2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

UP-SLAM: Adaptively Structured Gaussian SLAM with Uncertainty Prediction in Dynamic Environments

Wancai Zheng, Linlin Ou, Jiajie He, Libo Zhou, Yan Wei, Xinyi Yu

Abstract— **Recent 3D Gaussian Splatting (3DGS) techniques for visual Simultaneous Localization and Mapping (SLAM) have significantly progressed in tracking and high-fidelity mapping. However, their sequential optimization framework and sensitivity to dynamic objects limit real-time performance and robustness in real-world scenarios. We present UP-SLAM, a real-time RGB-D SLAM system for dynamic environments that decouples tracking and mapping through a parallelized framework. A probabilistic anchor is employed to manage Gaussian primitives adaptively, enabling efficient initialization and pruning without hand-crafted thresholds. To robustly filter dynamic regions during tracking, we propose a training-free uncertainty estimator that fuses multi-modal residuals to estimate per-pixel motion uncertainty, achieving open-set dynamic object handling without reliance on semantic labels. Furthermore, a temporal encoder is designed to enhance rendering quality, while a shallow multilayer perception transforms low-dimensional features into DINO features, enriching the Gaussian field and enhancing uncertainty prediction robustness. Extensive experiments on multiple challenging datasets suggest that UP-SLAM outperforms state-of-the-art methods in both localization accuracy (by 59.8%) and rendering quality (by 4.72 dB PSNR), while maintaining real-time performance and producing reusable, artifact-free static maps in dynamic environments. The Project Page.**

I. Introduction

Visual Simultaneous Localization and Mapping (SLAM) is a core technology for embodied intelligence and virtual reality. Traditional SLAM [1]–[4] assumes static environments, limiting its applicability in dynamic real-world scenarios. Recent approaches [5], [6] employ object detection and multi-view geometry to handle dynamic objects, but they rely heavily on prior knowledge and detection reliability.

Advances in high-fidelity scene representations, such as Neural Radiance Fields [7] (NeRF) and 3D Gaussian Splatting [8] (3DGS), have motivated interest in introducing uncertainty modeling into 3D reconstruction. Recent studies [9]–[11] show that incorporating uncertainty prediction can significantly enhance robustness to transient scene elements. These uncertainty-aware models can achieve high-quality reconstructions even under intermittent occlusions. However, these methods depend on advantageous conditions, such as accurate camera poses and sparse viewpoints, which are challenging to achieve in SLAM systems using continuous frame inputs. WildGS-SLAM [12] addresses these issues through a sequential uncertainty-estimation pipeline, but it faces challenges in achieving high-precision localization in real time and requires re-optimization after the image sequence to improve accuracy. Moreover, its coupled tracking and mapping design, in which camera pose estimation is followed by scene optimization and dynamic object recognition is performed after map convergence, further constrains real-time performance.

To address these challenges, a real-time RGB-D SLAM system named UP-SLAM is presented for robust pose estimation and static scene rendering in dynamic environments. Our approach compresses 3DGS into structured anchors encoded by multiple shallow multilayer perceptrons (MLPs). A probabilistic attribute is introduced to enable adaptive adjustment of anchors to delete redundant anchors caused by dynamic objects. Furthermore, by decoupling motion mask generation from map optimization, UP-SLAM enables parallel tracking and mapping, supporting real-time localization. In the tracking process, we propose a training-free, optimization-based multi-modal consistency estimation method that fuses geometric cues with DINO features for effective dynamic object recognition. In the mapping process, to further enhance rendering under dynamic conditions, a temporal encoder that leverages sinusoidal positional encoding is designed to embed inter-frame information into the MLP, thereby increasing the representational capacity. In addition, the inconsistent appearance and motion of dynamic objects across frames provide valuable cues for uncertainty prediction. Therefore, robust DINO features are fed into a shallow MLP for per-pixel uncertainty estimation, enabling continuous motion mask refinement and enhancing reconstruction robustness. Our primary contributions are as follows:

- We propose an uncertainty-aware parallel tracking and mapping framework that decouples motion mask generation from map optimization, mitigating dynamic interference while enabling real-time localization and artifact-free map rendering without semantic priors.
- We propose an adaptive structured 3DGS scene representation with a probabilistic model, which supports automatic allocation and pruning of Gaussian primitives in dynamic environments. This approach enhances localization accuracy and reduces model size.
- We integrated our method into ORB-SLAM3 [1] and evaluated it on multiple datasets. The results demon-

This research was supported by the Baima Lake Laboratory Joint Funds of the Zhejiang Provincial Natural Science Foundation under Grant No. LBMHD24F0300023, and the National Natural Science Foundation of China under Grants 62373329, the Zhejiang Natural Science Foundation under Grant LZ25F03000. (Corresponding authors: Xinyi Yu and Linlin Ou)

The authors are with the College of Information Engineering, Zhejiang University of Technology, Hangzhou 310023, China yuxy@zjut.edu.cn, linlinou@zjut.edu.cn

strate that our approach remains competitive with the state of the art in localization accuracy, real-time performance, rendering quality, and model size.

II. Related work

A. Traditional Visual SLAM

Over the past few decades, visual SLAM has made remarkable progress, leading to the development of numerous outstanding algorithms [1]–[3]. However, these methods generally assume a static environment and neglect the velocity components in observations, making them susceptible to severe drift when dynamic objects are present. To address this issue, dynamic SLAM methods [13], [14] based on object detection or semantic segmentation have been proposed. DynaSLAM [5] leverages predefined semantic labels from Mask R-CNN [15], combined with geometric constraints, to reduce the influence of dynamic objects. However, this dependence on prior labels significantly limits the generalizability of the method. To reduce reliance on semantic priors, the method [16] takes advantage of spatial correlations between 3D points to eliminate dynamic features, while other methods [17], [18] detect inconsistencies between point clouds to localize moving objects. Most existing methods emphasize tracking robustness but neglect the construction of semantically enriched and reusable maps, which are critical for high-level tasks such as navigation and scene understanding. To bridge this gap, we design to efficiently distill high-dimensional visual features from DINOv2 [19] into the 3DGS representation, enabling the construction of a dense, feature-rich map that serves as a reliable foundation for downstream robotic tasks.

B. NeRF and 3DGS SLAM

a) NeRF SLAM: Neural implicit SLAM methods have recently gained attention for their ability to reconstruct high-fidelity scenes with continuous representations. iMAP [20] first introduced volumetric NeRF representations into SLAM using a single MLP and feature grid, but its long-term localization suffers from catastrophic forgetting. NICE-SLAM [21] alleviates this with a hierarchical grid-based structure, improving scalability and real-time performance. Later works [22]–[24] integrate signed distance fields into hybrid representations, enhancing both reconstruction quality and efficiency. Although effective in static scenes, these methods degrade under dynamic conditions, motivating extensions of NeRF-based SLAM to dynamic environments. For example, RodynSLAM [25] combines semantic priors and an optical flow estimator to mask dynamic regions. The method [26] pre-trains a dynamic object classifier and incrementally updates it by feeding features from objects identified through residuals between the map and ground truth. This enables the system to learn new dynamic features over time. However, such approaches heavily rely on prior knowledge of object classes, making them less generalizable to real-world open-set environments that contain unknown dynamic objects.

b) 3DGS SLAM: Recently, the emergence of 3DGS techniques, leveraging GPU tile-based acceleration frameworks, has enabled extremely high-frequency rendering. Several 3DGS-based SLAM systems have been proposed [27]–[30]. SplaTAM [31] proposes a silhouette tracking strategy and uses geometric cues to initialize Gaussians in under-reconstructed regions. However, the coupling between tracking and mapping limits real-time performance and robotic deployment. To address this, Photo-SLAM [32] decouples tracking and mapping by using ORB-SLAM3 for real-time pose estimation and introducing an image pyramid optimization to enhance reconstruction, achieving real-time tracking and high-quality mapping on embedded devices. Nevertheless, these methods still face challenges in dynamic environments. DG-SLAM [33] addresses this by combining semantic segmentation with motion masks derived from spatial geometry consistency across frames, thereby removing the need for prior knowledge of object classes. Gassidy [34] employs segmentation and a Gaussian mixture model to mask dynamic objects for robust tracking and high-quality map reconstruction. These methods obtain a fixed motion mask from the tracking, which can reduce the robustness of mapping. WildGS-SLAM [12] represents a recent advancement in dynamic SLAM by introducing an uncertainty-aware dynamic object recognition strategy. However, its inefficient Gaussian primitive management introduces high computational overhead, and the sequential pipeline further limits real-time performance. In contrast, tracking and mapping are decoupled in our method to improve efficiency, and DINO features are fed into a shallow MLP to support continuous refinement of the motion mask, thereby supporting real-time tracking and artifact-free, high-quality mapping in challenging dynamic environments.

III. Approach

Fig. 1 shows an overview of UP-SLAM, which takes RGB-D image sequences $\{D, C\} \in \mathbb{R}^{H\times W}$ as input and employs a parallel tracking and mapping architecture. The tracking thread (Sec. III-D) performs real-time localization and generates keyframes for mapping, with dynamic regions detected via multi-modal residuals from the mapping thread. The mapping thread (Sec. III-C) employs probabilistic anchors and uncertainty estimation to construct an adaptively structured 3DGS representation. Through rendering, we obtain images, depth maps, and low-dimensional DINO features, and the low-dimensional features are mapped to a high-dimensional space via an MLP.

A. Preliminaries

Following the vanilla 3DGS [8] framework, the entire scene is represented by a set of anisotropic Gaussian ellipsoids $\mathbf{G}$:

$$\mathbf{G} = \{G_i : (\mu_i, o_i, c_i, \Sigma_i) | i = 1, ..., N\}, \tag{1}$$

where each Gaussian is defined by its color $c \in \mathbb{R}^3$, opacity $o \in [0, 1]$, position $\mu \in \mathbb{R}^3$, and covariance matrix $\Sigma \in \mathbb{R}^{3\times3}$.

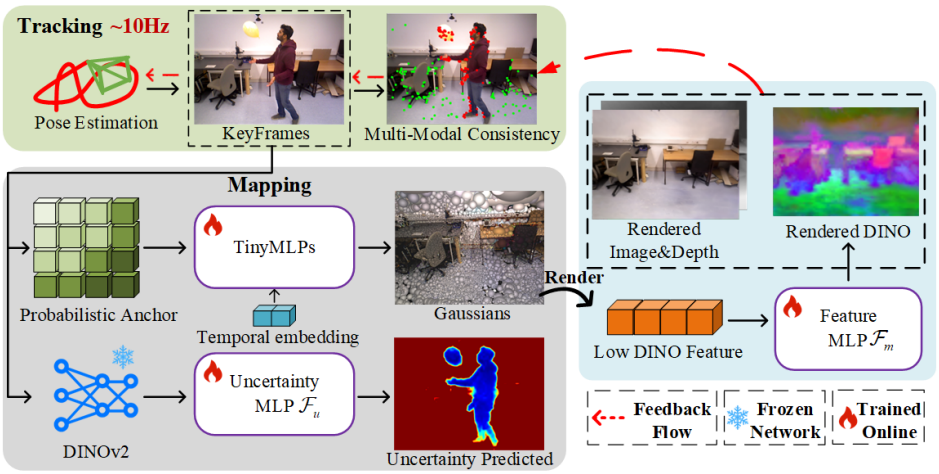

Fig. 1. **System overview.** UP-SLAM is a parallel tracking and mapping system that enables real-time localization and high-fidelity, artifact-free mapping.

The covariance matrix Σ is decomposed as $\Sigma_i = RSS^TR^T$, where S is a scale matrix and R is a rotation matrix.

The camera-to-world transformation T_{wc} is obtained from the pose estimation, after which each 3D Gaussian point G_i is projected onto the image plane for rendering, as follows:

$$\Sigma' = JT_{wc}^{-1}\Sigma T_{wc}^{-T}J^T, \tag{2}$$

where J denotes the Jacobian matrix of the affine approximation to the projective function. Following α-blending, the rendered color $\tilde{C}$, depth $\tilde{D}$, and accumulated transmittance $\tilde{T}$ are computed by aggregating Gaussian contributions along each ray, as formulated below:

$$\{\tilde{C},\tilde{D}\} = \sum_{i=1}^{N}\{c_i,z_i\}\sigma_i\prod_{j=1}^{i-1}(1-\sigma_j), \tilde{T} = \sum_{i=1}^{N}\sigma_i\prod_{j=1}^{i-1}(1-\sigma_j), \tag{3}$$

where c_i is the color of the i-th Gaussian, σ_i is defined by the Gaussian distribution and learned opacity o_i, and z_i denotes its depth in camera coordinates. In the optimization of Gaussian parameters, we incorporate geometric supervision as follows:

$$\begin{aligned} L_g = \lambda_1\big(\lambda\|\tilde{C}-C\|_2^2 + (1-\lambda)(1-\textbf{SSIM}(\tilde{C},C))\big) \\ + \lambda_2\|\tilde{D}-D\|_2^2, \end{aligned} \tag{4}$$

where **SSIM** is structural similarity index measure [35], $\{\lambda_*\}$ are hyperparameters.

B. Uncertainty Model

The uncertainty is modeled using a Bayesian learning framework, where the model predicts a Gaussian distribution to represent the uncertainty of each pixel, rather than outputting a single deterministic value. The uncertainty model is implemented via a lightweight MLP F_u. For each pixel, we compute the residual R between the rendered value and the ground truth, with σ denoting the predicted uncertainty. The loss for each pixel is defined as:

$$L_u = -\log\Big(\frac{1}{\sqrt{2\pi\sigma^2}}\exp\Big(-\frac{R}{2\sigma^2}\Big)\Big) = \frac{R}{2\sigma^2} + \lambda_3\log\sigma. \tag{5}$$

The first term is regularized by the second term, which corresponds to the log-partition function of the normal distribution and prevents a trivial minimum at $\sigma = \infty$ [36].

The DINO features are inherently robust to appearance variations across frames [10], making them well-suited for dynamic scenes with inconsistent appearance features. Therefore, DINO features are incorporated into both color and depth information to achieve joint constraints across appearance, geometry, and semantics. Additionally, we employ the accumulated transmittance $\tilde{T}$ as a visibility mask to prevent low-opacity regions from contributing. The total residuals R are defined as:

$$\begin{aligned} R = (\tilde{T} < 0.1)\big(\lambda_1^t\big|\tilde{C}-C\big| + \lambda_2^t\big|\tilde{D}-(D\otimes\mathbf{B})\big| \\ + \lambda_3^t\left\lceil 1 - \frac{F\cdot\hat{F}}{\|F\|_2\|\hat{F}\|_2}\right\rceil^1\big). \end{aligned} \tag{6}$$

The $\mathbf{B}$ is a 3×3 box filter applied to depth via convolution ($\otimes$), and $\lceil\rceil^1$ indicates that the output is capped at 1. F denotes the visual features extracted by DINOv2 [19], while $\hat{F}$ signifies the decoded high-dimensional visual features. Since DINOv2 is defined per image patch, we perform bilinear interpolation to upsample it to the image size for similarity calculation.

C. Mapping

a) Adaptively Structured Gaussian: Existing 3DGS SLAM [27], [30], [31] techniques require the rapid identification of under-reconstructed regions and the initialization of new Gaussian primitives to improve tracking efficiency. However, the presence of dynamic objects introduces disturbances that significantly degrade their performance. In addition, poorly chosen thresholds can cause excessive GPU memory consumption, reduction in computational efficiency, and deterioration of rendering quality.

An incrementally probabilistic anchor method is proposed to achieve adaptively structured 3DGS, eliminating the need for complex threshold tuning and manual management of Gaussian primitives. Specifically, an MLP [37] decodes the k Gaussian attributes from the anchor features $\hat{f}_v$, along with the relative direction δ_{vc} and distance $\vec{d}_{vc}$ between the camera center and the anchor, where initial anchors are extracted from the downsampled depth map. In contrast, our anchors are equipped with probabilistic attributes, where the

probability value reflects the degree of motion at each anchor. This probabilistic representation is more suitable for dynamic environments. The probabilistic anchor update equation is as follows [38]:

$$P(n|z_{1:t}) = [1 + \frac{1 - P(n|z_t)}{P(n|z_t)} \frac{1 - P(n|z_{1:t-1})}{P(n|z_{1:t-1})} \frac{P(n)}{1 - P(n)}]^{-1}. \tag{7}$$

This update equation is based on Bayes theorem and requires a prior probability $P(n)$, the current observation z_t (i.e., whether the camera ray can reach the anchor point), and the likelihood model $P(n|z_{1:t-1})$ to update the dynamic probability of each anchor. $P(n \mid z_t)$ denotes the probability that anchor n is occupied, given the observation z_t.

b) Temporal Encoding: Methods such as [37], [39] introduce appearance embeddings into the color prediction to improve rendering quality in wild reconstruction. These methods are primarily designed to improve rendering quality through better representation learning. In the context of SLAM, the method [40] leverages the pose as an additional input to the MLP, utilizing the characteristics of SLAM to enhance performance. However, since the rotation matrix lies on the special orthogonal group $SO(3)$, a non-Euclidean manifold with nonlinear constraints, conventional MLPs struggle to model rotational variations effectively [41], leading to suboptimal performance. Given that SLAM operates on temporally correlated image sequences where pose evolution is time-dependent, we propose a temporal encoding method to further enhance rendering quality. Specifically, each sequence t is mapped to a temporal embedding $\ell_t = \{\sin(\pi t), \cos(\pi t)\} \in \mathbb{R}^2$, which improves the representational capacity of all MLPs. For example, the color $\{\mathrm{c}\}$ is predicted using an MLP conditioned on both spatial and temporal features:

$$\{c_0, ..., c_{k-1}\} = F_c(\hat{f}_v, \delta_{vc}, \vec{d}_{vc}, \ell_t). \tag{8}$$

Similarly, opacity $\{o\}$, rotation $\{q\}$, and scale $\{s\}$ are each predicted by their individual MLPs.

c) Visual Feature: The inclusion of high-dimensional visual features significantly expands the Gaussian optimization space, leading to higher memory consumption and reduced computational efficiency. Inspired by [28], [42], anchor features are employed to decode low-dimensional Gaussian visual attributes $\{f\} \in \mathbb{R}^{k \times N_l}$ via an MLP F_d:

$$\{f_0, ..., f_{k-1}\} = F_d(\hat{f}_v, \delta_{vc}, \vec{d}_{vc}, \ell_t), \tag{9}$$

Similar to color rendering, the low-dimensional features are rendered through the 3DGS framework, yielding the rendered feature representation $\tilde{F}$, as defined below:

$$\tilde{F} = \sum_{i=1}^{N} f_i \sigma_i \prod_{j=1}^{i-1} (1 - \sigma_j). \tag{10}$$

To align the low-dimensional Gaussian parameters with the high-dimensional N_h visual features, we employ a shallow MLP $\mathcal{F}_m$ to map them into a higher-dimensional space and obtain high-dimensional visual features $\hat{F}$:

$$\hat{F} = \mathcal{F}_m(\tilde{F}) \in \mathbb{R}^{N_h}. \tag{11}$$

We supervise the learning of DINO features F and rendering features $\hat{F}$ through a loss function L_m:

$$L_m = \frac{1}{N_d} \sum_{i=0}^{N_d} \left(1 - \frac{F_i \cdot \hat{F}_i}{\|F_i\|_2 \left\|\hat{F}_i\right\|_2}\right), \tag{12}$$

where N_d is the feature dimension of DINO, i is the i-th vector. Since $N_l << N_h$, visual features $\hat{F}$ are efficiently distilled into the 3DGS representation, preserving optimization efficiency while reducing memory and computational overhead.

d) Uncertainty Prediction for Mapping: Mapping process is extended to include not only the optimization of the static scene representation but also the refinement of the motion mask, thereby enhancing both robustness and rendering quality. Specifically, DINO features are fed into an MLP $\mathcal{F}_u$, which predicts per-pixel uncertainty:

$$\sigma = \mathcal{F}_u(F), \tag{13}$$

where the parameters of $\mathcal{F}_u$ are optimized under the supervision of the loss function L_u. The uncertainty map is then binarized to generate a motion mask $M(\sigma) = \delta(\tau\{\sigma\} > 0.9)$, where δ denotes an indicator function, and τ denotes a normalization function.

To ensure the multimodal consistency of $\mathbf{G}$ during optimization in dynamic environments, Eq. 4 is accordingly modified as follows:

$$L = M(L_g + \lambda_4 L_m) + \lambda_5 \bar{s}, \tag{14}$$

where $\bar{s}$ denotes the mean scale, introduced to prevent scale explosion. During each Gaussian optimization iteration, the uncertainty MLP $\mathcal{F}_u$ is optimized simultaneously.

D. Tracking

Previous methods such as [12], [26], [33] follow a sequential tracking-mapping pipeline, in which camera pose estimation is followed by scene representation optimization, and dynamic object recognition is typically performed after map convergence. This tight coupling limits real-time localization in dynamic environments.

Therefore, to achieve a parallel tracking and mapping framework, a training-free estimator is proposed to decouple motion mask generation from global mapping optimization. By exploiting the fast rendering capabilities of 3DGS, multimodal residuals R are computed and fed in real time into the following cost function ξ, from which an uncertainty map $\sigma_t \in \mathbb{R}^{H \times W}$ is optimized:

$$\xi(\sigma_t) = \arg\min_{\sigma_t} \frac{1}{HW} \left\{ \sum_{i=1}^{H} \sum_{j=1}^{W} \frac{1}{2} \left(\frac{R_{ij}}{\sigma_t^2} + \log \sigma_t \right) \right\} \tag{15}$$

Since tracking emphasizes real-time performance, prolonged iterative refinement of the motion mask is impractical. In addition, the presence of non-rigid dynamic objects may lead to the inclusion of potentially dynamic feature points as landmarks, thereby degrading long-term localization. WildGS-SLAM [39] addresses this issue by

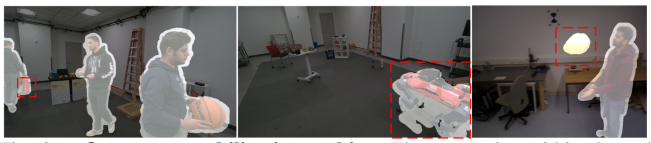

Fig. 2. **Open-set capability in tracking.** The categories within the red box are all without pre-training, and the bright regions indicate the detected dynamic objects.

resetting the map and pose during initialization to mitigate dynamic interference. However, such a strategy is more appropriate for offline 3D reconstruction than for robotics applications. In contrast, our method employs the YOLO-based [43] segmentation result M_{YOLO}, which includes only segmentation without semantic information, to cross-validate and expand the motion mask, mitigating initialization errors and non-rigid motion issues. We generate the motion mask by integrating intersection-over-union (IoU) and residual cues, formulated as $M_t = IoU(M(\sigma_t), M_{\text{YOLO}}) > 0.1$, which enables a more robust and comprehensive exclusion of dynamic regions. During pose estimation, feature points within M_t are excluded to achieve precise localization. Although YOLO is trained on a closed set, our residual-guided refinement leverages residual cues rather than semantic labels, enabling UP-SLAM to generalize to unseen dynamic objects (Fig. 2).

IV. Experiments

A. Experimental Setup

To demonstrate the competitiveness of our approach, we compare it against 13 methods, categorized as follows: (a) Classic SLAM methods: ORB-SLAM3 [1]; (b) Classic dynamic SLAM methods: DynaSLAM [5], ReFusion [44]; (c) NeRF-based SLAM methods: NICE-SLAM [21]; (d) NeRF-based dynamic SLAM: RoDyn-SLAM [25]; (e) 3DGS-based SLAM: Photo-SLAM [32], GS-SLAM [30], SplaTAM [31], RTG-SLAM [45], MonoGS [46]; (f) 3DGS-based dynamic SLAM methods: DG-SLAM [33], Gassidy [34], WildGS-SLAM [12]. In WildGS-SLAM*, the function of continuing pose optimization after stopping image input is disabled, while the back-end scene optimization remains enabled, making the system more suitable for robotics applications. All methods are evaluated on dynamic datasets (TUM RGB-D [47], Bonn RGB-D [48], MoCap RGB-D [12]) and the static ScanNet dataset [49]. UP-SLAM is fully implemented in C++ and CUDA, and runs on a desktop equipped with Intel i7-12700KF and an NVIDIA RTX 4060ti 16G GPU. We set the loss weight: $\{\lambda, \lambda_1, \lambda_2, \lambda_3, \lambda_4, \lambda_5\}$=$\{0.8, 0.6, 1.0, 0.4, 0.01\}$, the residuals weight is $\{\lambda_1^t, \lambda_2^t, \lambda_3^t\}$=$\{0.25, 0.7, 0.1\}$, the refinement iteration count: 20000, and low-dimensional $N_l = 16$. **Bold** is the best result, and underline is the second best result. We report original results for non-open-source methods and average results over 5 runs on the same device for open-source methods.

TABLE I

Tracking results on Bonn RGB-D dataset. "X" denotes a tracking failure. (ATE RMSE↓[cm])

Method	Ball.	Ball.2	Ball_t	Ps_t	Ps_t2	Mv_b2	Avg.
ORB-SLAM3	5.8	17.7	3.1	70.7	77.9	3.5	29.78
NICE-SLAM	X	66.8	21.2	54.9	45.3	31.9	-
Photo-SLAM	6.9	26	3.2	76.4	87.4	3.6	33.91
GS-SLAM	37.5	26.8	31.9	46.8	50.4	4.8	33.03
DynaSLAM	3.0	2.9	4.9	6.1	7.8	3.9	4.76
RoDyn-SLAM	7.9	11.5	13.3	14.5	13.8	12.6	12.26
GassiDy	**2.6**	7.6	-	10.3	13	5.4	-
DG-SLAM	3.7	4.1	10	4.5	6.9	3.5	5.45
WildGS-SLAM*	2.8	2.8	3.6	4.3	4.1	3.4	3.5
UP-SLAM	2.8	**2.7**	**2.9**	**4.0**	**3.6**	**3.2**	**3.2**

TABLE II

Tracking results on ScanNet dataset. OOM means the 16GB GPU ran out of memory. (ATE RMSE↓ [cm])

Method	00	59	106	169	182	207	Avg.
MonoGS	9.8	32.1	8.9	10.7	21.74	7.9	15.19
SplaTAM	12.8	10.1	17.7	12.1	**9.27**	7.5	11.57
Photo-SLAM	8.41	7.82	9.64	9.31	10.87	7.34	8.89
RTG-SLAM	8.6	7.73	9.23	8.73	11.75	7.8	8.97
DG-SLAM	**7.9**	11.5	8.0	**8.3**	14.56	8.2	9.74
WildGS-SLAM*	OOM	7.6	9.3	8.4	10.9	**6.2**	-
UP-SLAM	8.2	**7.3**	**7.9**	8.8	12.31	7.0	**8.58**

B. Evaluation of Tracking Performance

a) Static Scenes: UP-SLAM is evaluated on the public static ScanNet [49] dataset (Table II) and the TUM-RGBD *Fr3/office* (Fr3/of) sequence (Table III) to assess its robustness. While dynamic object recognition is utilized to improve the robustness of SLAM systems in dynamic environments, inaccurate recognition can adversely affect localization accuracy in static scenes. As shown in Table II, our approach achieves **11.9**% improvement on average compared to DG-SLAM, which is also designed for dynamic scenes. WildGS-SLAM encounters a memory overflow of 16GB on sequence *00*, causing failure.

b) Dynamic Scenes: As shown in Table IV, our method improves localization accuracy by an average of 84.7% compared with DG-SLAM. This is because DG-SLAM achieves open-set capability based on historical geometric information, which makes it less robust in complex dynamic environments. While DynaSLAM performs well in Table III due to its predefined dynamic object handling strategy, it exhibits noticeable drift in Tables I, IV. This degradation arises from the presence of numerous dynamic objects that are difficult to predefine in those datasets, especially in the *Table2* and the *Umbrella* sequences. In Table III, the sequences include both static (Fr3/of) and dynamic scenes. On dynamic sequences, DG-SLAM and WildGS-SLAM achieve ATEs of 1.68 and 1.66, respectively, whereas our method achieves 1.42, corresponding to a **14.5**% improvement over

TABLE III

Tracking results on TUM RGB-D dataset. "X" denotes a tracking failure. (ATE RMSE↓ [cm])

Method	Fr3/w/x	Fr3/w/hf	Fr3/w/st	Fr3/s/x	Fr2/dp	Fr3/of	Avg.
ORB-SLAM3	28.1	30.5	2.0	1.0	1.5	1.0	10.68
NICE-SLAM	113.8	X	88.2	7.9	X	6.9	-
Photo-SLAM	60.4	35.7	13.7	1.0	**0.6**	1.0	18.73
DynaSLAM	1.5	2.9	0.7	1.6	0.9	1.1	1.45
RoDyn-SLAM	8.3	5.6	1.7	5.1	5.6	2.8	4.85
DG-SLAM	1.7	1.8	0.7	1.0	3.2	2.3	1.78
WildGS-SLAM*	**1.4**	**1.6**	**0.4**	1.1	3.8	12.7	3.5
UP-SLAM	1.6	2.6	0.7	**0.9**	1.3	**1.0**	**1.35**

TABLE IV

TRACKING RESULTS ON MOCAP RGB-D DATASET. "X" DENOTES A TRACKING FAILURE. (ATE RMSE↓[CM])

Method	ANYmal1	ANYmal2	Ball	Crowd	Person	Racket	Stones	Table1	Table2	Umbrella	Avg.
NICE-SLAM	X	123.6	21.1	X	150	X	134	138	X	23.8	-
ReFusion	4.2	5.6	5.0	91.9	5.0	10.4	39.4	99.1	101	10.7	37.23
Photo-SLAM	79.5	11.8	50.3	105	27.5	38.2	113	39	64	84	61.46
DynaSLAM	1.6	**0.5**	0.5	1.7	**0.5**	0.8	2.1	1.2	34.8	34.7	7.84
DG-SLAM	1.2	2.1	0.8	1.3	1.5	1.6	1.5	2	57.9	1.3	7.06
WildGS-SLAM*	0.5	1.4	**0.3**	**0.4**	2.8	**0.6**	2.4	1.5	5.7	**0.5**	1.61
UP-SLAM	**0.4**	0.6	0.6	1.1	1.1	0.9	**1.0**	**0.7**	**3.6**	0.8	**1.08**

TABLE V

RENDERING PERFORMANCE COMPARISON OF SLAM METHODS ON BONN RGB-D DATASET.

Seq.	Metric	Ball.	Ball.2	Ball_t	Ps_t	Ps_t2	M_b2	Avg.
SplaTAM	PSNR↑	20.55	18.74	20.44	17.41	16.27	22.43	19.30
	SSIM↑	0.829	0.756	0.819	0.438	0.625	0.881	0.724
	LPIPS↓	0.184	0.247	0.207	0.307	0.339	0.158	0.240
Photo-SLAM	PSNR↑	20.82	22.80	25.31	22.63	23.72	25.60	23.48
	SSIM↑	0.814	0.830	0.833	0.803	0.814	0.859	0.825
	LPIPS↓	0.210	0.175	0.183	0.272	0.254	0.159	0.208
DG-SLAM	PSNR↑	17.15	16.32	16.63	18.62	17.60	18.48	17.46
	SSIM↑	0.779	0.752	0.672	0.748	0.715	0.805	0.745
	LPIPS↓	0.393	0.396	0.535	0.506	0.540	0.415	0.464
WildGS-SLAM	PSNR↑	25.02	24.24	22.33	22.93	22.82	23.25	23.43
	SSIM↑	0.961	0.950	0.929	0.941	0.946	0.921	0.941
	LPIPS↓	0.143	0.154	0.212	0.198	0.163	0.245	0.185
UP-SLAM	PSNR↑	**29.31**	**28.03**	**27.58**	**27.98**	**27.47**	**27.67**	**28.0**
	SSIM↑	**0.965**	**0.967**	**0.946**	**0.952**	**0.956**	**0.955**	**0.956**
	LPIPS↓	**0.089**	**0.100**	**0.144**	**0.128**	**0.118**	**0.128**	**0.117**

TABLE VII

RENDERING PERFORMANCE COMPARISON OF SLAM METHODS ON TUM RGB-D DATASET.

Seq.	Metric	Fr3/w/x	Fr3/w/hf	Fr3/w/st	Fr3/s/x	Fr2/dp	Fr3/of	Avg.
Spla TAM	PSNR↑	18.83	16.82	22.30	24.10	20.89	21.90	20.8
	SSIM↑	0.771	0.688	0.874	0.922	0.845	0.82	0.83
	LPIPS↓	0.252	0.328	0.156	0.123	0.233	0.218	0.215
Photo-SLAM	PSNR↑	15.80	16.84	17.79	22.24	18.7	22.74	19.01
	SSIM↑	0.591	0.636	0.724	0.635	0.859	0.78	0.7
	LPIPS↓	0.369	0.358	0.192	0.238	0.159	0.154	0.283
DG-SLAM	PSNR↑	12.36	12.87	11.71	11.43	16.72	16.14	13.53
	SSIM↑	0.553	0.581	0.549	0.565	0.695	0.613	0.593
	LPIPS↓	0.559	0.518	0.515	0.519	0.476	0.547	0.522
WildGS -SLAM	PSNR↑	18.93	17.36	21.75	22.14	20.30	21.4	20.31
	SSIM↑	0.889	0.827	0.940	0.937	0.822	0.837	0.875
	LPIPS↓	0.19	0.255	0.104	0.117	0.212	0.275	0.192
UP-SLAM	PSNR↑	**26.38**	**22.27**	**27.11**	**28.29**	**23.47**	**23.50**	**25.17**
	SSIM↑	**0.957**	**0.896**	**0.963**	**0.969**	**0.905**	**0.904**	**0.932**
	LPIPS↓	**0.098**	**0.163**	**0.074**	**0.080**	**0.238**	**0.226**	**0.147**

WildGS-SLAM. Across the combined sequences, our method attains a **61.4**% improvement over WildGS-SLAM.

C. Evaluation of Rendering Performance

As dynamic objects are limited in the TUM and Bonn datasets, we pre-obtain their masks to evaluate rendering quality. For the MoCap dataset, where many dynamic objects cannot be predefined, we provide visualization results in Fig. 4.

As reported in Tables V and VII, our method achieves a notable improvement in rendering quality, with an average PSNR gain of **4.72** dB. Additionally, the absence of a robust Gaussian primitive initialization strategy in DG-SLAM leads to incomplete reconstructions, significantly degrading rendering quality. Our method also achieves superior rendering quality on the static *Fr3/of* in Table VII.

Figs. 3 and 4 provide a visual comparison of the rendered results. The two static SLAM methods, SplaTAM and Photo-SLAM, fail to generate a static map. Both DG-SLAM and WildGS-SLAM exhibit varying degrees of failure. In contrast, UP-SLAM effectively removes dynamic objects and constructs a high-fidelity, artifact-free static map.

TABLE VI

ABLATION STUDY ON BONN RGB-D DATASET. RESULTS ARE AVERAGED OVER SIX SEQUENCES.

	ATE↓	PSNR↑	model size↓	Similarity↑
w/o Temp.	3.37	26.6	7.04	78.6
w/o Seg.	3.46	27.1	7.03	78.5
w/o Prob.	3.57	27.74	22.92	79.2
UP-SLAM(Our)	**3.2**	**28**	**7.01**	**79.5**

D. Ablation Study

An ablation study is conducted to assess the contributions of each component, as presented in Table VI. The temporal encoding primarily enhances rendering quality, yielding an average PSNR improvement of **1.4** dB.

During initialization, non-rigid objects can cause dynamic keypoints to be misclassified as static landmarks. Incorporating YOLO helps mitigate this, but our method remains effective without it, demonstrating that the YOLO-based module improves performance but is not essential, further highlighting the robustness of our approach.

The probabilistic anchor update module significantly reduces model size, improving its suitability for deployment on embedded platforms. Without anchor updates, Gaussian primitives cannot be effectively pruned, leading to slower map updates and weakened residual feedback to the tracking thread, ultimately degrading pose estimation accuracy.

Moreover, UP-SLAM improves similarity scores to nearly **80**%, demonstrating its potential for downstream applications such as object-level navigation and semantic understanding.

E. Runtime Analysis

Table VIII presents the runtime analysis. By decoupling motion mask generation from map optimization, our system

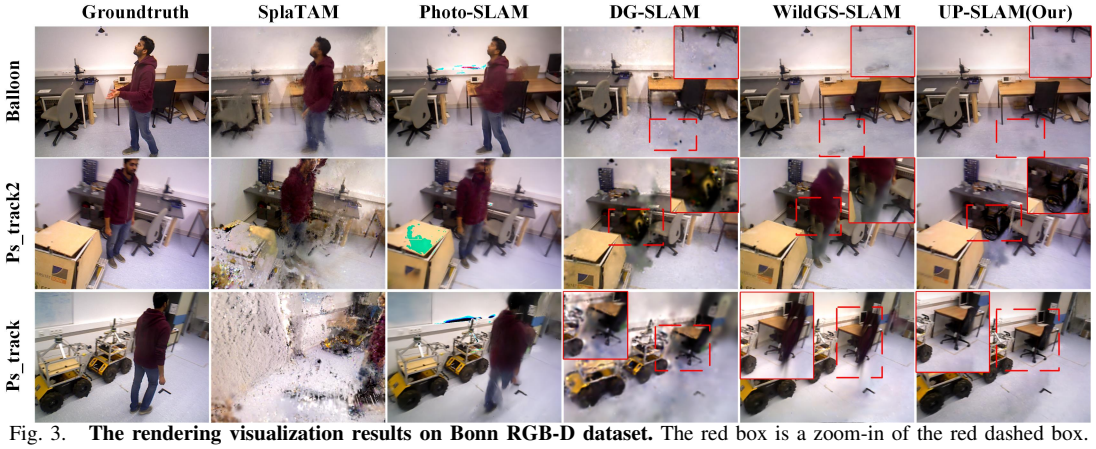

Fig. 3. **The rendering visualization results on Bonn RGB-D dataset.** The red box is a zoom-in of the red dashed box.

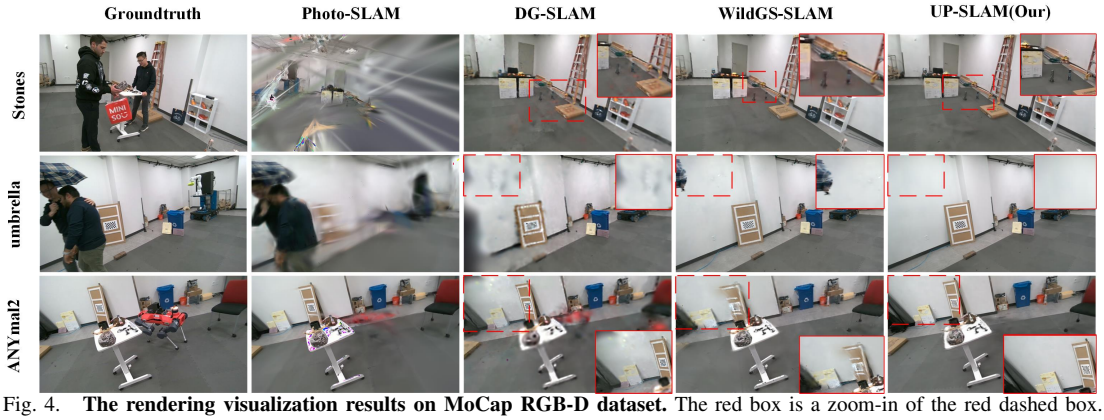

Fig. 4. **The rendering visualization results on MoCap RGB-D dataset.** The red box is a zoom-in of the red dashed box.

TABLE VIII

ADDITIONAL EXPERIMENTS ON BONN BALLOON SEQUENCE. "-" INDICATES THE ABSENCE OF THIS COMPONENT.

	Avg. /frame↓[ms]	Total Time (+refine)↓[s]	Model Size↓[MB]
SplatTAM	4046	1776.54(+0)	29.9
WildGS-SLAM	1838	1526.584(+719.61)	8.8
DG-SLAM	1011	**444.1684**(+0)	-
UP-SLAM(Our)	**78**	694.814(+660.309)	**4.9**

achieves a parallel tracking and mapping architecture. This enables a processing rate of **12** Hz tracking, meeting the real-time localization requirements for robotics. Compared to WildGS-SLAM, we use the same number of refinement iterations but achieve a 2× speed-up. SplaTAM and DG-SLAM do not perform refinement, so no additional time is required. DG-SLAM excludes segmentation time from its reported runtime. In contrast, our evaluation incorporates both segmentation and visual feature extraction. Although DG-SLAM achieves the fastest runtime, our approach attains a more favorable balance between reconstruction quality and localization speed. This trade-off is justified, as mapping is generally less constrained by real-time requirements than tracking. Additionally, the use of probabilistic anchor updates and MLP-based Gaussian attribute encoding significantly reduces the overall model size.

V. CONCLUSIONS

We presented UP-SLAM, a real-time RGB-D SLAM system that parallelizes tracking and mapping, adaptively manages Gaussian primitives, and robustly filters dynamic regions via training-free uncertainty estimation. With enhanced rendering and uncertainty prediction through temporal encoding and DINO features, UP-SLAM achieves superior localization accuracy and map quality while maintaining real-time performance in dynamic environments. In future work, we plan to extend our method to large-scale scenes and apply it to embodied AI.

REFERENCES

[1] C. Campos, R. Elvira, J. J. G. Rodríguez, J. M. Montiel, and J. D. Tardós, "Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam," *IEEE transactions on robotics*, vol. 37, no. 6, pp. 1874–1890, 2021.

[2] R. Wang, M. Schworer, and D. Cremers, "Stereo dso: Large-scale direct sparse visual odometry with stereo cameras," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 3903–3911.

[3] R. Gomez-Ojeda, F.-A. Moreno, D. Zuniga-Noël, D. Scaramuzza, and J. Gonzalez-Jimenez, "Pl-slam: A stereo slam system through the combination of points and line segments," *IEEE Transactions on Robotics*, vol. 35, no. 3, pp. 734–746, 2019.

[4] K. Xu, Y. Hao, S. Yuan, C. Wang, and L. Xie, "Airslam: An efficient and illumination-robust point-line visual slam system," *IEEE Transactions on Robotics*, 2025.

[5] B. Bescos, J. M. Fácil, J. Civera, and J. Neira, "Dynaslam: Tracking, mapping, and inpainting in dynamic scenes," *IEEE robotics and automation letters*, vol. 3, no. 4, pp. 4076–4083, 2018.

[6] Z. Zheng, S. Lin, and C. Yang, "Rld-slam: A robust lightweight vi-slam for dynamic environments leveraging semantics and motion information," *IEEE Transactions on Industrial Electronics*, 2024.

[7] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "Nerf: Representing scenes as neural radiance fields for view synthesis," *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, 2021.

[8] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3d gaussian splatting for real-time radiance field rendering." *ACM Trans. Graph.*, vol. 42, no. 4, pp. 139–1, 2023.

[9] W. Ren, Z. Zhu, B. Sun, J. Chen, M. Pollefeys, and S. Peng, "Nerf on-the-go: Exploiting uncertainty for distractor-free nerfs in the wild," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 8931–8940.

[10] J. Kulhanek, S. Peng, Z. Kukelova, M. Pollefeys, and T. Sattler, "Wildgaussians: 3d gaussian splatting in the wild," *arXiv preprint arXiv:2407.08447*, 2024.

[11] A. Markin, V. Pryadilshchikov, A. Komarichev, R. Rakhimov, P. Wonka, and E. Burnaev, "T-3dgs: Removing transient objects for 3d scene reconstruction," *arXiv preprint arXiv:2412.00155*, 2024.

[12] J. Zheng, Z. Zhu, V. Bieri, M. Pollefeys, S. Peng, and I. Armeni, "Wildgs-slam: Monocular gaussian splatting slam in dynamic environments," *arXiv preprint arXiv:2504.03886*, 2025.

[13] W. Wu, L. Guo, H. Gao, Z. You, Y. Liu, and Z. Chen, "Yolo-slam: A semantic slam system towards dynamic environment with geometric constraint," *Neural Computing and Applications*, pp. 1–16, 2022.

[14] C. Yu, Z. Liu, X.-J. Liu, F. Xie, Y. Yang, Q. Wei, and Q. Fei, "Ds-slam: A semantic visual slam towards dynamic environments," in *2018 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 2018, pp. 1168–1174.

[15] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 2961–2969.

[16] W. Dai, Y. Zhang, P. Li, Z. Fang, and S. Scherer, "RGB-D slam in dynamic environments using point correlations," *IEEE transactions on pattern analysis and machine intelligence*, vol. 44, no. 1, pp. 373–389, 2020.

[17] S. Li and D. Lee, "Rgb-d slam in dynamic environments using static point weighting," *IEEE Robotics and Automation Letters*, vol. 2, no. 4, pp. 2263–2270, 2017.

[18] X. Yu, W. Zheng, and L. Ou, "Cpr-slam: Rgb-d slam in dynamic environment using sub-point cloud correlations," *Robotica*, vol. 42, no. 7, pp. 2367–2387, 2024.

[19] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, *et al.*, "Dinov2: Learning robust visual features without supervision," *arXiv preprint arXiv:2304.07193*, 2023.

[20] E. Sucar, S. Liu, J. Ortiz, and A. J. Davison, "iMAP: Implicit mapping and positioning in real-time," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2021, pp. 6229–6238.

[21] Z. Zhu, S. Peng, V. Larsson, W. Xu, H. Bao, Z. Cui, M. R. Oswald, and M. Pollefeys, "Nice-slam: Neural implicit scalable encoding for slam," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2022, pp. 12 786–12 796.

[22] X. Yang, H. Li, H. Zhai, Y. Ming, Y. Liu, and G. Zhang, "Vox-fusion: Dense tracking and mapping with voxel-based neural implicit representation," in *2022 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*. IEEE, 2022, pp. 499–507.

[23] H. Wang, J. Wang, and L. Agapito, "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*, 2023, pp. 13 293–13 302.

[24] M. M. Johari, C. Carta, and F. Fleuret, "Eslam: Efficient dense slam system based on hybrid representation of signed distance fields," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 17 408–17 419.

[25] H. Jiang, Y. Xu, K. Li, J. Feng, and L. Zhang, "Rodyn-slam: Robust dynamic dense rgb-d slam with neural radiance fields," *IEEE Robotics and Automation Letters*, 2024.

[26] B. Li, Z. Yan, D. Wu, H. Jiang, and H. Zha, "Learn to memorize and to forget: A continual learning perspective of dynamic slam," in *European Conference on Computer Vision*. Springer, 2024, pp. 41–57.

[27] W. Zheng, X. Yu, J. Rong, L. Ou, Y. Wei, and L. Zhou, "Gsorb-slam: Gaussian splatting slam benefits from orb features and transmittance information," *IEEE Robotics and Automation Letters*, vol. 10, no. 9, pp. 9400–9407, 2025.

[28] L. Li, L. Zhang, Z. Wang, and Y. Shen, "Gs3lam: Gaussian semantic splatting slam," in *Proceedings of the 32nd ACM International Conference on Multimedia*, 2024, pp. 3019–3027.

[29] V. Yugay, Y. Li, T. Gevers, and M. R. Oswald, "Gaussian-slam: Photo-realistic dense slam with gaussian splatting," *arXiv preprint arXiv:2312.10070*, 2023.

[30] C. Yan, D. Qu, D. Xu, B. Zhao, Z. Wang, D. Wang, and X. Li, "Gs-slam: Dense visual slam with 3d gaussian splatting," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 19 595–19 604.

[31] N. Keetha, J. Karhade, K. M. Jatavallabhula, G. Yang, S. Scherer, D. Ramanan, and J. Luiten, "Splatam: Splat track & map 3d gaussians for dense rgb-d slam," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 21 357–21 366.

[32] H. Huang, L. Li, H. Cheng, and S.-K. Yeung, "Photo-slam: Real-time simultaneous localization and photorealistic mapping for monocular stereo and rgb-d cameras," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 21 584–21 593.

[33] Y. Xu, H. Jiang, Z. Xiao, J. Feng, and L. Zhang, "Dg-slam: Robust dynamic gaussian splatting slam with hybrid pose optimization," *arXiv preprint arXiv:2411.08373*, 2024.

[34] L. Wen, S. Li, Y. Zhang, Y. Huang, J. Lin, F. Pan, Z. Bing, and A. Knoll, "Gassidy: Gaussian splatting slam in dynamic environments," *arXiv preprint arXiv:2411.15476*, 2024.

[35] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.

[36] R. Martin-Brualla, N. Radwan, M. S. Sajjadi, J. T. Barron, A. Dosovitskiy, and D. Duckworth, "Nerf in the wild: Neural radiance fields for unconstrained photo collections," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 7210–7219.

[37] T. Lu, M. Yu, L. Xu, Y. Xiangli, L. Wang, D. Lin, and B. Dai, "Scaffold-gs: Structured 3d gaussians for view-adaptive rendering," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 20 654–20 664.

[38] A. Hornung, K. M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, "Octomap: An efficient probabilistic 3d mapping framework based on octrees," *Autonomous robots*, vol. 34, pp. 189–206, 2013.

[39] J. Xu, Y. Mei, and V. Patel, "Wild-gs: Real-time novel view synthesis from unconstrained photo collections," *Advances in Neural Information Processing Systems*, vol. 37, pp. 103 334–103 355, 2024.

[40] T. Wen, Z. Liu, B. Lu, and Y. Fang, "Scaffold-slam: Structured 3d gaussians for simultaneous localization and photorealistic mapping," *arXiv preprint arXiv:2501.05242*, 2025.

[41] J. Chen, Y. Yin, T. Birdal, B. Chen, L. J. Guibas, and H. Wang, "Projective manifold gradient layer for deep rotation regression," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 6646–6655.

[42] R.-Z. Qiu, G. Yang, W. Zeng, and X. Wang, "Feature splatting: Language-driven physics-based scene synthesis and editing," *arXiv preprint arXiv:2404.01223*, 2024.

[43] G. Jocher, A. Chaurasia, and J. Qiu, "Ultralytics yolov8," 2023. [Online]. Available: https://github.com/ultralytics/ultralytics

[44] E. Palazzolo, J. Behley, P. Lottes, P. Giguere, and C. Stachniss, "Refusion: 3d reconstruction in dynamic environments for rgb-d cameras exploiting residuals," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2019, pp. 7855–7862.

[45] Z. Peng, T. Shao, Y. Liu, J. Zhou, Y. Yang, J. Wang, and K. Zhou, "Rtg-slam: Real-time 3d reconstruction at scale using gaussian splatting," in *ACM SIGGRAPH 2024 Conference Papers*, 2024, pp. 1–11.

[46] H. Matsuki, R. Murai, P. H. Kelly, and A. J. Davison, "Gaussian splatting slam," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 18 039–18 048.

[47] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers, "A benchmark for the evaluation of rgb-d slam systems," in *2012 IEEE/RSJ international conference on intelligent robots and systems*. IEEE, 2012, pp. 573–580.

[48] E. Palazzolo, J. Behley, P. Lottes, P. Giguère, and C. Stachniss, "ReFusion: 3D Reconstruction in Dynamic Environments for RGB-D Cameras Exploiting Residuals," in *iros*, 2019. [Online]. Available: https://www.ipb.uni-bonn.de/pdfs/palazzolo2019iros.pdf

[49] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner, "Scannet: Richly-annotated 3d reconstructions of indoor scenes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 5828–5839.