Neurocomputing 711 (2027) 135368

Contents lists available at ScienceDirect

Neurocomputing

journal homepage: www.elsevier.com/locate/neucom

Normal-aware simultaneous localization and mapping with 3D Gaussian splatting

Zhengwu Shi [a], Yuan Rao [a], Xiaowei Zhou [a], Baogui Huan [a], Junyu Dong [a,*], Hui Yu [a,b], Sheng Chen [a,c]

[a] Faculty of Information Science and Engineering, Ocean University of China, Qingdao 266100, China
[b] The cSCAN Centre, University of Glasgow, Glasgow, UK
[c] School of Electronics and Computer Science, University of Southampton, Southampton SO17 1BJ, UK

ARTICLE INFO

Communicated by R. CONG

Keywords:
Simultaneous localization and mapping
3D Gaussian splatting
Image synthesis
3D reconstruction

ABSTRACT

SLAM is a fundamental capability for autonomous systems operating in unknown environments. Recent 3D Gaussian Splatting (3DGS)-based SLAM methods have demonstrated promising reconstruction quality and rendering efficiency. However, existing approaches often suffer from insufficient geometric constraints, leading to redundant Gaussian representations and unstable localization, especially under monocular RGB input. To reduce geometric ambiguity during online 3DGS-based SLAM, we propose NA-SLAM, a unified normal-aware framework for both RGB and RGB-D input modalities. NA-SLAM actively uses covariance-derived Gaussian normals as geometric cues through self-supervised normal regularization, rotation-guided keyframe management for improved multi-view coverage, and normal-guided filtering of geometrically unreliable primitives. These coupled components coordinate geometric regularization, observation management, and Gaussian morphology without requiring ground-truth normals or an external normal-estimation network. Experiments on the TUM RGB-D and Replica datasets demonstrate that the proposed method improves localization accuracy and reconstruction quality compared with existing 3DGS-based SLAM approaches. Under RGB-only input, NA-SLAM reduces the average trajectory error by 42% compared with MonoGS while using significantly fewer Gaussian primitives for scene representation. The proposed framework supports both RGB and RGB-D inputs and achieves competitive performance in dense SLAM and novel-view rendering tasks.

1. Introduction

Simultaneous Localization and Mapping (SLAM) plays an important role in robotics, augmented reality, and autonomous navigation systems [1–4]. Monocular SLAM is particularly attractive because it only requires low-cost RGB cameras and can be deployed on lightweight platforms. However, achieving accurate localization and dense scene reconstruction from monocular input remains challenging due to the absence of direct geometric measurements.

Traditional feature-based SLAM methods achieve reliable camera tracking through sparse feature matching and geometric optimization. However, these methods usually produce sparse or semi-dense maps and often struggle in texture-poor environments. Recent learning-based approaches and neural scene representations have improved dense reconstruction quality. In particular, Neural Radiance Fields (NeRFs) [5] provide high-quality rendering results but require expensive optimization and are difficult to deploy in real-time SLAM systems.

Recently, 3D Gaussian Splatting (3DGS) [6,7] has emerged as an efficient scene representation for real-time rendering and reconstruction. Compared with NeRF-based methods, 3DGS provides faster rendering and lower optimization overhead, making it suitable for online SLAM systems. Several recent studies [8–11] have incorporated 3DGS into SLAM frameworks and demonstrated promising reconstruction quality. Among them, MonoGS enables monocular SLAM without explicit depth priors by initializing Gaussian primitives from multi-view observations.

Despite these advantages, existing 3DGS-based SLAM methods still suffer from geometric ambiguity during optimization. Since Gaussian primitives are mainly constrained by photometric consistency, redundant or incorrectly initialized Gaussians may remain in the scene representation, especially under RGB-only input. This issue can lead to

* Corresponding author.
Email addresses: shizhengwu@stu.ouc.edu.cn (Z. Shi), raoyuan@ouc.edu.cn (Y. Rao), zhouxiaowei@ouc.edu.cn (X. Zhou), xunbaogui@stu.ouc.edu.cn (B. Huan), dongjunyu@ouc.edu.cn (J. Dong), hui.yu@glasgow.ac.uk (H. Yu), sqc@ecs.soton.ac.uk (S. Chen).

https://doi.org/10.1016/j.neucom.2026.135368
Received 27 May 2026; Received in revised form 27 July 2026; Accepted 9 October 2026
Available online 10 October 2026
0925-2312/© 2026 Published by Elsevier B.V.

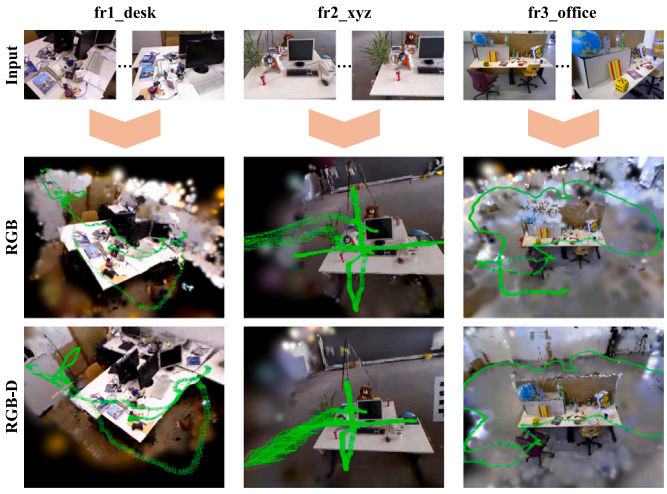


Fig. 1. Under RGB input, NA-SLAM achieves geometrically consistent reconstruction without requiring explicit depth estimation while also supporting RGB-D input.

inaccurate surface structures, unstable tracking, and unnecessary memory consumption. In addition, insufficient viewpoint observations during online mapping further increase optimization uncertainty.

To address the geometric ambiguity of Gaussian primitives during online SLAM, we propose a unified normal-aware framework based on 3D Gaussian Splatting, termed NA-SLAM. The central methodological idea is to use the normal direction induced by the covariance ellipsoid as an actively optimized geometric cue rather than merely as a derived geometric attribute. Based on this idea, a self-supervised normal constraint regularizes Gaussian geometry using RGB and, when available, depth discontinuities, without requiring ground-truth normals or an external normal-estimation network. Rotation-guided keyframe management improves the multi-view observation coverage required for reliable normal-constrained mapping, while normal-guided filtering suppresses Gaussian primitives with excessive shortest-axis scales. These three components address geometric regularization, observation sufficiency, and primitive morphology, respectively, forming a coupled normal-aware mapping process. Some representative results of the proposed framework are shown in Fig. 1.

We evaluate the proposed method on the TUM RGB-D and Replica datasets using both RGB and RGB-D inputs. Experimental results show that NA-SLAM improves localization accuracy and reconstruction quality. It also reduces the number of Gaussian primitives on the evaluated TUM sequences, whereas its effect on the final map size is dataset-dependent.

2. Related work

2.1. Visual SLAM

Traditional visual SLAM systems simultaneously estimate egomotion and reconstruct environmental geometry through sequential camera observations, typically employing feature-based tracking or direct photometric alignment for incremental 3D mapping. In the field of visual SLAM, several classical algorithms have been widely studied and applied. Parallel tracking and mapping (PTAM) [12] was pioneering in separating tracking and mapping into two distinct threads, utilizing nonlinear optimization for pose estimation, which at the time provided groundbreaking advantages in both speed and accuracy. However, its reliance on patch matching makes it susceptible to reduced accuracy due to variations in lighting intensity. Additionally, the absence of a loop closure process prevents the elimination of accumulated errors. ORB-SLAM [13–15], which uses robust ORB features and a comprehensive loop closure mechanism, improves long-term stability. But it fails to produce dense scene reconstructions and is ineffective in low-texture environments. Large-scale direct monocular SLAM (LSD-SLAM) [16] generates semi-dense maps and is suitable for low-texture scenes but faces memory constraints in large-scale environments. Semi-direct visual odometry (SVO) [17] combines the high efficiency of direct methods with the high robustness of feature-based approaches, making it suitable for high-speed platforms, but it lacks long-term stability and an effective loop closure strategy.

Despite their strengths, these traditional visual SLAM systems often fall short in 3D reconstruction, particularly in providing high-precision, detailed, and structurally complete 3D models, highlighting the need for further advancements.

2.2. Learning-based SLAM

Traditional SLAM methods primarily rely on geometric feature matching and optimization techniques to achieve environment mapping and robot localization. However, with the advancement of deep learning technologies, researchers have begun to explore how deep neural networks can be utilized to enhance the performance of SLAM systems. DeepVO [18], one of the pioneering efforts, utilizes recurrent neural networks to achieve end-to-end camera trajectory estimation. Despite its innovative approach, DeepVO exhibits issues with cumulative error over long sequences and sensitivity to initial pose estimates. Deep virtual stereo odometry [19] leverages deep depth prediction to enhance direct sparse odometry (DSO) [20]. It uses a deep learning model to predict depth maps from monocular images, effectively transforming the monocular setup into a virtual stereo system. However, this method is computationally intensive, limiting its real-time performance.

More recently, DROID-SLAM [21] recurrently updates camera poses and pixelwise depth through a differentiable dense bundle-adjustment layer. By directly learning motion fields from optical flow using RAFT [22], DROID-SLAM simplifies the process, eliminating the need for complex feature matching and improving both efficiency and robustness. DeepVIO [23], another notable advancement, combines visual and inertial sensor data through deep learning models to exploit their complementary characteristics, achieving more accurate pose estimation, particularly in high-speed and dynamic scenarios. D3VO [24] proposes a deep learning-driven direct visual odometry method that not only retains the advantages of traditional direct methods but also improves illumination invariance and scale estimation through deep learning, ensuring good performance across various environments. Recently, several end-to-end deep learning frameworks for 3D reconstruction, such as DUSt3R [25] and VGGT [26], have been proposed, followed by the emergence of SLAM systems built upon these architectures [27,28]. However, the aforementioned approaches rely on extensive datasets and prolonged training periods to ensure their accuracy.

In short, current deep learning-based SLAM methods still face challenges, such as computational efficiency, data requirements, and cross-domain generalization capabilities.

2.3. 3DGS-based SLAM

Recent developments in neural rendering techniques, such as neural radiance fields (NeRF) and 3DGS, have significantly advanced the field of SLAM. Notably, 3DGS has emerged as a promising approach for real-time applications in 3D reconstruction and mapping.

Previous studies [29–36] have demonstrated the effectiveness of incorporating 3DGS into SLAM systems, showing significant improvements over learning-based SLAM in terms of real-time performance, memory efficiency, and reconstruction quality. However, integration of 3DGS into SLAM has introduced new challenges. First, the position optimization of 3DGS relies heavily on depth information or sufficient viewpoint observations, which cannot always be guaranteed within the SLAM framework. Second, balancing the computational cost of 3DGS

Table 1
Comparison of representative visual SLAM and reconstruction methods.

Method	Input modality	External pre-training	Dense reconstruction
ORB-SLAM2 [14]	RGB/RGB-D	No	No
DROID-SLAM [21]	RGB/RGB-D	Yes	Yes
VGGT [26]	RGB	Yes	Yes
iMAP [37]	RGB-D	No	Yes
NICE-SLAM [38]	RGB-D	Yes	Yes
Vox-Fusion [39]	RGB-D	No	Yes
ESLAM [40]	RGB-D	No	Yes
Co-SLAM [41]	RGB-D	No	Yes
Point-SLAM [42]	RGB-D	Yes	Yes
SplaTAM [32]	RGB-D	No	Yes
GS-SLAM [34]	RGB-D	No	Yes
MonoGS [10]	RGB/RGB-D	No	Yes
Our NA-SLAM	RGB/RGB-D	No	Yes

optimization with the real-time efficiency of SLAM remains a critical issue for maintaining real-time performance.

Among existing 3DGS-based SLAM systems, SplaTAM [32] and GS-SLAM [34] mainly focus on RGB-D input. SplaTAM employs a rendered silhouette mask together with color and depth observations for camera tracking and structured Gaussian map expansion. GS-SLAM introduces adaptive Gaussian expansion and deletion, as well as a coarse-to-fine strategy for efficient RGB-D camera tracking. These methods benefit from measured depth as a direct geometric constraint. In contrast, NA-SLAM supports both RGB and RGB-D inputs and is intended in particular, to reduce geometric ambiguity when measured depth is unavailable. Under RGB-only input, NA-SLAM follows MonoGS in randomly initializing the depths of newly generated Gaussian primitives and subsequently constraining their geometry through multi-view optimization.

MonoGS [10] is the most closely related method because it also uses 3D Gaussians as the unified representation for direct camera tracking, incremental mapping, and novel-view rendering under both monocular and RGB-D inputs. However, MonoGS does not explicitly coordinate normal consistency, rotational viewpoint change, and Gaussian morphology during online map optimization. Building upon this framework, NA-SLAM introduces three complementary mechanisms: self-supervised normal regularization to constrain Gaussian geometry, rotation-guided keyframe management to improve multi-view observation coverage, and normal-guided filtering to suppress geometrically unreliable Gaussian primitives. The proposed normal constraint requires neither ground-truth normals nor an additional pretrained normal-estimation network.

Recent methods address structural consistency and map redundancy from different perspectives. MG-SLAM [8] incorporates line features and the Manhattan-world hypothesis to improve the completeness of structured indoor scenes, but its geometric prior assumes dominant orthogonal directions and requires RGB-D input. NA-SLAM instead imposes local normal consistency without assuming a global Manhattan structure. FIGS-SLAM [36] progressively introduces image-frequency components during mapping and removes Gaussians according to their rendering influence, including opacity, transmittance, and co-visibility. In contrast, our filtering criterion is derived from the normal-axis morphology of each Gaussian, while the rotation-guided keyframe strategy improves the observations used to optimize these normals. Therefore, NA-SLAM differs from existing approaches in jointly integrating normal-domain regularization, viewpoint-aware keyframe management, and shape-aware Gaussian filtering within a unified RGB/RGB-D SLAM framework, as shown in Table 1.

3. Approach

As illustrated in Fig. 2, we constrain and optimize the pose estimation by leveraging the discrepancy between the input images and those rendered from the 3D Gaussian map. For keyframe management, we employ an adaptive strategy based on pose variation metrics and multiple references derived from 3D Gaussian observations to manage keyframes effectively. Subsequently, we augment and optimize the 3D Gaussian map using the perspectives provided by these keyframes. In this process, we apply self-supervised smooth normal constraints to ensure that the 3D Gaussian ellipsoids are optimized to more geometrically consistent positions and shapes. Additionally, we filter Gaussian primitives of the 3D Gaussian ellipsoids, removing those that do not adequately conform to the surface, to ensure that the 3D Gaussian map accurately represents the real-world scene. The three components of NA-SLAM are organized around the reliability of normal-constrained Gaussian optimization. The normal constraint defines the geometric regularization applied during mapping, keyframe management maintains the multi-view observations required to optimize this constraint, and normal-guided filtering controls the morphology of the Gaussian primitives participating in subsequent optimization. This approach not only enhances the fidelity of the map representation but also indirectly improves localization accuracy.

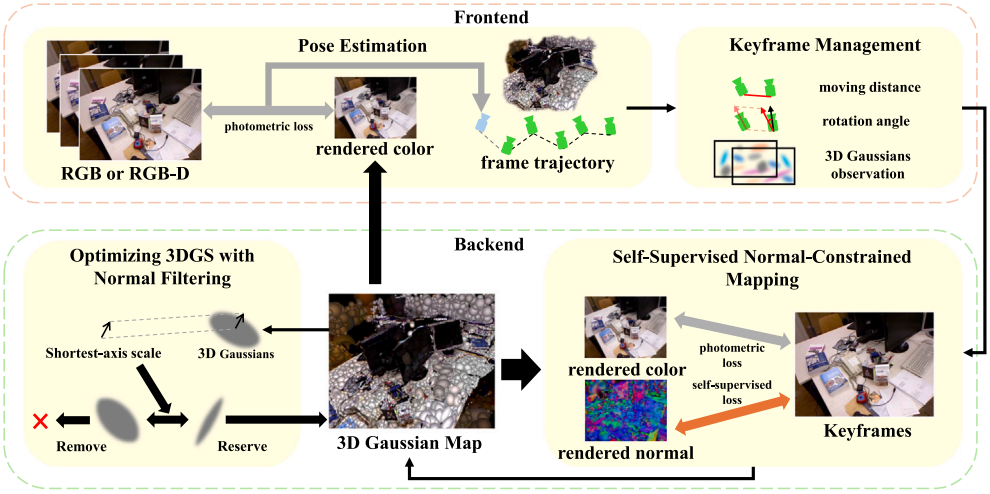


Fig. 2. Overview of NA-SLAM: In the front end, we optimize the pose based on the loss between rendered and input images. Concurrently, we manage the keyframe window based on translational distance, rotational angle, and Gaussian observation overlap. During mapping, we fix the keyframe poses and optimize the 3D Gaussian map using photometric loss and the self-supervised loss. Simultaneously, we optimize the 3D Gaussian ellipsoids in the scene through normal filtering.

3.1. 3D Gaussian map

The scene map is composed of a set of N anisotropic Gaussian functions $\{G_i\}_{i=1}^{N}$. Each Gaussian function G_i includes a color value c_i, opacity o_i, spatial position p_i, and a covariance matrix Σ_i. By splatting and blending the N Gaussian functions as observed from the current viewpoint, the rendered image pixels C_p can be synthesized:

$$C_p = \sum_{i \in N} c_i o_i \prod_{j=1}^{i-1} (1 - o_j). \tag{1}$$

For the surface normals of the rendered image pixels, the covariance matrix Σ_i in the Gaussian function can be represented in the form of an ellipsoid as follows:

$$\Sigma_i = R\, S\, S^{\mathrm{T}} R^{\mathrm{T}}, \tag{2}$$

where $R \in SO(3)$ and $S = diag(s_1, s_2, s_3)$ are the rotation and scaling matrices of the Gaussian ellipsoid, respectively.

We approximate the direction of the smallest semi-axis in the scaling matrix S as the normal direction $\mathbf{n}$ of the 3DGS, which can be derived as follows:

$$\mathbf{n} = R[r, :], \; r = \arg\min([s_1, s_2, s_3]). \tag{3}$$

The corresponding normal rendering is expressed as:

$$N_p = \sum_{i \in N} n_i o_i \prod_{j=1}^{i-1} (1 - o_j), \tag{4}$$

where n_i plays a role similar to that of c_i, representing the accumulation of normal samples on the pixel resulting from 3D Gaussian splatting.

Similarly, the depth of rendered image pixels can be obtained as follows:

$$D_p = \sum_{i \in N} d_i o_i \prod_{j=1}^{i-1} (1 - o_j), \tag{5}$$

where d_i represents the depth of the 3DGS center projected onto the image plane via splatting in the current view.

3.2. Pose estimation

First, we derive the gradient propagation between the rendered image and the 3D Gaussians. To make the constraints and optimization process provided by 2D images differentiable with respect to the 3D Gaussians, the transformation between a 3D Gaussian $G_W(p_W, \Sigma_W)$ and its projection as a 2D Gaussian $G_I(p_I, \Sigma_I)$ on the image plane is defined as follows:

$$p_I = \pi(T_{cw} \cdot p_W), \; \Sigma_I = J\, W\, \Sigma_W W^{\mathrm{T}} J^{\mathrm{T}}, \tag{6}$$

where T_{cw} represents the current camera pose in the world coordinate system, π represents the projection operation from the world frame to the camera, J is the linearized Jacobian matrix of the projection, and W is the rotation component of the pose T_{cw}.

According to the chain rule, we can precompute and simplify the gradient propagation process. Specifically, we obtain:

$$\frac{\partial p_W}{\partial T_{cw}} = \begin{bmatrix} I & -p_W^{\times} \end{bmatrix}, \; \frac{\partial \mathrm{W}}{\partial T_{cw}} = \begin{bmatrix} 0 & -\mathrm{W}_{:,1}^{\times} \\ 0 & -\mathrm{W}_{:,2}^{\times} \\ 0 & -\mathrm{W}_{:,3}^{\times} \end{bmatrix}, \tag{7}$$

where $\times$ denotes the skew-symmetric matrix of a 3D vector, and $W_{:,i}$ refers to the i-th column of the matrix.

Therefore, by enforcing pixel-wise loss constraints and leveraging the above formulations, the error is backpropagated to the pose matrix, enabling the optimizer to iteratively refine it toward the correct pose.

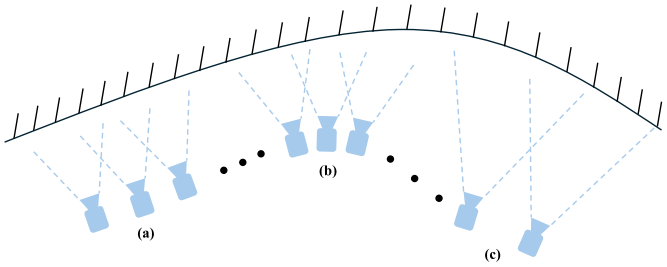


Fig. 3. Illustration of keyframe strategy.

For the tracking process, we render the viewpoint image using the existing 3D Gaussian map and optimize the camera pose using photometric error as the loss function. The loss is defined as follows:

$$L_I = \left\| I_r - I_{gt} \right\|_1, \tag{8}$$

where I_r is the rendered image, and I_{gt} is the ground truth image observed from the current viewpoint.

When the depth input D_{gt} is available, the tracking loss is augmented with a depth-based term computed from the rendered depth D_r as follows:

$$L_D = \left\| D_r - D_{gt} \right\|_1. \tag{9}$$

3.3. Rotation-guided keyframe management

To ensure adequate observational coverage during mapping and to effectively constrain the position, opacity, color, and surface normals of each 3D Gaussian, our keyframe selection strategy simultaneously takes into account the variations in the observed Gaussians and the motion of the camera. As shown in Fig. 3, we simultaneously evaluate the changes in translation and rotation and adaptively adjust the keyframe selection threshold. To provide effective constraints for normals, we increase the number of keyframes when surface observations are sparse.

For the addition of keyframes, we consider two dimensions as references: intersection over union (IoU) and camera motion. The IoU as one of the criteria for keyframe selection is defined as follows:

$$IoU(i, k) = \frac{|G_i^v \cap G_k^v|}{|G_i^v \cup G_k^v|}, \tag{10}$$

where G^v denotes the set of observed 3DGS ellipsoids, i denotes the current frame, and k is the previous keyframe. When the IoU is below the threshold, the frame is considered a keyframe candidate.

Additionally, we consider the variation V in different dimensions of motion. It is defined as follows:

$$V = \frac{M_d \cdot f_s(W_i)}{t_{ik}}, \tag{11}$$

where M_d is the median depth of the current frame, t_{ik} is the distance from the previous keyframe, W_i is the rotation matrix relative to the previous frame, and $f_s(W)$ is the scale factor function, defined as follows:

$$f_s(W) = \left(1 - \frac{1}{\pi} \arccos\left(\frac{\mathrm{trace}(W) - 1}{2}\right)\right)^2. \tag{12}$$

We further incorporate instantaneous rotation and short-term displacement metrics within the keyframe window, normalizing them against the current depth to ensure scale consistency. When V is below the threshold, the frame is determined to be a keyframe. Additionally, if V is further below the smaller threshold, the frame is forcibly considered a keyframe, ignoring the IoU.

For the removal of redundant frames, when the number of keyframes exceeds the sliding window limit, we calculate the overlap coefficient (OC) between the keyframes in the current window and the most recently inserted keyframe. The OC is defined as follows:

$$OC = \frac{|G_i \cap G_j|}{\min(|G_i|, |G_j|)}, \tag{13}$$

where G_i and G_j represent the sets of observed 3DGS ellipsoids for keyframes i and j, respectively.

To balance computational efficiency and real-time requirements, we limit the number of keyframes within the keyframe window to a predefined constant N. The keyframe is discarded if its OC drops below a specified threshold or if the total number of keyframes exceeds the predefined maximum.

3.4. Self-supervised normal-constrained mapping

During the mapping process, new 3DGS ellipsoids are generated and optimized based on the added keyframes. When depth observations are available, we initialize the 3DGS ellipsoids in the vicinity of the observed depth. Otherwise, we use a larger variance for random initialization of the 3DGS. For each keyframe within the keyframe window, we use the input color and depth (if available) as constraints, consistent with the tracking process. We optimize by calculating the loss using the Manhattan distance.

However, newly inserted Gaussian primitives are initially observed from only a limited number of viewpoints and may therefore remain geometrically underconstrained. This problem is more pronounced with RGB-only input, where surface depth is not directly observed and ground-truth surface normals are unavailable.

Inspired by edge-aware regularization in self-supervised depth estimation [43,44], we model the rendered normal field as locally piecewise smooth. RGB and depth gradients are used as spatial confidence weights rather than being assumed to correspond exactly to geometric boundaries. For RGB input, the edge-aware normal smoothness loss is defined as

$$L_{IS} = |\partial_x \mathbf{n}_r^*| e^{-|\partial_x I_{gt}|} + |\partial_y \mathbf{n}_r^*| e^{-|\partial_y I_{gt}|}, \tag{14}$$

where $\mathbf{n}_r^* = \mathbf{n}_r / \|\mathbf{n}_r\|$ is the unit-normalized rendered normal, and $\partial_x(\cdot)$ and $\partial_y(\cdot)$ denote first-order differences along the horizontal and vertical image directions, respectively.

Eq. (14) can be interpreted as an edge-aware weighted total-variation regularizer on the rendered normal field. In a locally low-gradient image region, the exponential weight approaches one and neighboring normals are encouraged to remain consistent. Near a strong image gradient, the weight decreases and the normal smoothness penalty is relaxed. Therefore, the loss does not enforce a normal discontinuity at every RGB edge. An edge caused by texture or illumination only weakens the local regularization, while the Gaussian geometry remains jointly determined by the photometric loss and multi-view observations.

When depth input is available, we additionally define a depth-guided normal smoothness loss as

$$L_{DS} = |\partial_x \mathbf{n}_r^*| e^{-|\partial_x D_{gt}|} + |\partial_y \mathbf{n}_r^*| e^{-|\partial_y D_{gt}|}, \tag{15}$$

where D_{gt} denotes the input depth map. Compared with RGB gradients, depth discontinuities provide more direct evidence of geometric boundaries. We use them as edge-aware weights for the rendered normal field rather than explicitly computing a target normal map from the input depth.

The mapping objective is defined according to the available input modality:

$$\mathcal{L}_{\text{mapping}} = \begin{cases} \mathcal{L}_I + \lambda_S \mathcal{L}_{IS}, & \text{RGB}, \\ \lambda_I \mathcal{L}_I + (1-\lambda_I)\mathcal{L}_D + \lambda_S \mathcal{L}_{IS} + (1-\lambda_S)\mathcal{L}_{DS}, & \text{RGB-D}, \end{cases} \tag{16}$$

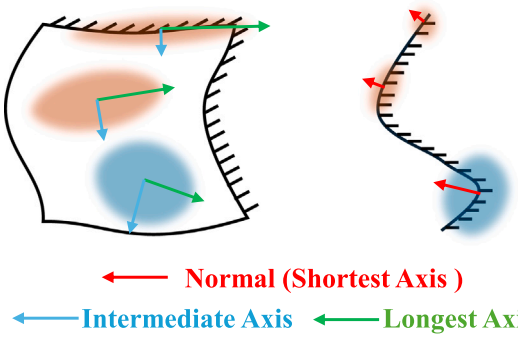


Fig. 4. Relationship between Gaussian morphology and normal-axis filtering. The shortest principal axis defines the estimated normal direction, and its scale represents the Gaussian thickness along that direction. (a) a spindle-like Gaussian has one elongated axis and two relatively short axes. (b) a disc-like Gaussian has one short normal axis and two larger tangential axes. (c) a volumetrically thick Gaussian has an excessive shortest-axis scale and is removed by Eq. (18).

where λ_I balances the photometric and depth terms, while λ_S controls the contributions of the RGB-guided and depth-guided normal smoothness terms. We set $\lambda_I = 0.95$ for RGB-D input and $\lambda_S = 0.1$ for both input modalities.

3.5. Optimizing 3DGS with normal filtering

As defined in Eqs. (2) and (3), the normal direction of a Gaussian is approximated by the principal axis corresponding to its smallest scale. The scale of this axis therefore represents the spatial thickness of the Gaussian along its estimated normal direction.

For a Gaussian aligned with a local surface, its spatial support perpendicular to the surface should remain limited, whereas its tangential scales may be larger to represent a planar patch or an elongated structure. As illustrated in Fig. 4, a disc-like Gaussian has one short normal axis and two larger tangential axes, while a spindle-like Gaussian has one elongated axis and two relatively short axes. Both morphologies contain at least one sufficiently thin principal direction. In contrast, when the shortest-axis scale is large, all three principal-axis scales are large and the Gaussian occupies a volumetric region rather than providing a locally surface-aligned representation.

Based on this geometric interpretation, we augment the original Gaussian pruning procedure by constraining the shortest-axis scale. Let the shortest-axis scale of the i-th Gaussian be

$$s_i^{\min} = \min\left(s_{i,1}, s_{i,2}, s_{i,3}\right). \tag{17}$$

The normal-guided filtering criterion is then defined as

$$o_i = 0, \quad \text{if } s_i^{\min} > \lambda_{\text{scale}}, \tag{18}$$

where λ_{scale} controls the maximum permitted thickness of a Gaussian along its normal direction.

4. Experiments

4.1. Implementation and experimental setup

Experiments were conducted on a workstation equipped with an Intel E5-2690 v4 2.60 GHz 14-core 28-thread CPU and a single NVIDIA 3090 GPU. We evaluated the accuracy on the TUM RGB-D dataset [45] and the Replica dataset [46]. The evaluation focuses on the bounded indoor environments provided by TUM RGB-D and Replica. These datasets are widely used in 3DGS-based SLAM and provide complementary conditions: TUM RGB-D contains real handheld sensor observations, whereas Replica provides rendered sequences with accurate scene geometry and

Table 2
Comparison of tracking results (RMSE of ATE in cm) on the TUM dataset. Red, blue and green values indicate the best, second-best and third-best performance, respectively.

Input	Type	Method	fr1_desk	fr2_xyz	fr3_office	Avg.
RGB	Traditional SLAM	DSO [20]	22.4	1.10	9.50	11.0
		ORB-SLAM2 [14]	2.00	0.60	2.30	1.60
	Learning-based SLAM	DROID-VO [21]	5.20	10.70	7.30	7.73
		DROID-SLAM [21]	1.80	0.50	2.80	1.70
	3DGS-based SLAM	MonoGS [10]	4.15	4.79	4.39	4.44
		NA-SLAM (Ours)	3.13	2.24	2.34	2.57
RGB-D	NeRF-based SLAM	iMAP [37]	4.90	2.00	5.80	4.23
		NICE-SLAM [38]	4.26	6.19	6.87	5.77
		Vox-Fusion [39]	3.52	1.49	26.01	10.34
		ESLAM [40]	2.47	1.11	2.42	2.00
		Co-SLAM [41]	2.40	1.70	2.40	2.17
		Point-SLAM [42]	4.34	1.31	3.48	3.04
	3DGS-based SLAM	MonoGS [10]	1.52	1.58	1.65	1.58
		SplaTAM [32]	3.35	1.24	5.16	3.25
		GS-SLAM [34]	3.30	1.32	6.60	3.70
		NA-SLAM (Ours)	1.48	1.27	1.46	1.40

camera trajectories. They enable evaluation with RGB and RGB-D inputs and facilitate comparison with existing methods. However, the current experiments do not cover large-scale or outdoor environments; therefore, the reported results should not be interpreted as evidence of generalization to these settings.

We follow the hyperparameter settings of MonoGS for implementation consistency. In particular, the hyperparameter λ_{scale} is fixed at 0.1. For a fair comparison, all basic keyframe-related parameters are kept identical to those of MonoGS [10]. Specifically, the keyframe window size is set to N = 8, the IoU and OC thresholds are set to 0.95 and 0.3, respectively, and the regular and forced motion thresholds are set to 1/0.05 and 1/0.08. Keyframe selection is evaluated for every incoming frame, while redundant-keyframe removal is triggered whenever a new keyframe is inserted. The only modification to the baseline selection criterion is the rotation-dependent factor $f_s(W)$, which enables the threshold to adapt jointly to rotational and translational camera motion. The normal constraint is evaluated at every mapping iteration. Normal gradients are computed using neighboring-pixel differences, with valid-pixel and rendered-opacity masks applied to both RGB and RGB-D inputs; an additional valid-depth mask is enabled for RGB-D input. Normal-guided Gaussian filtering is synchronized with the original MonoGS pruning operation and is triggered when $n_{map} \bmod 150 = 50$. The optimization schedule follows MonoGS: the first frame is initialized using 1050 mapping iterations, camera tracking uses at most 100 iterations per input frame, and each mapping update uses 150 iterations. All remaining optimizer and learning-rate settings are unchanged from the corresponding MonoGS configurations.

Our comparison methods include traditional SLAM methods (DSO [20], ORB-SLAM2 [14]), deep learning-based SLAM (DROID-SLAM [21]) and its standalone visual odometry component (DROID-VO [21]), NeRF-based SLAM methods (iMAP [37], NICE-SLAM [38], Vox-Fusion [39], ESLAM [40], Co-SLAM [41], Point-SLAM [42]), and 3D Gaussian splatting-based SLAM methods (MonoGS [10], SplaTAM [32], GS-SLAM [34]).

To demonstrate the effectiveness of our method, we carry out comparative evaluations with the aforementioned approaches focusing on localization accuracy and reconstruction quality. We further compare the storage efficiency of our approach with the baseline, and also conduct ablation studies to confirm the necessity of the proposed strategy combinations.

4.2. Tracking accuracy

To assess camera tracking accuracy, we report the root mean square error (RMSE) of the absolute trajectory error (ATE) of the keyframes. For map quality evaluation, we use standard photometric rendering quality metrics, PSNR, SSIM and LPIPS, following the evaluation protocol outlined in [10] and [32].

In Table 2, NA-SLAM achieves improved localization accuracy compared with MonoGS, outperforming existing monocular 3DGS-based SLAM methods, across all three sequences. In the third sequence **fr3_office**, it approaches the accuracy of the traditional feature-based method with loop closure, ORB-SLAM2. With RGB-D input, NA-SLAM achieves the overall best results. In comparison to the systems utilizing only RGB inputs, the incorporation of depth input provides additional geometric constraints.

As shown in Table 3, NA-SLAM outperforms the baseline algorithm MonoGS in all sequences and overall average in the Replica dataset. Moreover, NA-SLAM achieves the best overall localization accuracy among the compared methods.

The comparison of localization accuracy across the two datasets demonstrates the effectiveness of our proposed self-supervised normal constraint, the associated keyframe strategy, and the normal-guided filtering strategy. In particular, with RGB input, our method effectively constrains mapping optimization, thereby improving localization accuracy.

4.3. Reconstruction performance

Fig. 5 depicts the rendering results of our NA-SLAM and the baseline method MonoGS together with the ground truth (GT) for the TUM dataset with RGB input only. As demonstrated in the figure, our proposed method facilitates the rapid convergence of 3D Gaussian ellipsoids to their accurate positions and shapes without the need for depth priors. In particular, on the *fr3_office* sequence, our method reconstructs more background details, even when these areas are observed from limited viewpoints. Table 4 compares the reconstruction performance of our approach with the baseline MonoGS on the TUM dataset. NA-SLAM consistently improves PSNR across all three sequences, although the improvements are modest. Under RGB input, the average PSNR increases from 21.85 to 22.20 dB, while under RGB-D input it increases from 24.16 to 24.38 dB. The remaining metrics exhibit mixed results. Under RGB input, the average SSIM increases slightly from 0.731 to 0.737, whereas LPIPS changes from 0.333 to 0.339. Under RGB-D input, SSIM remains nearly unchanged, changing from 0.802 to 0.801, while LPIPS changes from 0.228 to 0.232. These results indicate that the improvement on TUM is primarily reflected in pixel-wise reconstruction fidelity and does not represent a consistent improvement in perceptual rendering quality. The mixed results may reflect a trade-off between appearance fitting and the geometric regularization imposed on the Gaussian representation.

Table 3
Comparison of tracking results (RMSE of ATE in cm) on the Replica dataset. Red, blue and green values indicate the best, second-best and third-best performance, respectively.

Method	r0	r1	r2	o0	o1	o2	o3	o4	Avg.
iMAP [37]	3.12	2.54	2.31	1.69	1.03	3.99	4.05	1.93	2.58
NICE-SLAM [38]	0.97	1.31	1.07	0.88	1.00	1.06	1.10	1.13	1.07
Vox-Fusion [39]	1.37	4.70	1.47	8.48	2.04	2.58	1.11	2.94	3.09
ESLAM [40]	0.71	0.70	0.52	0.57	0.55	0.58	0.72	0.63	0.63
Point-SLAM [42]	0.61	0.41	0.37	0.38	0.48	0.54	0.69	0.72	0.53
MonoGS [10]	0.47	0.43	0.31	0.70	0.57	0.31	0.31	3.20	0.79
NA-SLAM (Ours)	0.42	0.26	0.31	0.46	0.40	0.25	0.22	1.77	0.51

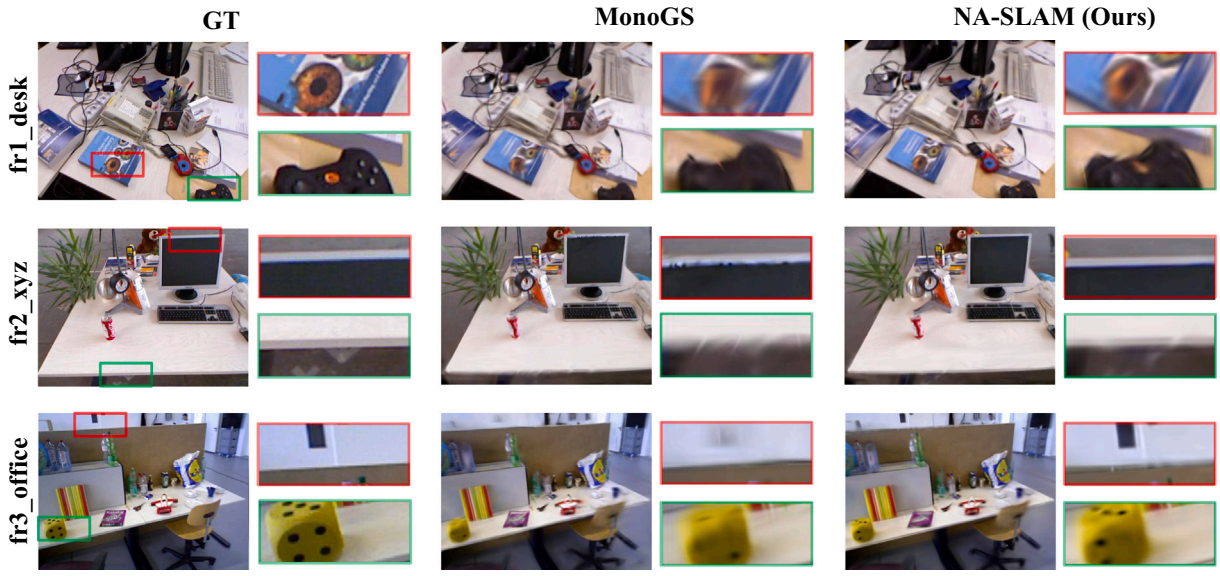


Fig. 5. Rendering examples on the TUM dataset.

Table 4
Comparison of rendering results on the TUM dataset.

Input	Method	Metric	fr1_desk	fr2_xyz	fr3_office	Avg.
RGB	MonoGS	PSNR↑	21.03	22.08	22.44	**21.85**
		SSIM↑	0.706	0.720	0.767	**0.731**
		LPIPS↓	0.353	0.297	0.350	**0.333**
	NA-SLAM	PSNR↑	21.20	22.71	22.70	**22.20**
		SSIM↑	0.711	0.730	0.771	**0.737**
		LPIPS↓	0.365	0.305	0.348	**0.339**
RGB-D	MonoGS	PSNR↑	23.20	24.68	24.60	**24.16**
		SSIM↑	0.780	0.795	0.830	**0.802**
		LPIPS↓	0.251	0.216	0.218	**0.228**
	NA-SLAM	PSNR↑	23.64	24.81	24.68	**24.38**
		SSIM↑	0.781	0.793	0.830	**0.801**
		LPIPS↓	0.252	0.222	0.221	**0.232**

The TUM sequences used in this study contain real handheld camera motion and incidental image degradation. However, the current experiments are not designed for a controlled robustness evaluation under severe motion blur, low illumination, or abrupt viewpoint changes. Since NA-SLAM relies on photometric consistency for tracking and map optimization, these conditions may reduce usable image gradients and inter-frame overlap, thereby degrading pose estimation and normal-constrained mapping. The proposed method primarily addresses geometric ambiguity in 3DGS-based SLAM rather than image degradation at the sensor level. Accordingly, robustness under severe illumination and motion degradation remains a limitation to be investigated in future work.

Fig. 6 compares the rendering results of our NA-SLAM method with the baseline MonoGS on the RGB-D Replica dataset. The results demonstrate that our method converges rapidly to high-fidelity reconstructions and reduces visible artifacts near uncertain boundaries. Table 5 shows our reconstruction performance on the Replica dataset. Compared to other methods, NA-SLAM achieves the best results in both PSNR and LPIPS metrics. Although NA-SLAM achieves a slightly lower SSIM than the learning-based NeRF methods in the SSIM metric, we still obtain superior overall results compared to the baseline methods, demonstrating the effectiveness of our proposed constraints and strategies.

Replica does not provide explicit ground-truth normal maps for the evaluated views. We therefore derive reference normals from its rendered depth maps by back-projecting local depth neighborhoods into the camera coordinate system and estimating the corresponding surface directions. Since the RGB and depth observations are rendered from the same dense scene geometry, the resulting normals provide a geometrically consistent reference for qualitative comparison. However, these depth-derived reference normals are not used during the optimization of either MonoGS or NA-SLAM.

As shown in Fig. 7, NA-SLAM produces more spatially coherent normal fields than MonoGS over large, locally smooth surfaces, such as walls and floors. This behavior is consistent with the intended effect of the proposed edge-aware normal regularization. Importantly, NA-SLAM does not directly minimize the angular difference between the rendered Gaussian normals and the depth-derived reference normals. In regions containing strong RGB or depth gradients, the smoothness weight is reduced, allowing the Gaussian parameters to remain jointly determined by the photometric, depth, and multi-view objectives. Consequently, the proposed constraint encourages stable local normal orientations where



Fig. 6. Rendering examples on the Replica dataset.

Table 5
Comparison of rendering results on the Replica dataset. Red, blue and green values indicate the best, second-best and third-best performance, respectively.

Method	Metric	r0	r1	r2	o0	o1	o2	o3	o4	Avg.
NICE-SLAM [38]	PSNR(dB)↑	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 ↓	0.33	0.271	0.208	0.229	0.181	0.235	0.209	0.198	0.233
Vox-Fusion [39]	PSNR(dB)↑	22.39	22.36	23.92	27.79	29.83	20.33	23.47	25.21	24.41
	SSIM ↑	0.683	0.751	0.798	0.857	0.876	0.794	0.803	0.847	0.801
	LPIPS ↓	0.303	0.269	0.234	0.241	0.184	0.243	0.213	0.199	0.236
Point-SLAM [42]	PSNR(dB)↑	32.40	34.08	35.5	38.26	39.16	33.99	33.48	33.49	35.17
	SSIM ↑	0.974	0.977	0.982	0.983	0.986	0.96	0.960	0.979	0.975
	LPIPS ↓	0.113	0.116	0.111	0.1	0.118	0.156	0.132	0.142	0.124
MonoGS [10]	PSNR(dB)↑	34.83	36.43	37.49	39.95	42.09	36.24	36.7	36.07	37.50
	SSIM ↑	0.954	0.959	0.965	0.971	0.977	0.964	0.963	0.957	0.960
	LPIPS ↓	0.068	0.076	0.075	0.072	0.055	0.078	0.065	0.099	0.070
NA-SLAM (Ours)	PSNR(dB)↑	36.83	38.72	39.21	43.30	43.44	37.61	37.46	37.48	39.26
	SSIM ↑	0.969	0.971	0.974	0.984	0.985	0.973	0.971	0.963	0.974
	LPIPS ↓	0.047	0.058	0.059	0.035	0.031	0.048	0.046	0.078	0.050

a piecewise-smooth prior is appropriate without forcing all Gaussians to reproduce a predefined normal field. This comparison should therefore be interpreted as qualitative evidence of functional normal consistency rather than as a supervised surface-normal estimation benchmark.

4.4. Gaussian map size analysis

Table 6 compares the final Gaussian map sizes of MonoGS and NA-SLAM under RGB and RGB-D inputs. Under RGB input, NA-SLAM consistently uses fewer Gaussian primitives while maintaining or improving localization and rendering performance. This result suggests that the proposed normal regularization and filtering strategy suppresses unnecessary or geometrically unreliable primitives when the scene geometry is inferred solely from multi-view RGB observations.

Under RGB-D input, the effect on map size varies with the characteristics of the depth observations. As illustrated in Fig. 8, the depth maps in the TUM RGB-D dataset are captured using a physical Microsoft Kinect sensor and are therefore affected by measurement noise, missing values, and boundary artifacts. Although the proposed method does not directly modify the input depth, the depth-guided normal regularization limits the propagation of these uncertainties into Gaussian geometry, and the scale-based filtering removes primitives with excessive thickness along their normal axes. This contributes to the reduction in Gaussian count observed on the TUM sequences. In contrast, the Replica RGB-D sequences are rendered from high-quality reconstructed scene models and provide dense, geometrically consistent depth observations. Since the depth measurements already impose strong constraints on Gaussian geometry, fewer primitives violate the filtering criterion, and the final map size is influenced more strongly by map expansion and densification. Consequently, the average Gaussian count on Replica increases modestly from $29.48 \times 10^{\wedge} 4$ to $30.69 \times 10^{\wedge} 4$, corresponding to approximately 4.1%, while the average storage increases from 19.06 MB to 20.00 MB.

The final Gaussian count reflects the combined effects of primitive generation, optimization, densification, and filtering, and therefore does not directly quantify representation redundancy. On Replica, the modest increase in map size is accompanied by improved localization and rendering performance, suggesting that the additional primitives are used to represent scene coverage or local details rather than simply duplicating existing geometry. Accordingly, the proposed filtering strategy is intended to improve the geometric reliability and compactness of the Gaussian representation, rather than to explicitly minimize the number of primitives in every sequence.

4.5. Ablation study

Table 7 reports the ablation results for normal-guided filtering, rotation-guided keyframe management, and the normal constraint on the TUM RGB sequences. The complete configuration achieves the lowest average ATE, demonstrating that the three components are complementary. However, the results do not indicate a monotonic improvement whenever an additional component is enabled. In particular, applying the normal constraint alone causes tracking failure on the **fr1** sequence.

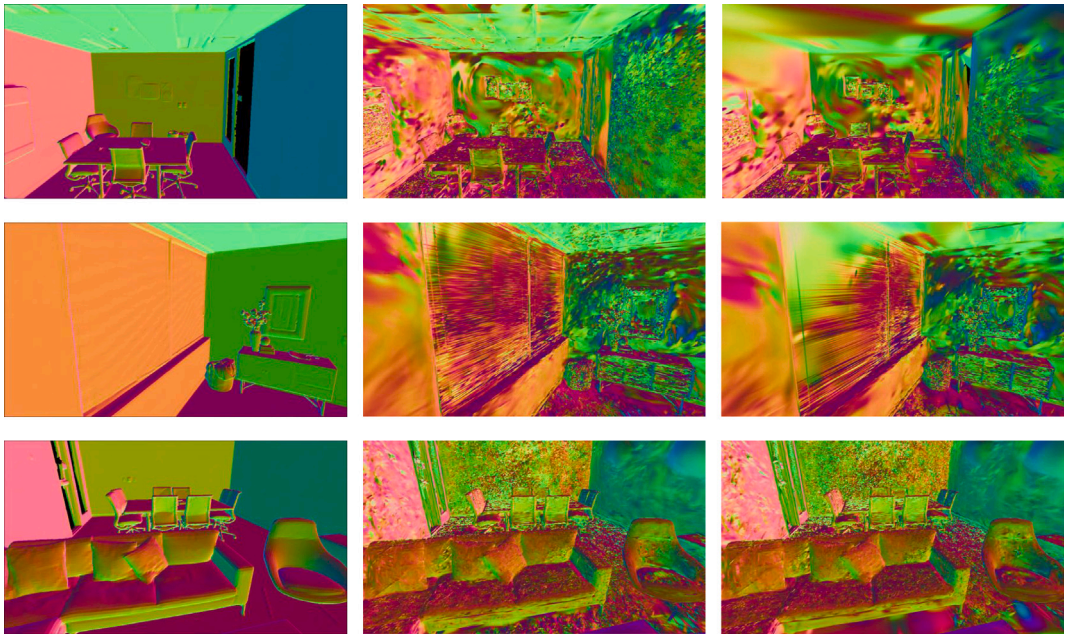


Fig. 7. Qualitative comparison of rendered Gaussian normals on representative Replica views. The reference normals are derived from the rendered Replica depth maps and are used only for qualitative comparison; they are not used to supervise or optimize either method. NA-SLAM produces spatially coherent normal fields on large planar regions while retaining sufficient flexibility around complex textures and structural transitions.

Table 6
Comparison of Gaussian map sizes under RGB and RGB-D inputs.

Method	Metric	TUM							Replica								
		RGB			RGB-D												
		fr1	fr2	fr3	fr1	fr2	fr3	Avg.	r0	r1	r2	o0	o1	o2	o3	o4	Avg.
MonoGS	Storage (MB)	1.73	3.22	2.04	2.29	1.90	2.90	**2.35**	22.1	17.1	19.6	15.5	12.0	24.6	24.0	17.6	**19.06**
	Number of Gaussians ($\times 10^4$)	2.67	4.97	3.15	3.54	2.94	4.48	**3.63**	34.21	26.41	30.34	23.91	18.64	38.08	37.08	27.19	**29.48**
NA-SLAM (Ours)	Storage (MB)	0.92	1.38	1.58	1.28	1.49	2.17	**1.47**	23.3	21.8	20.2	13.0	16.7	22.6	21.8	20.6	**20.00**
	Number of Gaussians ($\times 10^4$)	1.41	2.14	2.44	1.98	2.31	3.35	**2.27**	35.93	32.04	31.25	20.14	25.76	34.89	33.66	31.81	**30.69**

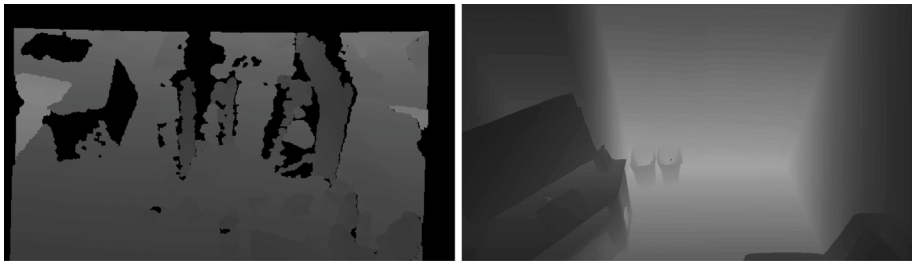


Fig. 8. Representative depth observations from the TUM RGB-D and Replica datasets. The TUM depth maps are acquired using a physical Kinect sensor and exhibit sensor-related missing measurements and boundary artifacts, whereas the Replica depth maps are rendered from reconstructed scene geometry and are comparatively dense and geometrically consistent.

The configurations containing normal-guided filtering generally achieve lower ATE than their unfiltered counterparts, which is consistent with its intended role of suppressing Gaussian primitives with unreliable normal-axis morphology. This improvement should not be attributed solely to a change in the final number of Gaussians, because the final map size is jointly affected by filtering, densification, and observation coverage.

The effectiveness of normal-constrained mapping depends on both the reliability of the estimated Gaussian normals and the availability of sufficient multi-view observations. Since the normal direction of a Gaussian is approximated by the shortest axis of its covariance ellipsoid, the estimated normal may become ambiguous when two or more axes have similar. Small changes in the optimized scales can then

Table 7
Ablation study on TUM RGB dataset showing RMSE of ATE in cm.

normal filtering	keyframe management	normal constraint	fr1	fr2	fr3	Avg.
×	×	×	4.15	4.79	4.39	4.44
✓	×	×	2.89	4.73	3.52	3.71
×	✓	×	3.93	4.78	4.14	4.28
×	×	✓	77.37	2.91	2.89	27.72
×	✓	✓	2.63	3.31	2.54	2.83
✓	×	✓	5.04	3.67	2.06	3.59
✓	✓	×	3.49	4.58	2.97	3.68
✓	✓	✓	**3.13**	**2.24**	**2.34**	**2.57**

alter the selected shortest-axis direction. Inaccurate image boundaries may also provide unreliable guidance for the edge-aware smoothness constraint. Moreover, under RGB-only input, the normal constraint provides local geometric regularization but does not supply absolute depth information. Newly inserted Gaussians may therefore remain underconstrained when they are observed from only a few viewpoints, particularly along trajectories with limited translational parallax.

We attribute the tracking failure of the normal-constraint-only configuration on **fr1**, resulting in an ATE of 77.37 cm, to the combined influence of these factors. The rotation-guided keyframe strategy improves viewpoint diversity and observation coverage by incorporating relative camera rotation into keyframe selection. Although keyframe management alone provides only a modest reduction in the average ATE, from 4.44 cm to 4.28 cm, its benefit becomes more evident when it is combined with the normal constraint: the ATE on **fr1** decreases from 77.37 cm to 2.63 cm. Normal-guided filtering provides an additional safeguard by removing Gaussians whose shortest-axis scales exceed the specified geometric threshold. Nevertheless, neither keyframe management nor filtering can recover geometric information that is absent from the observations. Consequently, ambiguous normal directions, insufficient multi-view observations, and limited translational parallax remain potential failure cases of the current framework.

Table 8 further reports the ablation results on the Replica RGB-D dataset. The complete configuration achieves the best average localization performance while maintaining rendering quality comparable to the other configurations. The differences among the rendering metrics are relatively modest, and individual configurations may perform better on particular metrics or sequences. Therefore, these results primarily demonstrate the contribution of the proposed components to localization accuracy and geometric stability, rather than a uniform improvement in every rendering metric.

Table 9
Hyperparameter sensitivity analysis on the TUM RGB dataset.

λ_S	λ_{scale}	ATE RMSE (cm) ↓	PSNR (dB) ↑	Number of Gaussians ($\times 10^4$)↓
0.10	**0.10**	**2.57**	**22.20**	**2.00**
0.05	0.10	3.12	21.81	2.96
0.20	0.10	2.88	22.09	2.15
0.10	0.05	3.16	22.17	3.49
0.10	0.20	3.24	21.99	2.13

4.6. Hyperparameter sensitivity analysis

We further evaluate the sensitivity of NA-SLAM to the normal loss weight λ_S and the normal filtering threshold λ_{scale}. A one-factor-at-a-time protocol is adopted for the three TUM RGB sequences. Specifically, λ_S is varied while fixing $\lambda_{scale} = 0.1$, and λ_{scale} is varied while fixing $\lambda_S = 0.1$. All other experimental settings remain unchanged. Table 9 reports the average ATE RMSE, PSNR, and number of Gaussian primitives over the three sequences.

The results show that the default setting, $\lambda_S = 0.1$ and $\lambda_{scale} = 0.1$, provides the best overall performance within the tested range. A smaller λ_S provides insufficient geometric regularization, whereas an excessively large weight may overemphasize normal smoothness relative to photometric consistency. Changing λ_{scale} in either direction also reduces tracking accuracy, indicating that overly relaxed or restrictive filtering can disturb the balance between retaining useful Gaussian primitives and suppressing unreliable representations. Notably, the average PSNR varies by less than 0.4 dB across all settings, while the ATE and Gaussian count exhibit larger variations. This suggests that the rendering quality of NA-SLAM is relatively stable within the tested range, whereas localization accuracy and map compactness are more sensitive to these parameters.

4.7. Runtime and GPU memory analysis

We further compare the runtime and GPU memory consumption of NA-SLAM with those of MonoGS on the TUM sequences. Both methods are evaluated using the hardware described in Section 4.1, with identical input sequences and optimization settings. CUDA operations are synchronized before timing. Data loading, first-frame initialization, and final backend refinement are excluded from the runtime measurements.

Table 8
Ablation study on Replica.

Method	Metric	r0	r1	r2	o0	o1	o2	o3	o4	Avg.
normal constraint	PSNR↑	36.11	38.26	39.09	43.11	43.41	36.77	36.99	36.64	**38.80**
	SSIM↑	0.964	0.968	0.973	0.984	0.982	0.958	0.966	0.960	**0.969**
	LPIPS↓	0.054	0.065	0.063	0.035	0.045	0.084	0.057	0.091	**0.062**
	RMSE(ATE)↓	0.63	0.39	0.50	0.55	0.83	0.29	0.27	2.32	**0.72**
normal filtering	PSNR↑	36.75	39.05	39.45	42.99	43.93	37.62	37.58	37.70	**39.38**
	SSIM↑	0.968	0.973	0.976	0.984	0.985	0.975	0.973	0.969	**0.975**
	LPIPS↓	0.045	0.051	0.051	0.032	0.030	0.043	0.040	0.059	**0.044**
	RMSE(ATE)↓	0.49	0.32	0.35	0.51	0.57	0.24	0.19	2.22	**0.61**
keyframe management	PSNR↑	36.63	38.76	39.41	43.47	43.94	37.63	37.59	37.30	**39.34**
	SSIM↑	0.968	0.972	0.976	0.985	0.985	0.975	0.974	0.968	**0.975**
	LPIPS↓	0.046	0.050	0.051	0.030	0.031	0.042	0.039	0.060	**0.043**
	RMSE(ATE)↓	0.48	0.36	0.51	0.47	0.58	0.24	0.23	2.62	**0.69**
NA-SLAM (Ours)	PSNR↑	36.83	38.72	39.21	43.30	43.44	37.61	37.46	37.48	**39.26**
	SSIM↑	0.969	0.971	0.974	0.984	0.985	0.973	0.971	0.963	**0.974**
	LPIPS↓	0.047	0.058	0.059	0.035	0.031	0.048	0.046	0.078	**0.050**
	RMSE(ATE)↓	0.42	0.26	0.31	0.46	0.40	0.25	0.22	1.77	**0.51**

Table 10
Runtime and GPU memory comparison on the TUM sequences.

Input	Method	Tracking time (ms/iter) ↓	Tracking time (ms/frame) ↓	Tracking FPS ↑	Mapping time (ms/iter) ↓	Frontend memory (MiB) ↓	Backend memory (MiB) ↓
RGB-D	MonoGS	11.76	816.86	1.27	115.11	978.85	887.86
	NA-SLAM	**10.60**	**709.06**	**1.45**	**100.50**	**938.66**	**657.61**
RGB	MonoGS	10.83	758.83	1.34	**103.56**	**1023.67**	817.41
	NA-SLAM	**9.72**	**674.73**	**1.52**	116.94	1072.99	**790.09**

Tracking allows at most 100 pose-optimization iterations per frame but terminates early when the pose-update criterion is satisfied. Therefore, the tracking time per frame is calculated from the accumulated time of the iterations actually executed, rather than from the maximum number of iterations. Tracking FPS is calculated as the number of processed frames divided by the accumulated frontend tracking time and consequently represents frontend tracking throughput. Mapping time is reported per optimization iteration. Since the frontend and backend operate in separate processes, their peak PyTorch-allocated GPU memory is measured separately. The peak-memory statistics cover the complete process lifetime, including initialization.

As shown in Table 10, NA-SLAM improves frontend tracking efficiency under both input modalities. For RGB input, the average tracking time decreases from 10.83 to 9.72 ms per iteration and from 758.83 to 674.73 ms per frame, while the tracking throughput increases from 1.34 to 1.52 FPS. The mapping time increases from 103.56 to 116.94 ms per iteration, corresponding to an overhead of approximately 12.9%. This additional cost arises from evaluating the normal-related losses during every mapping iteration. The frontend memory increases moderately by approximately 4.8%, while the backend memory remains comparable to that of MonoGS.

For RGB-D input, NA-SLAM reduces the average tracking time from 816.86 to 709.06 ms per frame and increases the tracking throughput from 1.27 to 1.45 FPS. The measured mapping time also decreases from 115.11 to 100.50 ms per iteration. Although the normal-related losses introduce additional operations, the net computational cost is also affected by the number of active Gaussian primitives. As reported in Table 6, NA-SLAM constructs smaller Gaussian maps on the evaluated TUM sequences. Suppressing geometrically unreliable primitives reduces the number of Gaussians involved in repeated rendering and gradient computation, thereby partially offsetting the cost of normal-constrained mapping.

The smaller and geometrically filtered Gaussian maps are consistent with the lower tracking time observed under both input modalities. In addition to reducing the rendering cost of each tracking iteration, improved map geometry may facilitate pose convergence and early termination. Overall, NA-SLAM achieves higher frontend tracking throughput on the evaluated TUM sequences while introducing only a moderate mapping overhead under RGB input and no substantial increase in GPU memory consumption.

5. Conclusions

In this paper, we presented NA-SLAM, a 3D Gaussian Splatting-based SLAM framework with self-supervised normal consistency constraints. The proposed method improves geometric coherence during map optimization by introducing normal-aware regularization without requiring explicit depth priors. In addition, we designed a rotation-guided keyframe management strategy and a normal-guided Gaussian filtering mechanism to improve viewpoint coverage and suppress geometrically unreliable Gaussian primitives.

Experimental results on the TUM RGB-D and Replica datasets demonstrate that the proposed method improves localization accuracy and reconstruction quality compared with existing 3DGS-based SLAM approaches. In particular, under RGB-only input for the evaluated TUM sequences, the proposed framework achieves better performance than MonoGS while using fewer Gaussian primitives.

The current evaluation is limited to static indoor scenes from the TUM RGB-D and Replica datasets. Consequently, the performance of NA-SLAM in large-scale, outdoor, and dynamic environments has not yet been established. Such environments introduce additional challenges, including unbounded map growth, long-term trajectory drift, substantial illumination changes, and dynamic objects. Moreover, the proposed normal-constrained mapping depends on sufficient overlapping multi-view observations. When the camera trajectory provides limited viewpoint overlap or translational parallax, the positions, scales, and normal directions of Gaussian primitives may remain underconstrained, particularly under monocular RGB input. Although the proposed keyframe management and Gaussian filtering strategies alleviate these problems, they cannot fully compensate for missing geometric observations. Future work will investigate scalable map representations, long-term geometric consistency, and improved robustness in outdoor and dynamic scenes.

CRediT authorship contribution statement

Zhengwu Shi: Writing – original draft, Software, Methodology. **Yuan Rao:** Writing – review & editing, Validation. **Xiaowei Zhou:** Writing – review & editing, Conceptualization. **Baogui Huan:** Resources, Data curation. **Junyu Dong:** Writing – review & editing, Funding acquisition. **Hui Yu:** Writing – review & editing. **Sheng Chen:** Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This work was supported by the Key Technology Research and Development Program of Shandong Province (Grant No. 2024ZLGX06 and Grant No. 2025CXPT185), the Major Program of the Natural Science Foundation of Shandong Province (Grant No. ZR2024ZD04), and in part by the Leverhulme Trust through project VP1-2020-044.

Data availability

Data will be made available on request.

References

[1] A. Singandhupe, H.M. La, A review of SLAM techniques and security in autonomous driving, in: Proc. IRC 2019 (Naples, Italy), Feb. 25–27, 2019, pp. 602–607.
[2] J. Cheng, et al., A review of visual SLAM methods for autonomous driving vehicles, Eng. Appl. Artif. Intell. 114, 104992, pp. Sep 2022) 1–17.
[3] L. Jinyu, et al., Survey and evaluation of monocular visual-inertial SLAM algorithms for augmented reality, Virtual Real. Intell. Hardw. 1 (4) (Aug 2019) 386–410.
[4] D. Zou, P. Tan, W. Yu, Collaborative visual SLAM for multiple agents: a brief survey, Virtual Real. Intell. Hardw. 1 (5) (Oct 2019) 461–482.

[5] F. Tosi, et al., How NeRFs and 3D Gaussian splatting are reshaping SLAM: a survey, IEEE Trans. Rob. 42, 139, pp.2026) 1405–1427.
[6] B. Kerbl, et al., 3D Gaussian splatting for real-time radiance field rendering, ACM Trans. Graph. 42 (4 139, pp.2023) 1–14.
[7] G. Chen, W. Wang, A survey on 3D Gaussian splatting, ACM Comput. Surv. 58, 313, pp.2026) 1–39.
[8] S. Liu, et al., mg-slam: structure Gaussian splatting SLAM with Manhattan world hypothesis, IEEE Trans, Autom. Sci. Eng. 22 (Jul 2025) 17034–17049.
[9] L. Chen, et al., G^2-mapping: general Gaussian mapping for monocular, RGB-D, and LiDAR-inertial-visual systems, IEEE Trans, Autom. Sci. Eng. 22 (Apr 2025) 12347–12357.
[10] H. Matsuki, et al., Gaussian splatting SLAM, in: Proc. CVPR 2024 (Seattle WA, USA), Jun. 17–21, 2024, pp. 18039–18048.
[11] K. Fang, et al., 2D Gaussian primitive SLAM: real-time dense SLAM with 2D Gaussian primitives, Neurocomputing 671 (Mar 2026) 132722.
[12] G. Klein, D. Murray, Parallel tracking and mapping for small AR workspaces, in: Proc. 6th IEEE and ACM Int. Symp. Mixed and Augmented Reality (Nara, Japan) Nov. 13–16, 2007, pp. 225–234.
[13] R. Mur-Artal, J.M.M. Montiel, J.D. Tardos, orb-slam: a versatile and accurate monocular SLAM system, IEEE Trans. Robotics 31 (5) (Oct 2015) 1147–1163.
[14] R. Mur-Artal, J.D. Tardós, orb-slam2: an open-source SLAM system for monocular, stereo, and RGB-D cameras, IEEE Trans. Robotics 33 (5) (Oct 2017) 1255–1262.
[15] C. Campos, et al., orb-slam3: an accurate open-source library for visual, visual-inertial, and multimap SLAM, IEEE Trans. Robotics 37 (6) (Dec 2021) 1874–1890.
[16] J. Engel, T. Schöps, D. Cremers, lsd-slam: large-scale direct monocular SLAM, in: Proc. ECCV 2014 (Zurich, Switzerland), Sep. 6–12, 2014, pp. 834–849.
[17] C. Forster, M. Pizzoli, D. Scaramuzza, SVO: fast semi-direct monocular visual odometry, in: textitProc. ICRA 2014 (Hong Kong, China) May 31–Jun. 7, 2014, pp. 15–22.
[18] S. Wang, R. Clark, H. Wen, N. Trigoni, DeepVO: towards end-to-end visual odometry with deep recurrent convolutional neural networks, in: Proc. IEEE Int. Conf. Robot. Autom. (ICRA) (Singapore), May 29–Jun. 3, 2017, pp. 2043–2050.
[19] N. Yang, et al., Deep virtual stereo odometry: leveraging deep depth prediction for monocular direct sparse odometry, in: Proc. ECCV 2018 (Munich, Germany), Sep. 8–14, 2018, pp. 817–833.
[20] J. Engel, V. Koltun, D. Cremers, Direct sparse odometry, IEEE Trans, Pattern Anal. Mach. Intell. 40 (3) (Mar 2018) 611–625.
[21] Z. Teed, J. Deng, droid-SLAM: deep visual SLAM for monocular, stereo, and RGB-D cameras, in: Proc. NIPS 2021, Dec.6–14, 2021, pp. 16558–16569.
[22] Z. Teed, J. Deng, RAFT: recurrent all-pairs field transforms for optical flow, in: Proc. ECCV 2020 (Glasgow, UK), Aug. 23–28, 2020, pp. 402–419.
[23] L. Han, et al., DeepVIO: self-supervised deep learning of monocular visual inertial odometry using 3D geometric constraints, in: Proc. IROS 2019 (Macau, China) Nov. 3–8, 2019, pp. 6906–6913.
[24] N. Yang, et al., D3VO: deep depth, deep pose and deep uncertainty for monocular visual odometry, in: Proc. CVPR 2020 (Seattle, WA, USA), Jun. 13–19, 2020, pp. 1281–1292.
[25] S. Wang, et al., DUSt3R: geometric 3D vision made easy, in: Proc. CVPR 2024 (Seattle, WA, USA), Jun. 17–21, 2024, pp. 20697–20709.
[26] J. Wang, et al., VGGT: visual geometry grounded transformer, in: Proc. CVPR 2025 (Nashville, TN, USA), Jun. 11–15, 2025, pp. 5294–5306.
[27] Y. Liu, et al., SLAM3R: real-time dense scene reconstruction from monocular RGB videos, in: Proc. CVPR 2025 (Nashville, TN, USA), Jun. 11–15, 2025, pp. 16651–16662.
[28] D. Maggio, H. Lim, L. Carlone, vggt-slam: dense RGB SLAM optimized on the SL(4) manifold, in: Proc. NIPS 2025, Dec.2–7, 2025, pp. 129839–129867.
[29] H. Huang, et al., Photo-SLAM: real-time simultaneous localization and photorealistic mapping for monocular stereo and RGB-D cameras, in: Proc. CVPR 2024 (Seattle, WA, USA), Jun. 17–21, 2024, pp. 21584–21593.
[30] S. Ha, J. Yeon, H. Yu, RGBD gs-icp SLAM, in: Proc. ECCV 2024 (Milan, Italy), Sep. 29–Oct. 4, 2025, pp. 180–197.
[31] Y.S. Hu, et al., MGSO: monocular real-time photometric SLAM with efficient 3D Gaussian splatting, in: Proc. ICRA 2025 (Atlanta, GA, USA), May 19–23, 2025, pp. 11061–11067.
[32] N. Keetha, et al., SplaTAM: splat track & map 3D Gaussians for dense RGB-D SLAM, in: Proc. CVPR 2024 (Seattle, WA, USA), Jun. 17–21, 2024, pp. 21357–21366.
[33] M. Mansour, et al., udgs-slam: unidepth assisted Gaussian splatting for monocular SLAM, Array 26 (Jul 2025) 100400.
[34] C. Yan, et al., gs-slam: dense visual SLAM with 3D Gaussian splatting, in: Proc. CVPR 2024 (Seattle, WA, USA), Jun. 17–21, 2024, pp. 19595–19604.
[35] Y. Zhou, et al., ousa-gs: real-time structurally-consistent 3D reconstruction of outdoor unbounded scenes via adaptive Gaussian splatting, Knowl.-Based Syst. 331 (Jan 2026) 114834.
[36] Z. Zhao, et al., figs-slam: Gaussian splatting SLAM with dynamic frequency control and influence-based pruning, Expert Syst. Appl. 294 (Jun 2025) 128763.
[37] E. Sucar, et al., iMAP: implicit mapping and positioning in real-time, in: Proc. ICCV 2021 (Montreal, QC, Canada), Oct. 10–17, 2021, pp. 6229–6238.
[38] Z. Zhu, et al., nice-SLAM: neural implicit scalable encoding for SLAM, in: Proc. CVPR 2022 (New Orleans, LA, USA) Jun. 18–24, 2022, pp. 12786–12796.
[39] X. Yang, et al., Vox-fusion: dense tracking and mapping with voxel-based neural implicit representation, in: Proc. ISMAR 2022 (Singapore), Oct. 17–21, 2022, pp. 499–507.
[40] M.M. Johari, C. Carta, F. Fleuret, ESLAM: efficient dense SLAM system based on hybrid representation of signed distance fields, in: Proc. CVPR 2023 (Vancouver, BC, Canada), Jun. 17–24, 2023, pp. 17408–17419.
[41] H. Wang, J. Wang, L. Agapito, Co-SLAM: joint coordinate and sparse parametric encodings for neural real-time SLAM, in: Proc. CVPR 2023 (Vancouver, BC, Canada), Jun. 17–24, 2023, pp. 17–24, pp. 13293–13302.
[42] E. Sandström, et al., Point-SLAM: dense neural point cloud-based SLAM, in: Proc. ICCV 2023 (Paris, France), Oct. 1–6, 2023, pp. 18433–18444.
[43] C. Godard, et al., Digging into self-supervised monocular depth estimation, in: Proc. ICCV 2019 (Seoul, South Korea), Oct. 27–Nov. 2, 2019, pp. 3828–3838.
[44] N. Zhang, et al., Lite-Mono: a lightweight CNN and transformer architecture for self-supervised monocular depth estimation, in: Proc. CVPR 2023 (Vancouver, BC, Canada), Jun. 17–24, 2023, pp. 18537–18546.
[45] J. Sturm, et al., A benchmark for the evaluation of RGB-D SLAM systems, in: Proc. 2012 IEEE/RSJ Int. Conf. Intelligent Robots and Systems, Vilamoura, Algarve, Portugal, Oct 7–12, 2012, pp. 573–580.
[46] J. Straub, et al., The replica dataset: a digital replica of indoor spaces, arXiv preprint arXiv:1906.05797, 2019.

Author biography



Zhengwu Shi received the B.Sc. degree from Shandong University of Science and Technology, Qingdao, China, in 2017, and the M.Sc. degree from Ocean University of China, Qingdao, China, in 2021. He is currently pursuing the Eng.D. degree with the Faculty of Information Science and Engineering, Ocean University of China. His research interests include computer vision and autonomous robotics.



Yuan Rao received B.Sc. degree from Ocean University of China, Qingdao, China, in 2017, M.E. degree from Dalian University of Technology, Dalian, China, in 2019, and Ph.D. degree from Ocean University of China, Qingdao, China, in 2024. He is currently a Lecturer with the Faculty of Information Science and Engineering at Ocean University of China. His research interests include underwater image processing, computer vision, and autonomous robotics.



Xiaowei Zhou received the B.Eng. and M.Eng. degrees in computer science from Ocean University of China, Qingdao, China, in 2014 and 2017, respectively, and the Ph.D. degree in computer science from University of Technology Sydney, Ultimo, NSW, Australia, in 2023. He is currently a Lecturer with the Faculty of Information Science and Engineering, Ocean University of China. His research interests include data mining and image processing.



Baogui Huan received the B.Eng. degree in communication engineering from Shandong University, Jinan, China, in 2007, and the M.Eng. degree in communication and information systems from Xi'an Technological University, Xi'an, China, in 2012. He is currently pursuing the Ph.D. degree in artificial intelligence at the Faculty of Information Science and Engineering, Ocean University of China, Qingdao, China. His research interests include remote sensing image processing and computer vision.



Junyu Dong received the B.Sc. and M.Sc. degrees from the Department of Applied Mathematics, Ocean University of China, in 1993 and 1999, respectively, and the Ph.D. degree in image processing from the Department of Computer Science, Heriot-Watt University, U.K., in November 2003. He joined the Ocean University of China in 2004. He is with the Faculty of Information Science and Engineering, Ocean University of China, Qingdao, Shandong 266100, China, and also with the State Key Laboratory of Physical Oceanography, Ocean University of China, Qingdao, Shandong 266100, China. His research interests include machine learning, big data, computer vision, and underwater image processing.



Hui Yu (Senior Member, IEEE) is a Professor of Visual and Cognitive Computing at the University of Glasgow, UK, where he leads the Visual Computing and Social Robot Group. His research interests include visual and cognitive computing and machine learning, with applications to image and video understanding, 3D/4D facial processing, social vision and robotics, human-machine interaction, and intelligent vehicles. He received an Industrial Fellowship from the Royal Academy of Engineering. He serves as an Associate Editor for IEEE Transactions on Human-Machine Systems, IEEE Transactions on Intelligent Vehicles, and IEEE/CAA Journal of Automatica Sinica.



Sheng Chen (IEEE Life Fellow) is an Emeritus Professor, University of Southampton, Southampton, U.K. He received the B.Eng. degree in control engineering from the East China Petroleum Institute, Dongying, China, in 1982, the Ph.D. degree in control engineering from City University, London, in 1986, and the higher doctoral (D.Sc.) degree from the University of Southampton, Southampton, U.K., in 2005. From 1986 to 1999, he held research and academic appointments with the University of Sheffield, the University of Edinburgh, and the University of Portsmouth, all in U.K. In September 1999, he joined the School of Electronics and Computer Science, University of Southampton, and retried in June, 2026. Since then he is a University Visitor associated with the School of Electronics and Computer Science, University of Southampton. In 2023, Dr Chen was appointed to a Distinguished Professor at Ocean University of China. Professor Chen has established an over 40-year distinguished academic career in the fields of adaptive signal processing, wireless communications, neural network and machine learning. He has authored/coauthored over 750 research papers. He have 23,000+ Web of Science citations with h-index 67, and 44,000+ Google Scholar citations with h-index 89. He is IEEE ComSoc Signal Processing and Computing for Communications (SPCC) 2025 Technical Recognition Award recipient. Professor Chen was elected to a fellow of the United Kingdom Royal Academy of Engineering in 2014. He is a fellow of Asia-Pacific Artificial Intelligence Association, a fellow of Industry Academy of the International Artificial Intelligence Industry Alliance, a fellow of IET, and one of the original 200 ISI Highly Cited Researcher in engineering (March 2004).