta_{exp}) - I \|_1 + \eta(1 - SSIM(\exp(\alpha_{exp})\hat{I} + \beta_{exp}), I), \\
 L_{depth} &= \| \hat{D} - D \|_1, \\
 L_{iso} &= \sum_{s_i \in M} \| s_i - \tilde{s}_i \|_1,
 \end{aligned} \tag{13}$$

where $SSIM(\cdot)$ refers to the structural similarity index measure [40], η is a hyperparameter used to control its weight, and $\bar{S}_i$ represents the mean of the scale factors S_i for the 3D Gaussians in the scene map M . The final loss is composed of the RGB and depth losses of the keyframes in all windows $W = W_s \cup W_r \cup W_{random}$ and scale factor isotropic regularization terms:

$$L = \lambda L_{iso} + \sum_{k \in W} (\gamma L_{rgb}^k + (1 - \gamma) L_{depth}^k). \quad (14)$$

Regarding scene editing, following [18,28], we densify the 3D Gaussians based on gradients and prune them based on their opacity and scale. Additionally, for resetting scene opacity, we adopt the same approach following [28].

4. Results

4.1. Experimental Setup

To evaluate the performance of RK-SLAM, we conducted experiments on the Replica dataset [41] (eight sequences), the TUM-RGBD dataset [42] (three sequences), and the ScanNet dataset [43] (five sequences). The Replica dataset consists of high-quality synthetic RGB-D images with a resolution of 1200×680 . And the TUM-RGBD dataset and ScanNet dataset contain low-quality real-world images captured by cameras and with a resolution of 640×480 , characterized by significant noise and blurriness. In particular, the depth images in the TUM-RGBD dataset and ScanNet dataset often suffer from information loss. We evaluated the effectiveness of our method on both types of datasets.

We compared RK-SLAM with state-of-the-art NeRF-based SLAM and 3DGS-based SLAM, including NICE-SLAM [36], Point-SLAM [38], Co-SLAM [37], SplaTAM [27], GS-SLAM [26], MonoGS [28], and CG-SLAM [39]. For RK-SLAM, the results were derived as the average of three experimental runs.

We evaluated the tracking and mapping performance of RK-SLAM separately. For tracking, we used the average absolute trajectory error (ATE RMSE) as the evaluation metric. For mapping, we employed the peak signal-to-noise ratio (PSNR), structural similarity index measure (SSIM), and learned perceptual image patch similarity (LPIPS) [44].

In the experiments, the maximum size of the sliding window W_s was set to 3, as discussed in detail in Section 4.3. Additionally, inspired by MonoGS [28], we limited the maximum size of the related frame window W_r to 7 to prevent excessive computational overhead caused by an increasing number of keyframes. This ensured that the total keyframe window ($W_s + W_r$) size remained at a maximum of 10, consistent with MonoGS. Meanwhile, the maximum size of the random keyframe window W_{random} was set to 2. For loss computation, γ was set to 0.9, η to 0.05, and λ to 10. RK-SLAM was implemented in Python (v.3.7) using the PyTorch (v.1.12.1) framework, incorporating CUDA code for 3DGS. We ran our SLAM on a server with AMD EPYC 7V12 and NVIDIA GeForce RTX 3090 (Santa Clara, CA, USA).

4.2. Quantitative Evaluation

4.2.1. Tracking Performance

Table 1 presents a comparison of RK-SLAM with the NeRF-based and 3DGS-based SLAM algorithms on the TUM-RGBD dataset. Evidently, RK-SLAM achieved state-of-the-art performance, outperforming the second-best approach by 0.25 cm on average. Notably, in the fr3/Office sequence, our method outperformed the second-best approach by 0.34 cm (approximately a 23% improvement), which is a significant margin. Table 2 shows the performance of RK-SLAM on the ScanNet dataset. Although its performance on ScanNet was not as strong as on the TUM-RGBD dataset, it still achieved good results on

some sequences, securing a place in the top three in terms of average performance. This demonstrates the potential of RK-SLAM.

Table 1. Camera tracking results on TUM-RGBD dataset (ATE RMSE [cm], the smaller, the better). The table marks the **first**, **second**, and **third** best results.

Methods	fr1/desk	fr2/xyz	fr3/Office	Avg.
NICE-SLAM	2.70	1.80	3.00	2.50
Co-SLAM	2.40	1.70	2.40	2.17
Point-SLAM	4.34	1.31	3.48	3.04
GS-SLAM	3.30	1.30	6.60	3.73
SplaTAM	3.35	1.24	5.16	3.25
CG-SLAM	2.43	1.20	2.45	2.03
MonoGS	1.50	1.44	1.49	1.47
RK-SLAM (Ours)	1.30	1.21	1.15	1.22

Table 3 compares RK-SLAM with the NeRF-based and 3DGS-based SLAM algorithms on the Replica dataset. RK-SLAM, again, achieved the best performance. Although the margin over the second-best result was only 0.01 cm, RK-SLAM achieved the best performance in five out of eight sequences, significantly outperforming the second-best method, which ranked first in only two sequences. Moreover, it is noteworthy that the standard deviation of RK-SLAM across all sequences was 0.037, which was significantly better than the second-best at 0.058 and the third-best at 0.212. This indicates that RK-SLAM delivers more stable performance across different sequences.

Table 2. Camera tracking results on ScanNet dataset (ATE RMSE [cm], the smaller, the better). The table marks the **first**, **second**, and **third** best results.

Methods	Sc.0059	Sc.0106	Sc.0169	Sc.0181	Sc.0207	Avg.
NICE-SLAM	14.00	7.90	10.90	13.40	6.20	10.48
Co-SLAM	12.29	10.9	6.62	13.43	7.13	10.07
Point-SLAM	8.29	11.86	22.16	14.77	9.54	13.32
SplaTAM	10.10	17.72	12.08	11.10	7.46	11.69
CG-SLAM	7.46	8.88	8.16	11.60	5.34	8.31
RK-SLAM (Ours)	6.62	12.59	12.44	11.88	8.14	10.33

Table 3. Camera tracking results on Replica dataset (ATE RMSE [cm], the smaller, the better). The table marks the **first**, **second**, and **third** best results.

Methods	R0	R1	R2	O0	O1	O2	O3	O4	Avg.
Point-SLAM	0.61	0.41	0.37	0.38	0.48	0.54	0.69	0.75	0.52
GS-SLAM	0.48	0.53	0.33	0.52	0.41	0.59	0.46	0.7	0.50
SplaTAM	0.31	0.40	0.29	0.47	0.27	0.29	0.32	0.55	0.36
MonoGS	0.33	0.22	0.29	0.36	0.19	0.25	0.12	0.81	0.32
CG-SLAM	0.29	0.27	0.25	0.33	0.14	0.28	0.31	0.29	0.27
RK-SLAM (Ours)	0.27	0.24	0.25	0.26	0.19	0.24	0.32	0.28	0.26

In summary, we evaluated the performance of RK-SLAM on two types of datasets, and it consistently achieved competitive results. Notably, compared to CG-SLAM, RK-SLAM demonstrated a comparable performance on the Replica dataset while significantly outperforming it on the TUM-RGBD dataset. The results clearly highlight the robustness of RK-SLAM with depth.

4.2.2. Rendering Performance

Tables 4 and 5 present a comparison of the mapping quality on the TUM-RGBD and Replica datasets, respectively. As shown in Table 4, RK-SLAM achieved comparable results. Compared to the top-performing SplaTAM algorithm, RK-SLAM demonstrated a competitive performance and even surpassed it in the fr1/desk sequence. As shown in Table 5, RK-SLAM achieved state-of-the-art results on the Replica dataset. Although it lags slightly in the SSIM metric, it shows significant advantages in the PSNR and LPIPS metrics. In the PSNR metric, RK-SLAM achieved the best results in six out of the eight sequences, outperforming other methods by a considerable margin. Similarly, for the LPIPS metric, RK-SLAM secured the best results in four out of the eight sequences.

Table 4. Map quality on TUM-RGBD dataset. ↑ means that the bigger the metric, the better, and ↓ means that the smaller the metric, the better. The table marks the first, second, and third best results.

Methods	Metrics	fr1/desk	fr2/xyz	fr3/Office	Avg.
Point-SLAM	PSNR ↑	13.79	17.62	18.29	16.57
	SSIM ↑	0.625	0.710	0.749	0.695
	LPIPS ↓	0.545	0.584	0.452	0.527
SplaTAM	PSNR ↑	21.49	25.06	21.17	22.57
	SSIM ↑	0.839	0.950	0.861	0.883
	LPIPS ↓	0.255	0.099	0.221	0.192
RK-SLAM (Ours)	PSNR ↑	22.31	22.47	20.67	21.82
	SSIM ↑	0.741	0.729	0.710	0.727
	LPIPS ↓	0.254	0.220	0.251	0.242

Table 5. Map quality on Replica dataset. ↑ means that the bigger the metric, the better, and ↓ means that the smaller the metric, the better. The table marks the first, second, and third best results.

Methods	Metrics	R0	R1	R2	O0	O1	O2	O3	O4	Avg.
NICE-SLAM	PSNR ↑	22.12	22.47	24.52	29.07	30.34	19.66	22.23	24.94	24.42
	SSIM ↑	0.689	0.757	0.814	0.874	0.886	0.797	0.801	0.856	0.809
	LPIPS ↓	0.330	0.271	0.208	0.229	0.181	0.235	0.209	0.198	0.233
Co-SLAM	PSNR ↑	27.27	28.45	29.06	34.14	34.87	28.43	28.76	30.91	30.24
	SSIM ↑	0.910	0.909	0.932	0.961	0.969	0.938	0.941	0.955	0.939
	LPIPS ↓	0.324	0.294	0.266	0.209	0.196	0.258	0.229	0.236	0.252
Point-SLAM	PSNR ↑	32.40	34.08	35.50	38.26	39.16	33.99	33.48	33.49	35.17
	SSIM ↑	0.974	0.977	0.982	0.983	0.986	0.960	0.960	0.979	0.975
	LPIPS ↓	0.113	0.116	0.111	0.100	0.118	0.156	0.132	0.142	0.124
GS-SLAM	PSNR ↑	31.56	32.86	32.59	38.70	41.17	32.36	32.03	32.92	34.27
	SSIM ↑	0.968	0.973	0.971	0.986	0.993	0.978	0.970	0.968	0.975
	LPIPS ↓	0.094	0.075	0.093	0.050	0.033	0.094	0.110	0.112	0.082
SplaTAM	PSNR ↑	32.86	33.89	35.25	38.26	39.17	31.97	29.70	31.81	34.11
	SSIM ↑	0.980	0.970	0.980	0.980	0.980	0.970	0.950	0.950	0.970
	LPIPS ↓	0.070	0.100	0.080	0.090	0.090	0.100	0.120	0.150	0.100
RK-SLAM (Ours)	PSNR ↑	32.90	34.55	39.69	39.49	39.17	33.39	34.43	34.98	36.08
	SSIM ↑	0.934	0.942	0.926	0.971	0.967	0.940	0.947	0.948	0.947
	LPIPS ↓	0.079	0.077	0.076	0.045	0.061	0.109	0.080	0.077	0.076

In the comparisons above, RK-SLAM demonstrates a strong performance on both the Replica dataset, with complete depth information, and the TUM-RGBD dataset, which

contains missing and noisy depth data. This highlights RK-SLAM's excellent map reconstruction capabilities and its robustness to disturbances in depth information.

4.3. Ablation Study

We conducted experimental evaluations on the TUM-RGBD dataset to assess the performance improvements brought by RK-SLAM's sub-Gaussian and related keyframe window. Table 6 presents the results of the ablation study, including both tracking and mapping performances. Notably, for the variant without the related keyframe window, we replaced it with the keyframe window proposed in [28] using the same parameter settings.

Table 6. Ablation study on TUM-RGBD dataset. The ✓ means with the corresponding method, and ✗ means without the corresponding method. ↑ means that the bigger the metric, the better, and ↓ means that the smaller the metric, the better. The table marks the first, second, and third best results.

Sub-Gaussian	Related Keyframe Window	Metrics	fr1/desk	fr2/xyz	fr3/Office	Avg.
✗	✓	PSNR ↑	21.50	21.92	20.06	21.16
		SSIM ↑	0.736	0.710	0.688	0.711
		LPIPS ↓	0.255	0.229	0.266	0.250
		ATE RMSE ↓	1.41	1.22	1.22	1.28
✓	✗	PSNR ↑	20.89	21.12	20.24	20.75
		SSIM ↑	0.726	0.705	0.700	0.710
		LPIPS ↓	0.266	0.245	0.268	0.260
		ATE RMSE ↓	1.44	1.36	1.38	1.39
✓	✓	PSNR ↑	22.31	22.47	20.67	21.82
		SSIM ↑	0.741	0.729	0.710	0.727
		LPIPS ↓	0.254	0.220	0.251	0.242
		ATE RMSE ↓	1.30	1.21	1.15	1.22

As shown in Table 6, the method without sub-Gaussians performs worse than RK-SLAM in both tracking and rendering. It indicates that the sub-Gaussian approach contributes to improvements in both aspects. Regarding the choice of keyframe windows, compared to the related keyframe window used in RK-SLAM, employing the sliding keyframe window proposed in [28] results in significant performance gaps in both tracking and rendering. This is especially evident in tracking performance, where the related keyframe window brings a notable improvement, achieving an average enhancement of 0.17 cm across the three sequences in the TUM-RGBD dataset. For the mapping results, as shown in Figure 4, we selected a frame from the fr3/Office sequence for comparison. It is evident that RK-SLAM demonstrates superior detail preservation, particularly when compared to the method without sub-Gaussians.

To explore the optimal size for the sliding window W_s , we conducted the following experiments. The purpose of the sliding window is to enable sub-Gaussians to adjust dynamically with the fine-tuned poses of the keyframes. Thus, we designed a sliding window with a relatively large size (set to 5) to track the pose changes of the keyframes within it. Theoretically, when a keyframe undergoes significant fine-tuning, it is necessary for the sub-Gaussians to adjust accordingly. Conversely, once the keyframe's pose stabilizes, the sub-Gaussians can be united into the refined Gaussians G_0 .



Figure 4. Comparison of mapping results in the ablation study. The first row shows the full images, while the second row provides zoomed-in views of the regions with the red box from the first row.

In Figure 5, we illustrate the relationship between the extent of the keyframe pose fine-tuning and the number of iterations. The shaded area represents the range of fine-tuning extents, while the solid line denotes the mean value at each iteration. From the result, we observe that the fine-tuning extent decreases overall with an increasing number of iterations. After three iterations, the adjustments become smaller and exhibit no significant changes. This indicates that the keyframe poses tend to stabilize after three iterations. Based on this result, we set the sliding window size to 3 in this study, allowing the sub-Gaussians to adapt dynamically during significant fine-tuning of the keyframe poses.

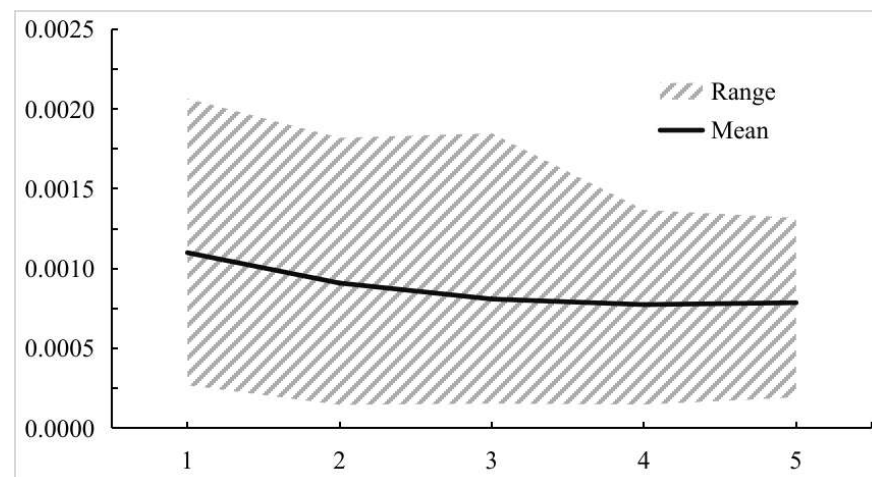


Figure 5. The extent of fine-tuning of the keyframe pose in the sliding window W_s with the number of iterations.

5. Discussion and Future Work

In our study, RK-SLAM demonstrated an excellent performance across both synthetic scenes with accurate and complete information and real-world scenes with perturbations and missing data. And they had different resolutions (1200×680 RGB-D images for Replica and 640×480 RGB-D images for TUM-RGBD and ScanNet). Our proposed SLAM system can run effectively across these datasets with varying resolutions and still achieve good results. This highlights the robustness of our proposed SLAM system to varying data. Particularly in the ablation experiments, we compared the related keyframe window with the sliding window approach proposed in [28]. The sliding window in [28] updates the window based on the sequence order, disregarding inter-frame correlations. In contrast,

the related keyframe window considers the content of the scene, breaking the original sequential order. Even if the orders of keyframes are far apart in the sequence, they are selected due to their relation and fine-tune the scene map and the poses of the keyframes together. The experimental results clearly demonstrate that this method can achieve better localization and mapping. Moreover, through the exploration of the sliding window, we discovered changes in the amount of keyframe fine-tuning during the iteration process. Leveraging this insight, we set the sliding window size to maximize the effectiveness of the sub-Gaussians.

However, RK-SLAM still has shortcomings.

1. In terms of mapping performance, RK-SLAM excels in PSNR and LPIPS metrics but shows a significant gap in the SSIM metric.
2. The datasets used in the experiments mostly consist of static scenes, whereas the real world is dynamic. Our SLAM system has no account for dynamic scenes and may, therefore, struggle to handle rapidly changing scenes.
3. Although we constrained the maximum sizes of the sliding window and the related window, the iterations still require a significant amount of computation, preventing real-time performance.

Therefore, in future work, we will address these issues. First, we need to investigate the reasons behind the significant discrepancy in the SSIM metric to improve our algorithm. Additionally, to enable an efficient algorithm deployment, we aim to design a new Gaussian mapping method that avoids the computational bottlenecks of the current approach. Furthermore, for dynamic scenes, we need to develop a new method to mitigate the impact of dynamic disturbances.

6. Conclusions

This paper presents a novel SLAM system based on 3D Gaussian splatting (3DGS), called RK-SLAM, which utilizes 3DGS as the core for tracking and mapping. The system incorporates two key technologies: the sub-Gaussian method and the related frame selection algorithm. The sub-Gaussian approach binds Gaussians to keyframes, allowing them to fine-tune in sync with the keyframe poses. This effectively mitigates the impact of tracking errors on mapping. The related frame selection algorithm calculates inter-frame relation based on co-visibility and spatial distance between frames, achieving a loop closure-like effect. This method overcomes the limitations of existing approaches, which struggle to relate frames that are far apart in terms of sequence order. As a result, RK-SLAM performs better in indoor SLAM sequences with cyclic structures, such as the Replica dataset and TUM-RGBD dataset. Our proposed SLAM system offers a potential solution for dense visual SLAM. Meanwhile, the sub-Gaussian method and related keyframe selection algorithm provide promising directions for future advancements in 3DGS-based SLAM systems.

Author Contributions: Conceptualization, X.M. and Y.J.; methodology, X.M. and C.S.; software, X.M.; validation, Y.J., C.S., and S.Z.; writing—original draft preparation, X.M.; writing—review and editing, Y.J., C.S., and S.Z.; supervision, S.Z.; funding acquisition, S.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the STI 2030—Major Projects 2021ZD0200400, National Natural Science Foundation of China (Grant No. 62203443), Major program of the National Natural Science Foundation of China (Grant No. T2293720/3724/3725), and the Excellent Youth Program of State Key Laboratory of Multimodal Artificial Intelligence Systems (E4SP100101).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Cadena, C.; Carlone, L.; Carrillo, H.; Latif, Y.; Scaramuzza, D.; Neira, J.; Reid, I.; Leonard, J.J. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Trans. Robot.* **2016**, *32*, 1309–1332. [\[CrossRef\]](#)
2. Kazerouni, I.A.; Fitzgerald, L.; Dooly, G.; Toal, D. A survey of state-of-the-art on visual SLAM. *Expert Syst. Appl.* **2022**, *205*, 117734. [\[CrossRef\]](#)
3. Engel, J.; Koltun, V.; Cremers, D. Direct sparse odometry. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *40*, 611–625. [\[CrossRef\]](#) [\[PubMed\]](#)
4. Qin, T.; Li, P.; Shen, S. Vins-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Trans. Robot.* **2018**, *34*, 1004–1020. [\[CrossRef\]](#)
5. Campos, C.; Elvira, R.; Rodríguez, J.J.G.; Montiel, J.M.; Tardós, J.D. Orb-slam3: An accurate open-source library for visual, visual-inertial, and multimap slam. *IEEE Trans. Robot.* **2021**, *37*, 1874–1890. [\[CrossRef\]](#)
6. Liu, S.; Zhang, Y.; Tang, K.; Yao, X. How Good is Neural Combinatorial Optimization? A Systematic Evaluation on the Traveling Salesman Problem. *IEEE Comput. Intell. Mag.* **2023**, *18*, 14–28. [\[CrossRef\]](#)
7. Ruetz, F.; Hernández, E.; Pfeiffer, M.; Oleynikova, H.; Cox, M.; Lowe, T.; Borges, P. Ovpc mesh: 3D free-space representation for local ground vehicle navigation. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; pp. 8648–8654.
8. Schöps, T.; Sattler, T.; Pollefeys, M. Surfelmeshing: Online surfel-based mesh reconstruction. *IEEE Trans. Pattern Anal. Mach. Intell.* **2019**, *42*, 2494–2507. [\[CrossRef\]](#)
9. Nießner, M.; Zollhöfer, M.; Izadi, S.; Stamminger, M. Real-time 3D reconstruction at scale using voxel hashing. *ACM Trans. Graph. (ToG)* **2013**, *32*, 1–11. [\[CrossRef\]](#)
10. Kähler, O.; Prisacariu, V.; Valentin, J.; Murray, D. Hierarchical voxel block hashing for efficient integration of depth images. *IEEE Robot. Autom. Lett.* **2015**, *1*, 192–197. [\[CrossRef\]](#)
11. Dai, A.; Nießner, M.; Zollhöfer, M.; Izadi, S.; Theobalt, C. Bundlefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration. *ACM Trans. Graph. (ToG)* **2017**, *36*, 1. [\[CrossRef\]](#)
12. Newcombe, R.A.; Izadi, S.; Hilliges, O.; Molyneaux, D.; Kim, D.; Davison, A.J.; Kohi, P.; Shotton, J.; Hodges, S.; Fitzgibbon, A. Kinectfusion: Real-time dense surface mapping and tracking. In Proceedings of the 2011 10th IEEE International Symposium on Mixed and Augmented Reality, Basel, Switzerland, 26–29 October 2011; pp. 127–136.
13. Whelan, T.; Johannsson, H.; Kaess, M.; Leonard, J.J.; McDonald, J. Robust real-time visual odometry for dense RGB-D mapping. In Proceedings of the 2013 IEEE International Conference on Robotics and Automation, Karlsruhe, Germany, 6–10 May 2013; pp. 5724–5731.
14. Weder, S.; Schonberger, J.L.; Pollefeys, M.; Oswald, M.R. Neurfusion: Online depth fusion in latent space. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 3162–3172.
15. Mildenhall, B.; Srinivasan, P.P.; Tancik, M.; Barron, J.T.; Ramamoorthi, R.; Ng, R. Nerf: Representing scenes as neural radiance fields for view synthesis. *Commun. ACM* **2021**, *65*, 99–106. [\[CrossRef\]](#)
16. Wang, Z.; Wu, S.; Xie, W.; Chen, M.; Prisacariu, V.A. NeRF-: Neural radiance fields without known camera parameters. *arXiv* **2021**, arXiv:2102.07064.
17. Pumarola, A.; Corona, E.; Pons-Moll, G.; Moreno-Noguer, F. D-nerf: Neural radiance fields for dynamic scenes. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Nashville, TN, USA, 20–25 June 2021; pp. 10318–10327.
18. Kerbl, B.; Kopanas, G.; Leimkühler, T.; Drettakis, G. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. *ACM Trans. Graph.* **2023**, *42*, 139. [\[CrossRef\]](#)
19. Wu, G.; Yi, T.; Fang, J.; Xie, L.; Zhang, X.; Wei, W.; Liu, W.; Tian, Q.; Wang, X. 4D gaussian splatting for real-time dynamic scene rendering. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 16–22 June 2024; pp. 20310–20320.
20. Yang, Z.; Gao, X.; Zhou, W.; Jiao, S.; Zhang, Y.; Jin, X. Deformable 3D Gaussians for high-fidelity monocular dynamic scene reconstruction. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 16–22 June 2024; pp. 20331–20341.

21. Rosinol, A.; Leonard, J.J.; Carlone, L. Nerf-slam: Real-time dense monocular slam with neural radiance fields. In Proceedings of the 2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Detroit, MI, USA, 1–5 October 2023; pp. 3437–3444.
22. Chung, C.M.; Tseng, Y.C.; Hsu, Y.C.; Shi, X.Q.; Hua, Y.H.; Yeh, J.F.; Chen, W.C.; Chen, Y.T.; Hsu, W.H. Orbee-slam: A real-time monocular visual slam with orb features and nerf-realized mapping. In Proceedings of the 2023 IEEE International Conference on Robotics and Automation (ICRA), London, UK, 29 May–2 June 2023; pp. 9400–9406.
23. Lisus, D.; Holmes, C.; Waslander, S. Towards open world nerf-based slam. In Proceedings of the 2023 20th Conference on Robots and Vision (CRV), Montreal, QC, Canada, 6–8 June 2023; pp. 37–44.
24. Tosi, F.; Zhang, Y.; Gong, Z.; Sandström, E.; Mattoccia, S.; Oswald, M.R.; Poggi, M. How nerfs and 3D gaussian splatting are reshaping slam: A survey. *arXiv* **2024**, arXiv:2402.13255.
25. Guo, C.; Gao, C.; Bai, Y.; Lv, X. RD-SLAM: Real-Time Dense SLAM Using Gaussian Splatting. *Appl. Sci.* **2024**, *14*, 7767. [[CrossRef](#)]
26. Yan, C.; Qu, D.; Xu, D.; Zhao, B.; Wang, Z.; Wang, D.; Li, X. Gs-slam: Dense visual slam with 3d gaussian splatting. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 16–22 June 2024; pp. 19595–19604.
27. Keetha, N.; Karhade, J.; Jatavallabhula, K.M.; Yang, G.; Scherer, S.; Ramanan, D.; Luiten, J. SplatAM: Splat Track & Map 3D Gaussians for Dense RGB-D SLAM. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 16–22 June 2024; pp. 21357–21366.
28. Matsuki, H.; Murai, R.; Kelly, P.H.; Davison, A.J. Gaussian splatting slam. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 16–22 June 2024; pp. 18039–18048.
29. Huang, Z.; Wen, Y.; Wang, Z.; Ren, J.; Jia, K. Surface reconstruction from point clouds: A survey and a benchmark. *IEEE Trans. Pattern Anal. Mach. Intell.* **2024**, *46*, 9727–9748. [[CrossRef](#)] [[PubMed](#)]
30. Kato, H.; Ushiku, Y.; Harada, T. Neural 3D mesh renderer. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 3907–3916.
31. Pontes, J.K.; Kong, C.; Sridharan, S.; Lucey, S.; Eriksson, A.; Fookes, C. Image2mesh: A learning framework for single image 3D reconstruction. In Proceedings of the Computer Vision–ACCV 2018: 14th Asian Conference on Computer Vision, Perth, Australia, 2–6 December 2018; Revised Selected Papers, Part I 14; Springer: Berlin/Heidelberg, Germany, 2019; pp. 365–381.
32. Cheung, G.K.; Kanade, T.; Bouguet, J.Y.; Holler, M. A real time system for robust 3D voxel reconstruction of human motions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No. PR00662), Hilton Head Island, SC, USA, 15 June 2000; Volume 2, pp. 714–720.
33. Liu, H.; Cong, Y.; Wang, S.; Fan, H.; Tian, D.; Tang, Y. Deep learning of directional truncated signed distance function for robust 3D object recognition. In Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, BC, Canada, 24–28 September 2017; pp. 5934–5940.
34. Mur-Artal, R.; Tardós, J.D. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE Trans. Robot.* **2017**, *33*, 1255–1262. [[CrossRef](#)]
35. Sucar, E.; Liu, S.; Ortiz, J.; Davison, A.J. imap: Implicit mapping and positioning in real-time. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 11–17 October 2021; pp. 6229–6238.
36. Zhu, Z.; Peng, S.; Larsson, V.; Xu, W.; Bao, H.; Cui, Z.; Oswald, M.R.; Pollefeys, M. NICE-SLAM: Neural Implicit Scalable Encoding for SLAM. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), New Orleans, LA, USA, 18–24 June 2022; pp. 12786–12796.
37. Wang, H.; Wang, J.; Agapito, L. Co-SLAM: Joint Coordinate and Sparse Parametric Encodings for Neural Real-Time SLAM. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, 18–22 June 2023; pp. 13293–13302.
38. Sandström, E.; Li, Y.; Van Gool, L.; Oswald, M.R. Point-slam: Dense neural point cloud-based slam. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Paris, France, 1–6 October 2023; pp. 18433–18444.
39. Hu, J.; Chen, X.; Feng, B.; Li, G.; Yang, L.; Bao, H.; Zhang, G.; Cui, Z. Cg-slam: Efficient dense rgb-d slam in a consistent uncertainty-aware 3D gaussian field. In Proceedings of the European Conference on Computer Vision, Paris, France, 26–27 March 2025; Springer: Berlin/Heidelberg, Germany, 2025; pp. 93–112.
40. Wang, Z.; Bovik, A.C.; Sheikh, H.R.; Simoncelli, E.P. Image quality assessment: From error visibility to structural similarity. *IEEE Trans. Image Process.* **2004**, *13*, 600–612. [[CrossRef](#)] [[PubMed](#)]
41. Straub, J.; Whelan, T.; Ma, L.; Chen, Y.; Wijmans, E.; Green, S.; Engel, J.J.; Mur-Artal, R.; Ren, C.; Verma, S.; et al. The Replica dataset: A digital replica of indoor spaces. *arXiv* **2019**, arXiv:1906.05797.
42. Sturm, J.; Engelhard, N.; Endres, F.; Burgard, W.; Cremers, D. A benchmark for the evaluation of RGB-D SLAM systems. In Proceedings of the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, Vilamoura-Algarve, Portugal, 7–12 October 2012; pp. 573–580.

43. Dai, A.; Chang, A.X.; Savva, M.; Halber, M.; Funkhouser, T.; Nießner, M. ScanNet: Richly-annotated 3D Reconstructions of Indoor Scenes. In Proceedings of the Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 21–26 July 2017.
44. Zhang, R.; Isola, P.; Efros, A.A.; Shechtman, E.; Wang, O. The unreasonable effectiveness of deep features as a perceptual metric. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 18–23 June 2018; pp. 586–595.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.