CoMA-SLAM: Collaborative Multi-Agent Gaussian SLAM with Geometric Consistency

Lin Chen[1*], Yongxin Su[2*], Jvboxi Wang[1], Pengcheng Han[1†], Zhenyu Xia[1], Shuhui Bu[1], Kun Li[1], Boni Hu[1], Shengqi Meng[1], Guangming Wang[3†]

[1]Northwestern Polytechnical University
[2] Beihang University
[3] Cambridge University
{npuchenlin,wangjv,hanpc1125,xiazhenyu,likunkelly,bushuhui,huboni,mengshengqi}@mail.nwpu.edu.cn,
zy2302321@buaa.edu.cn,gw462@cam.ac.uk

Abstract

Although Gaussian scene representation has achieved remarkable success in tracking and mapping, most existing methods are confined to single-agent systems. Current multi-agent solutions typically rely on centralized architectures, which struggle to account for communication bandwidth constraints. Furthermore, the inherent depth ambiguity of 3D Gaussian splatting poses notable challenges in maintaining geometric consistency. To address these challenges, we introduce CoMA-SLAM, the first distributed multi-agent Gaussian SLAM framework. By leveraging 2D Gaussian surfels and robust initialization strategy, CoMA-SLAM enhances tracking accuracy and geometry consistency. It efficiently manages communication bandwidth while dynamically scaling with the number of agents. Through the integration of intra- and inter-loop closure, distributed keyframe optimization and submap centric update, our framework ensures global consistency and robustly alignment. Synthetic and real-world experiments demonstrate that CoMA-SLAM outperforms state-of-the-art methods in pose accuracy, rendering fidelity, and geometric consistency while maintaining competitive efficiency across distributed multi-agent systems. Notably, by avoiding data transmission to a centralized server, our method reduces communication bandwidth by 99.8% compared to centralized approaches.

Code — https://github.com/npu-chenlin/CoMA-SLAM

1 Introduction

Visual Simultaneous Localization and Mapping (SLAM) provides agents like robots and autonomous vehicles with the ability to perceive and navigate their surroundings. Recent advances have pushed SLAM towards denser representations capable of Novel View Synthesis (NVS) (Matsuki et al. 2024; Keetha et al. 2024), opening up applications in AR/VR, digital twins, and immersive telepresence. However, the pursuit of high-fidelity NVS introduces trade-offs: while neural implicit representations (Mildenhall et al. 2020; Zhu et al. 2022) enable realistic view synthesis, their computationally intensive optimization hinders real-time performance; meanwhile, though 3D Gaussian Splatting (3DGS) (Kerbl et al. 2023)-based methods (Keetha et al. 2024; Matsuki et al. 2024; Chen et al. 2025) improve rendering speed, they often struggle with geometric distortions and inconsistencies under novel viewpoints.

These limitations are amplified in multi-agent scenarios. While distributing the mapping task across multiple agents promises faster environment coverage (Tian et al. 2022) and collaborative localization, maintaining global consistency and merging maps accurately remain significant hurdles. CP-SLAM (Hu et al. 2023) utilizes neural representations, inheriting their drawbacks: slow speed, poor rendering of real-world data, and limited scalability. MAGiC-SLAM (Yugay, Gevers, and Oswald 2024) is a Gaussian-based multi-agent system, but it is centralized and suffers from geometric distortions and inconsistencies. Moreover, both methods face significant communication and hardware limitations: centralized systems require high-bandwidth communication and rely on powerful servers, limiting scalability and flexibility in multi-agent deployments.

To address these intertwined challenges, we propose CoMA-SLAM, a fully distributed multi-agent Gaussian SLAM that combines the geometric strengths of 2D Gaussian Surfels (Huang et al. 2024a) with a robust multi-agent framework featuring global consistency mechanisms (see Fig. 1). Our key insight is that improving the geometric accuracy and view consistency within each agent's local SLAM process is fundamental to achieving accurate multi-agent tracking and high-fidelity global map reconstruction. By unifying distributed optimization, efficient geometry representation, and adaptive memory management, CoMA-SLAM sets a new benchmark for collaborative mapping.

- The first distributed multi-agent Gaussian SLAM system is proposed, achieving geometric accuracy, tracking robustness, and minimal communication overhead.
- A novel 2D Gaussian Surfels-based Tracking and Mapping framework is proposed. It introduces an adaptive growth strategy and a surface-fitting initialization, ensuring high-precision tracking and high-fidelity novel view

*These authors contributed equally.
†Corresponding author

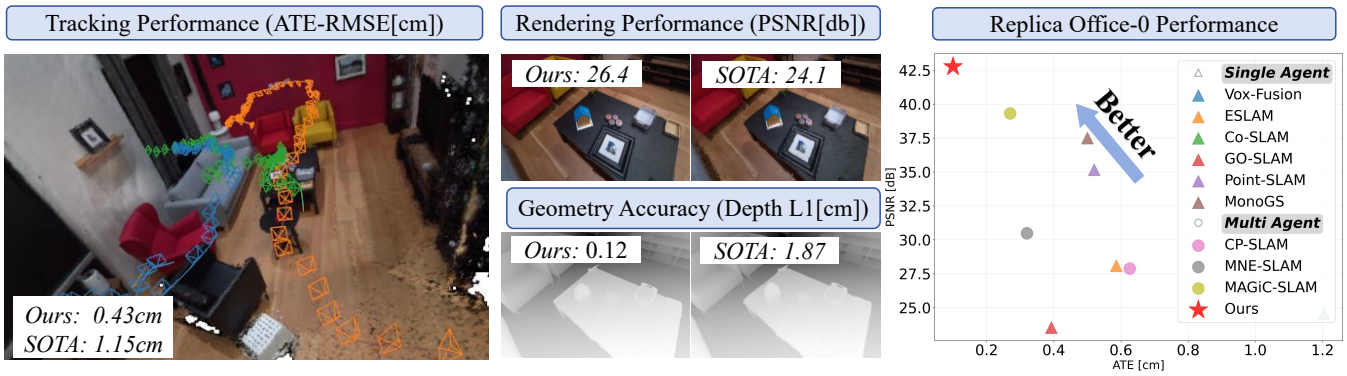

Figure 1: CoMA-SLAM is the first distributed multi-agent Gaussian SLAM system. Our system takes RGB-D input streams from multiple simultaneously operating agents, estimates their trajectories and reconstructs a 2D Surfel map that enables novel view synthesis. We present the high-fidelity 2D Surfel map of a real-world environment alongside multiple agent trajectories within it. Our method effectively utilizes information from multiple agents to achieve millimeters-level tracking accuracy. Our method allows for realistic rendering of color and depth, significantly improving the state-of-the-art method. Unlike previous methods, CoMA-SLAM is flexible in the number of agents it can handle.

synthesis with reduced memory footprint.

- A Distributed Pose Graph Optimization (DPGO) framework enabling online global consistency through relative-to-global optimization, achieving efficient pose optimization.
- Demonstrated state-of-the-art performance in tracking accuracy, rendering quality, and geometric fidelity on challenging multi-agent datasets compared to existing single- and multi-agent methods.

2 Related Work

2.1 Gaussian SLAM

3DGS (Kerbl et al. 2023) provides an explicit representation enabling high-fidelity rendering and faster optimization, which has inspired several 3DGS-based SLAM systems (Keetha et al. 2024; Matsuki et al. 2024; Yan et al. 2024; Peng et al. 2024; Huang et al. 2024b). Tracking is performed by differentiating the rendering process w.r.t. camera pose. However, despite their computational efficiency, such methods often struggle with geometric consistency, leading to tracking drift and rendering artifacts—particularly under significant viewpoint changes. To address this, 2D Gaussian Surfels (Huang et al. 2024a) improve geometry in single-agent settings, while concurrent work like LoopSplat (Zhu et al. 2024) extends 3DGS SLAM with loop closure. These advancements highlight the growing need to enhance geometric fidelity in Gaussian-based representations, especially for multi-agent systems. In addition to 2DGS (Huang et al. 2024a), methods like GOF (Yu, Sattler, and Geiger 2024) and RaDe-GS (Zhang et al. 2024) further tackle multiview consistency through Gaussian flattening. Our work bridges this gap by integrating the geometric strengths of 2D Gaussian Surfels with novel rendering techniques into a multi-agent framework, enabling robust loop closure.

2.2 Multi-Agent SLAM

Multi-agent SLAM systems can be centralized (Schmuck and Chli 2019; Lajoie and Beltrame 2023; Yugay, Gevers, and Oswald 2024) or distributed (Tian et al. 2022; Lajoie et al. 2020). Centralized systems typically aggregate data on a server for global optimization, offering potentially higher accuracy, while distributed systems offer robustness to communication constraints. Most existing multi-agent systems like CCM-SLAM (Schmuck and Chli 2019) and SWARM-SLAM (Lajoie and Beltrame 2023) focus on sparse mapping and do not support NVS. CP-SLAM (Hu et al. 2023) was the first NVS-capable multi-agent system but suffers from the limitations of its underlying neural representation and scalability issues, as discussed earlier. MAGiC-SLAM (Yugay, Gevers, and Oswald 2024) adopted a centralized 3D Gaussian design with loop closure, but its architecture limits scalability in practical multi-robot scenarios. Additionally its 3D geometric consistency degrades under viewpoint changes, causing depth artifacts that hurt tracking accuracy. Building on this direction, we present a fully distributed collaboration framework, leveraging the geometrically accurate 2D Gaussian Surfels and our proposed rendering and mapping techniques to achieve higher tracking robustness, improved reconstruction fidelity, and scalable multi-agent coordination.

3 Method

We introduce CoMA-SLAM, the first distributed multi-agent Gaussian SLAM system. It leverages 2D Gaussian Surfel for scene representation (Sec. 3.1), enhancing rendering accuracy through precise ray intersection. As shown in Fig. 2, our system ingests RGB-D streams and camera intrinsics $\{\mathbf{I}_t, \mathbf{D}_t, \mathbf{K}_t\}_{t=1}^{N}$, producing globally consistent trajectories and a high-fidelity Gaussian map $\{\mathbf{traj}_t, \mathcal{G}_t\}_{t=1}^{N}$. Each agent performs tracking and mapping independently and the Gaussian Surfel is carefully initialized and added (Sec. 3.2). The necessary information are periodically transmitted to other agents and the intra-inter loop closure con-

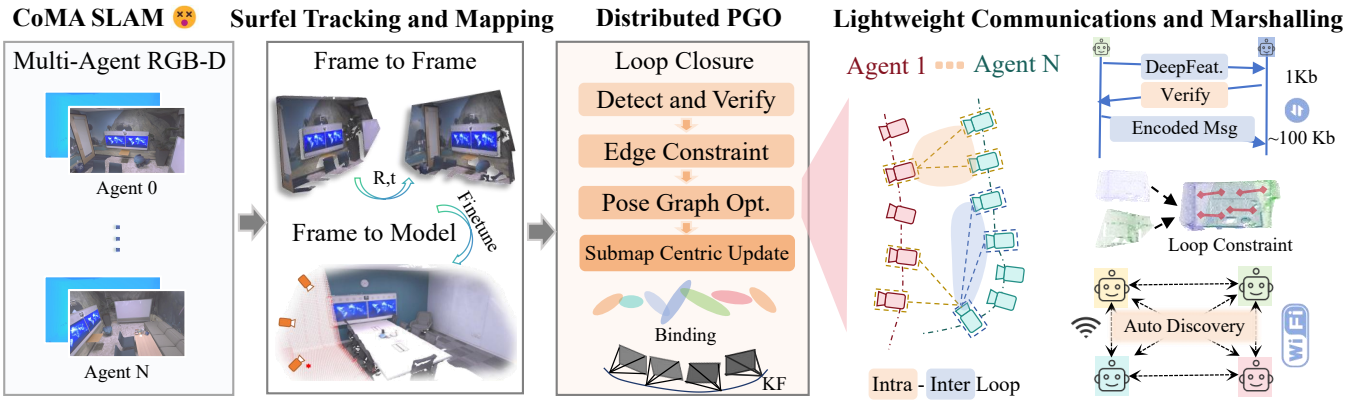

Figure 2: CoMA-SLAM Overview. It is the first distributed multi-agent collaborative Gaussian SLAM system. It leverages Gaussian Surfel as the scene representation, and each agent performs individual tracking and mapping with intra-loop closure. When communication is available, it performs inter loop closure and distributed PGO, and exchanges the image feature and compressed depth information to achieve global consistency. After the PGO, the map is updated by Submap Centric Update.

straint is thoroughly checked to enhance overall localization accuracy (Sec. 3.3). We detail the communication framework and depth compression mechanism in Sec. 3.4.

3.1 Gaussian Scene Representation

The map is represented by 2D Gaussian Surfels $\mathcal{G}$ and a set of cameras $\mathcal{K}$. Unlike 3D Gaussians, these primitives lie on tangent planes to the scene surface, offering better surface modeling and geometric accuracy (Huang et al. 2024a). The surfel $\mathcal{G}_i$ is defined by a center $\mu_i \in \mathbb{R}^3$, a geometry covariance $\mathbf{\Sigma}_i = \mathbf{R}_i\mathbf{S}_i \in \mathbb{R}^{3\times3}$, the third dimension of $\mathbf{S}_i$ is $\mathbf{0}$, an opacity o_i, and a color c_i. The camera $\mathcal{K}_j$ is defined by an intrinsic matrix $\mathbf{K}_j \in \mathbb{R}^{3\times3}$ and a pose $\mathbf{T}_j \in SE(3)$.

$$G = \{\mathcal{G}_i : (\mathbf{\Sigma}_i, \mu_i, o_i, c_i) | i = 1, \cdots, n\}, \quad (1)$$

$$K = \{\mathcal{K}_j : (\mathbf{K}_j, \mathbf{T}_j) | j = 1, \cdots, m\} \quad (2)$$

Given a ray $\mathbf{r}$ and a surfel $\mathcal{G}_i$ on the scene, the intersection point is denoted as p_i and the light absorption is denoted as $\alpha_i = o_i\mathcal{G}_i(p_i)$. The rendering process sequentially blends the surfels along the ray. This results in the rendered color C, depth D, and accumulated opacity A:

$$(C, D, A) = \sum_{i=1}^{n}(c_i, d_i, 1)\alpha_i\Pi_{j=1}^{i-1}(1-\alpha_j) \quad (3)$$

We use the accumulated opacity to normalize the adjusted depth, resulting in the expected depth $D_e = \frac{D}{A}$. It mitigates the underestimation issue caused by small opacity, which is common in the initial stages of mapping. D is used to represent D_e in the following for simplicity.

3.2 Gaussian Surfel Tracking and Mapping

Frame-to-Frame and Frame-to-Model Tracking. Given a colored point cloud P_t lifted from a RGB-D frame I_t, we perform Frame-to-Frame and Frame-to-Model tracking. Frame-to-Frame tracking involves iterative multi-scale dense registration, which includes minimizing geometric and photometric errors. This provides an initial guess for the relative pose $T_{t-1,t} \in SE(3)$. Detailed information can be found in (Park, Zhou, and Koltun 2017).

The initial pose $T_{t-1,t}$ is further refined by Frame-to-Model optimization, minimizing a rendering loss against the current submap G^s. The depth mask M_{depth} excludes pixels where the absolute depth difference exceeds the threshold τ_{depth}, as such pixels typically correspond to occlusions or dynamic objects. The opacity mask M_{opacity} excludes pixels with rendered opacity below the threshold τ_{opacity}, thereby preventing unreliable contributions from unmodeled regions. The tracking loss is formulated as:

$$\arg\min_{T_{t-1,t}} M_{\text{depth}}\, M_{\text{opacity}} \left[\mathcal{L}_1(\hat{I}, I_t) + \lambda_d\, \mathcal{L}_1(\hat{D}, D_t)\right], \quad (4)$$

where $(\hat{I}, \hat{D})$ denotes the color and depth rendered from the submap transformed by $T_{t-1,t}$, and (I_t, D_t) represents the observation. Here, $\mathcal{L}_1(\cdot,\cdot)$ is the L1 loss, λ_d weights the depth term, M_{depth} and M_{opacity} indicate that the loss is only computed for pixels where both masks are active.

Adaptive Gaussian Growing. Most Gaussian SLAM methods, such as SplaTAM (2024) and GS-SLAM (2024), generate Gaussian points at pixels with large depth differences. This easily leads to non-convergence and memory overload when rendered depth is inaccurate. We propose an efficient approach by directly integrating redundancy detection into the rendering pipeline. During alpha-blending, we check for existing Gaussian points near the depth $D(p)$. If none exists, the pixel p is unprojected and added. Additionally, pixels with opacity below τ_{opacity} are also added. This selective growing strategy ensures only necessary Gaussians are incorporated into the map, significantly improving rendering efficiency while maintaining geometric accuracy.

$$D(p) \notin G^s \text{ or } A(p) < \tau_{\text{opacity}} \rightarrow add(p) \quad (5)$$

Surface Fitting Initialization. Existing methods overlook the importance of orientation when initializing Gaussian points, which can lead to misaligned surfaces and degraded depth rendering, ultimately harming tracking accuracy and

efficiency. In contrast, our proposed initialization strategy explicitly determines the initial scale $\mathbf{S}$ and orientation $\mathbf{R}$ from the depth map and the derived local surface normals $\mathbf{n}$.

$$\mathbf{S} = diag(\frac{D(p)}{f_x}, \frac{D(p)}{f_y}, 0), \tag{6}$$

$$\mathbf{n} = \frac{\nabla_x D(p) \times \nabla_y D(p)}{\|\nabla_x D(p) \times \nabla_y D(p)\|}, \tag{7}$$

$$\mathbf{R} = (\mathbf{e}_1, \mathbf{e}_2, \mathbf{n}), \mathbf{e}_1 \perp \mathbf{e}_2 \perp \mathbf{n} \tag{8}$$

This ensures that the Gaussian points are properly placed and oriented to align with the underlying scene, improving the accuracy of rendered depth and, in turn, tracking precision. Moreover, by eliminating the need for additional orientation optimization, our approach achieves more efficient tracking without compromising quality.

Keyframe and Mapping. As the number of Gaussian surfels increases, the rendering efficiency exhibits a decline. To maintain online performance, we employ a submap strategy (Keetha et al. 2024) with adaptive keyframe selection. Each agent maintains a local submap G^s with a maximum size threshold θ_{sub}. When the submap size exceeds θ_{sub}, the current frame is identified as a new keyframe, and the submap is reset with initial Gaussian points from this frame.

The mapping optimization is performed every $N_{\text{opt}} = 5$ frames to maintain map quality. The surfels within the active submap are optimized using keyframes within the submap.

$$\mathcal{L}_{\text{mapping}} = \mathcal{L}_c + \lambda_d \mathcal{L}_1(\hat{D}, D_{gt}), \tag{9}$$

where $\mathcal{L}_c$ is the color loss combining $\mathcal{L}_1$ photometric loss with the SSIM term from (Kerbl et al. 2023).

3.3 Distributed Loop Closure

Recent methods such as CP-SLAM (2023) and MAGiC-SLAM (2024) employ a centralized server to manage global consistency and do not perform loop closure optimization during each agent's tracking process. This approach results in cumulative drift errors. In Fig. 3, we propose Distributed PGO to address this issue. By decomposing global PGO into per-agent submaps, we solve pairwise relative poses between agents, then lift to global poses via pose graph—avoiding centralized batch optimization. Specifically, during runtime, each agent not only performs intra-agent loop closure, but also relies on communication to exchange necessary information for inter-agent loop closure.

Intra-Agent Loop Closure. Leveraging the strong generalization ability of the foundation vision model for place recognition, we employ DinoV2 (Oquab et al. 2023) as the feature extractor and FAISS (Douze et al. 2024) for efficient feature retrieval. The system computes cosine similarity between the current frame and all historical keyframes in the database, identifying potential loop closures where the similarity exceeds threshold θ_{sim}. To ensure robust loop detection, we apply a multi-stage filtering strategy that includes: (1) temporal filtering $|t_i - t_j| > \tau_{\text{min}}$ to avoid detecting loops between temporally adjacent frames; (2) multi-frame verification checks consistency across frame windows around

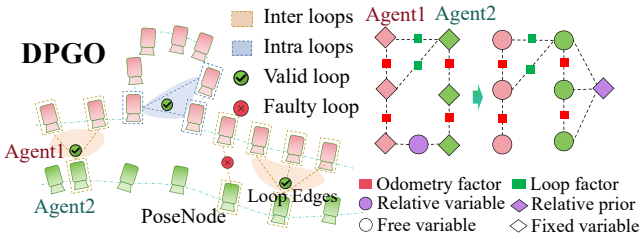

Figure 3: Distributed PGO. The loop is carefully detected to establish intra- and inter-agent constraints. The relative phase optimizes transformations between agents, while the global phase refines global poses of all keyframes using these transformations as initial guesses. ♢ and ◯ denote fixed and free variables during optimization, respectively.

keyframes i and j, accepting loops only if the average similarity across all frame pairs exceeds threshold θ_{multi}. When a loop is detected between frame i and frame j, the constraint relative pose $T_{ij} \in SE(3)$ is estimated. Instead of registering potentially inconsistent surfels means across agents/-time, we register the input colored point clouds (P_i, P_j) associated with the first frame of each loop-candidate submap. We use FPFH features with RANSAC for coarse alignment, followed by geometry and photometric alignment (Park, Zhou, and Koltun 2017). The intra-loop closure constraint is optimized by minimizing the sum of squared Mahalanobis distances of edge errors (Kümmerle et al. 2011):

$$\min_{\{T_k\}} \sum_{\langle i,j \rangle \in \mathcal{E}} e_{ij}^T \Omega_{ij} e_{ij}, \tag{10}$$

$$e_{ij} = \log(T_{ij}^{-1}(T_i^{-1} T_j)) \in \mathfrak{se}(3), \tag{11}$$

where $\mathcal{E}$ is the set of all constraints on loop closure and odometry, T_{ij} is the relative transformation measured, Ω_{ij} is the information matrix and $\log(\cdot)$ maps $SE(3)$ to its $\mathfrak{se}(3)$. It yields corrected absolute poses T_k^* for each submap.

Inter-Agent Loop Closure. In runtime, agents auto-discover through the local network and continuously send keyframe features to other agents. The depth map is compressed and transferred only when a loop is confirmed. Similar to intra-agent loop detection, temporal filtering and multi-frame verification are also applied.

We propose a relative-to-global registration approach for efficient inter-agent pose estimation. The relative phase focuses on the transformation between agents s and t. The global phase takes the relative transformation as an initial guess and estimates the global poses of all keyframes. Given multiple pairwise relative pose measurements $T_{i,j}^{s,t}$ between the frame i of agent s and the frame j of agent t, for agent s with frame pose T_i^s and agent t with frame pose T_j^t, the constraint is established through transformation $\hat{T}_{st}$:

$$\hat{T}_{st}^* = \arg\min_{T_{st}} \sum_{(i,j) \in \mathcal{E}_{s,t}} d\left(T_i^s T_{i,j}^{s,t} T_{st}, T_j^t\right)^2, \tag{12}$$

where $\mathcal{E}_{s,t}$ is the set of loop closure constraints between agent s and agent t, and $d(\cdot, \cdot)$ denotes a distance metric on

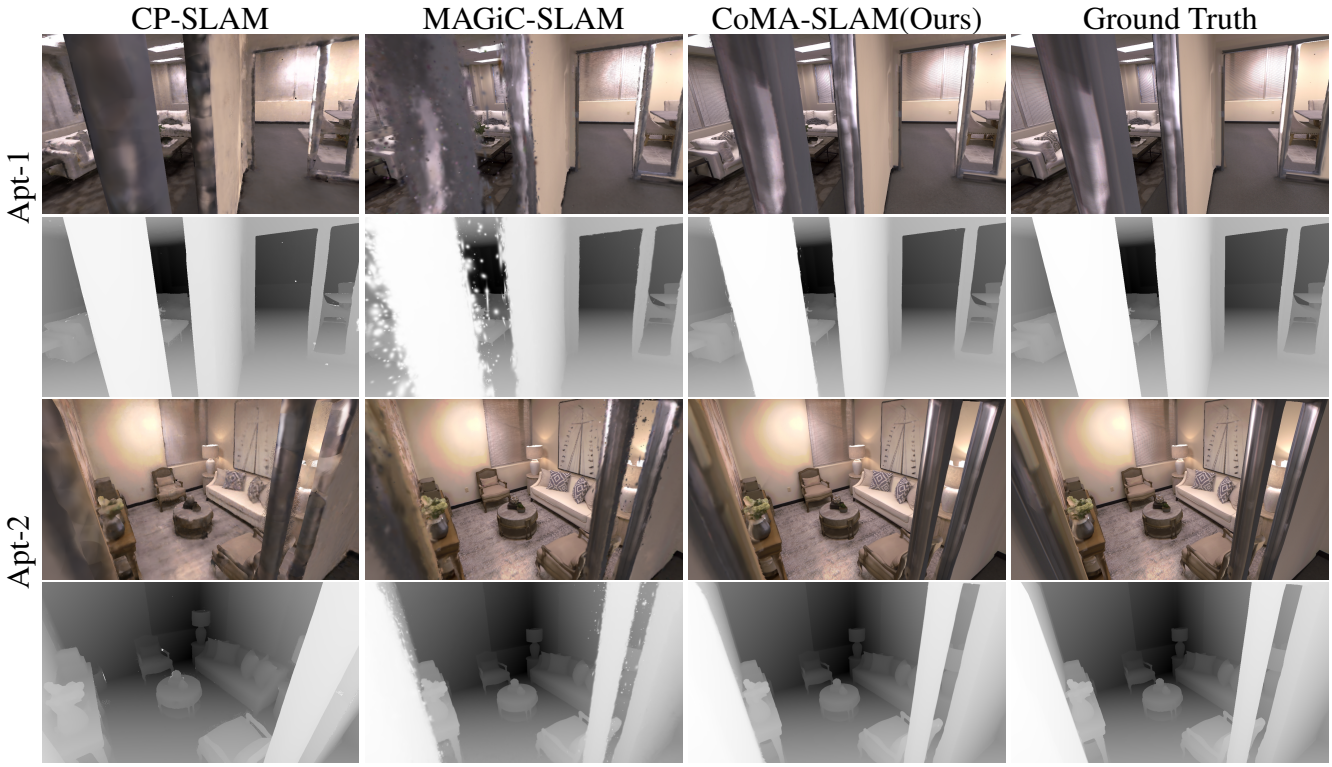

Figure 4: Rendering Comparision on ReplicaMultiagent. CoMA-SLAM achieves comparable rendering quality to the ground truth, and the depth maps are also very close to the ground truth, indicating good geometric consistency.

$SE(3)$. The optimized $\hat{T}_{st}^{*}$ serves as the unified transformation between agent s and t in the subsequent global PGO.

After the relative transformations T_{st} are estimated in the relative phase, we construct a global pose graph on each agent. In this graph, nodes represent submaps, and edges represent relative transformations: odometry edges $T_{i,i+1}$ from per-agent tracking, and inter-agent loop closure edges T_{st} from the relative phase. The global PGO jointly optimizes the global poses of all keyframes by minimizing the sum of squared Mahalanobis distances of edge errors.

Submap Centric Update. Global PGO typically requires dense point cloud adjustments for geometric consistency, which is challenging with differentiable rendering due to high computational cost. To address this, we propose a submap centric update mechanism that directly applies pose corrections to surfels, enabling efficient global refinement. Specifically, after global optimization, the correction transformation $T_i^{cor} = T_i^{g*}(T_i)^{-1}$ is computed for each submap i and applied to both camera poses and Gaussian Surfels G^i.

$$T_i^g \leftarrow T_i^{cor} T_i, \quad \mu_j^i \leftarrow T_i^{cor} \mu_j^i, \quad \mathbf{\Sigma}_j^i \leftarrow R_i^{cor} \mathbf{\Sigma}_j^i, \quad (13)$$

where R_i^{cor} is the rotational component of T_i^{cor}. This rigid transformation ensures surfel geometry aligns with the optimized pose while preserving computational efficiency.

3.4 Lightweight Communications

Centralized methods require transmitting raw data to a central server, whereas our method only transmits necessary poses and depth maps, resulting in minimal bandwidth usage. To further reduce bandwidth consumption, we also apply compression to the depth maps.

Depth Compression. It first quantizes the 16-bit depth image to 12-bit precision. The quantized image is then divided into 8×8 blocks, and for each block, the optimal predictor is selected to minimize prediction residuals. The residuals are processed using zigzag scanning and run-length compression. All data streams, including residuals, block predictor indices, and run-length data, are packed and further compressed using Zstd entropy coding. For a 1200×680 depth map, the compressed size is about 100KB, which is 16x smaller than the original 1.6MB.

4 Experiments

4.1 Experimental Setup

Implementation Details. For tracking optimization, we set the rotation and translation learning rates to 2e-4 and 2e-3, respectively. The opacity mask threshold τ_{opacity} is configured to 0.98, while the depth threshold τ_{depth} is set to 50 times the median depth error. The depth loss weight λ_d is set to 0.01. During mapping, the submap is reset every $\theta_{\text{sub}} = 20$ frames to maintain computational efficiency. The temporal adjacency threshold τ_{min} for loop closure detection

Method	Publication	ReplicaMultiagent (Ag.1 / Ag.2 / Avg)			AriaMultiagent (Ag.1 / Ag.2 / Ag.3 / Avg)	
		Off-0	Apt-0	Apt-1	Room0	Room1
CCM-SLAM	ICRA'17	9.84 / 0.76 / 5.30	✗	2.12 / 9.31 / 5.71	✗	✗
Swarm-SLAM	RAL'24	1.07 / 1.79 / 1.42	1.61 / 1.98 / 1.80	4.62 / 6.50 / 5.56	6.11 / 8.43 / 4.82 / 6.45	4.29 / 4.95 / 5.12 / 4.78
CP-SLAM	NIPS'23	0.50 / 0.79 / 0.65	0.62 / 1.28 / 0.95	1.11 / 1.72 / 1.42	0.68 / 5.39 / - /3.03	5.06 / 0.68 / - / 2.87
MAGiC-SLAM	CVPR'25	0.31 / 0.24 / 0.27	**0.13** / 0.21 / **0.16**	**0.21** / 0.30 / **0.26**	0.67 / 1.13 / 1.67 / 1.15	0.96 / 0.53 / 0.46 / 0.65
Ours(w/o LC)	-	0.15 / 0.23 / 0.19	0.19 / 0.31 / 0.25	0.38 / 0.43 / 0.41	0.69 / 0.61 / 0.59 / 0.63	0.99 / 0.91 / 0.58 / 0.83
Ours	-	**0.10 / 0.10 / 0.10**	**0.13 / 0.19 / 0.16**	0.24 / **0.29** / 0.27	**0.44 / 0.38 / 0.32 / 0.38**	**0.48 / 0.39 / 0.28 / 0.38**

Table 1. Tracking performance evaluation on both AriaMultiagent and ReplicaMultiagent datasets. Results marked with ✗ indicate invalid outputs, while "-" denotes unsupported configurations in baseline methods. "Ag.1" and "Ag.2" denote the first and second agents.(ATE RMSE [cm] ↓)

Method	Metric	Off-0	Apt-0	Apt-1	Apt-2	Avg.
CP-SLAM	PSNR [dB] ↑	28.56	26.12	12.16	23.98	22.71
	SSIM ↑	0.87	0.79	0.31	0.81	0.69
	LPIPS ↓	0.29	0.41	0.97	0.39	0.51
	Depth L1 [cm] ↓	2.74	19.93	66.77	2.47	22.98
MAGiC-SLAM	PSNR [dB] ↑	39.32	36.96	30.01	30.73	34.26
	SSIM ↑	0.99	0.98	0.95	0.96	0.97
	LPIPS ↓	0.05	0.09	0.18	0.17	0.12
	Depth L1 [cm] ↓	0.41	0.64	3.16	0.99	1.30
Ours	PSNR [dB] ↑	**42.77**	**42.28**	**33.86**	**34.59**	**38.38**
	SSIM ↑	**1.00**	**0.99**	**0.98**	**0.98**	**0.99**
	LPIPS ↓	**0.02**	**0.03**	**0.08**	**0.08**	**0.05**
	Depth L1 [cm] ↓	**0.22**	**0.27**	**0.93**	**0.48**	**0.48**

Table 2. Training view performance on ReplicaMultiagent. Our method consistently outperforms all the baselines.

Method	Metric	Room0	Room1	Avg.
MAGiC-SLAM	PSNR [dB] ↑	23.45	21.78	22.61
	SSIM ↑	0.89	**0.85**	0.87
	LPIPS ↓	0.22	0.21	0.22
	Depth L1 [cm] ↓	1.33	4.96	3.15
Ours	PSNR [dB] ↑	**25.41**	**21.80**	**23.60**
	SSIM ↑	**0.90**	**0.85**	**0.88**
	LPIPS ↓	**0.21**	**0.20**	**0.20**
	Depth L1 [cm] ↓	**0.70**	**4.80**	**2.8**

Table 3. Novel view synthesis performance on AriaMultiagent dataset. The global map built by merging the maps from three agents is evaluated by synthesizing novel views.

is set to 20 frames. All experiments were conducted on a workstation equipped with an NVIDIA RTX 4090 GPU, Intel Core i9-12900KF CPU.

Datasets. We evaluate our system on two established multi-agent benchmarks: ReplicaMultiagent (Hu et al. 2023) and AriaMultiagent (Yugay, Gevers, and Oswald 2024). ReplicaMultiagent comprises synthetic sequences with four distinct scenarios (Office-0, Apt-0, Apt-1, Apt-2) featuring two agents operating in Replica (Straub et al. 2019) environments. AriaMultiagent is derived from real-world egocentric Aria (Pan et al. 2023) recordings in two indoor environments, with sequences simulating three concurrent agents.

Evaluation Metrics. Tracking accuracy is quantified by Root Mean Square Absolute Trajectory Error (ATE RMSE) (Hu et al. 2023). For rendering quality assessment, we perform scene merging followed by 3000 iterations of fine-tuning, then evaluate using PSNR, SSIM (Wang et al. 2004), and LPIPS (Zhang et al. 2018) metrics by comparing full-resolution rendered images against input training views. Depth accuracy is measured using the L1 error between rendered and ground truth depth maps.

4.2 Tracking Performance

Extensive experiments were conducted on two multi-agent benchmark datasets. As demonstrated in Table 1, CoMA-SLAM consistently achieves the lowest tracking errors across nearly all sequences and agents, demonstrating substantial performance advantages over state-of-the-art multi-agent SLAM methods. The surfel ray intersection mechanism substantially enhances tracking robustness for each individual agent. Furthermore, the incorporation of loop closure optimization effectively reduces trajectory drift, enabling millimeter-level positioning accuracy on average. Notably, while conventional methods such as CCM-SLAM (Schmuck and Chli 2019) frequently fail in challenging scenarios, neural-based approaches (Hu et al. 2023) and 3DGS-based methods (Yugay, Gevers, and Oswald 2024) exhibit considerably larger tracking errors compared to our 2D surfel representation approach.

To further substantiate these findings, we provide single-agent tracking results in the appendix, systematically comparing our method against feature-based, neural-based, and Gaussian-based approaches. These supplementary experiments demonstrate the robustness of our surfel-based representation in multi-agent environments.

4.3 Rendering Performance

As shown in Tables 2 and 3, CoMA-SLAM achieves rendering improvements over NeRF-based method CP-SLAM (Hu et al. 2023) and the previous state-of-the-art Gaussian method MAGiC-SLAM (Yugay, Gevers, and Oswald 2024), attaining higher PSNR/SSIM and lower LPIPS/Depth L1 metrics. These results demonstrate the superiority of our geometrically precise surfel representation, particularly in Depth L1 metrics where geometric accuracy is critical. As shown in Fig. 4, while MAGiC-SLAM (Yugay, Gevers, and

Method	Map/Fr.	Track/Fr.	Memory	Comm./Fr.
MAGiC-SLAM	**0.31s**	**0.10s**	54.6MB	1.7MB
Ours	0.36s	0.16s	**47.4MB**	**2.2KB**

Table 4. Performance comparison of runtime, memory, and communication data size. The reported memory consumption refers to the scene representation storage size. Communication indicates the bandwidth required for data exchange.

Method	ATE [cm]↓	PSNR [dB]↑	Depth L1 [cm]↓
w/o 2DGS	0.24	42.21	0.31
w/o Surfel Initialization	0.18	41.40	0.37
w/o Loop Closure	0.21*	42.73*	0.23*
Ours (Full)	**0.10**	**42.77**	**0.22**

Table 5. Ablation Study on Key Components. The * indicates the metrics are computed by independently evaluating the method on each agent and then averaging the results.

Oswald 2024) struggles with real-world data, our method shows clear qualitative advantages in preserving fine details and structural fidelity. These benefits arise from our surfel-based framework that enables accurate sub-map rendering through explicit surface modeling, loop-closure optimized pose correction, and an effective map-merging strategy, collectively ensuring superior performance.

Although the Surfel representation inherently faces challenges in rendering (Huang et al. 2024a), our method achieves superior quality in both training and novel view synthesis compared to state-of-the-art methods MAGiC-SLAM (Yugay, Gevers, and Oswald 2024). This is attributed to our initialization strategy, which positions Surfel closely to the object surface, allowing for efficient mapping that can effectively produce high-quality renderings.

4.4 Runtime, Memory and Bandwidth Analysis

As shown in Table 4, CoMA-SLAM demonstrates competitive runtime performance compared to MAGiC-SLAM on the ReplicaMultiagent Office-0 dataset. Although our per-frame mapping speed is marginally slower, we achieve a 15.4% reduction in memory footprint through the adaptive growth strategy. More importantly, our distributed design requires only 2.2KB per frame on average for inter-agent communication, showcasing superior scalability in multi-agent scenarios—a critical advantage over centralized approaches that are constrained by the computational and memory limits of a single platform. By leveraging local processing, CoMA-SLAM seamlessly scales with the number of agents without relying on a centralized bottleneck.

4.5 Ablation Studies

Ablation studies are conducted to evaluate the contribution of key components in our system. All experiments are conducted on the ReplicaMultiagent Office-0 dataset.

2D Gaussian Surfels vs 3D Gaussians. We compare our 2D Gaussian Surfel representation against 3D Gaussians in Table 5. While maintaining comparable rendering quality, the 2D representation achieves superior tracking accuracy. This is because 2D Gaussian Surfel enhances surface modeling and geometric consistency through accurate intersection computation. Although 2D Gaussian Surfel lacks one dimension compared to 3D Gaussian, which slightly hurts the rendering efficiency (Huang et al. 2024a), our method still achieves comparable rendering quality to 3D Gaussian.

Method	ATE [cm] ↓	PSNR [dB] ↑	Gaussian Count ↓
w/o Adaptive Growth	**0.10**	42.60	1.05M
Ours (Full)	**0.10**	**42.77**	**0.71M**

Table 6. Ablation Study on Adaptive Growth. The Gaussian Count is evaluated after merging the submaps.

Surface Fitting Initialization. We compare our initialization strategy with random initialization of the scale and orientation of Gaussian surfels. The results show that random initialization leads to notably worse tracking and rendering performance. This is because random initialization produces degraded rendering quality, which in turn requires more optimization iterations to achieve similar results as our method.

Adaptive Gaussian Growth. We evaluate our adaptive growth strategy against MAGiC-SLAM's default Gaussian spawning method in Table 6. Quantitative results demonstrate our approach maintains superior rendering quality while reducing Gaussian primitives by 32%. This efficiency gain compensates for the 2D Surfel's inherent renderer overhead, as described in Sec. 4.4.

Loop Closure. Loop closure can reduce tracking drift and improve overall localization accuracy. Here we only compare the localization accuracy of within single agent, because without loop closure, we cannot recover the global consistent pose. As shown in Table 5, the performance drops by 0.11cm without Loop Closure.

4.6 Limitations and Future Work

Although 2D Gaussian Surfels offer advantages in geometric consistency, their rendering efficiency lags behind that of 3D Gaussian renderers. This fundamental limitation persists even after reducing Gaussian primitive counts. Future work could explore faster convergence methods to improve frame rate while maintaining robustness and accuracy.

5 Conclusion

We presented CoMA-SLAM, the first distributed multi-agent Gaussian SLAM system that achieves state-of-the-art performance by integrating the geometric advantages of 2D Gaussian Surfels into a globally consistent multi-agent framework. By enhancing the accuracy and robustness of each agent's local SLAM process and combining it with effective loop closure and map merging strategies, our system delivers superior tracking accuracy, high-fidelity reconstructions, and realistic novel view synthesis capabilities.

References

Chen, L.; Hu, B.; Wang, J.; Bu, S.; Wang, G.; Han, P.; and Chen, J. 2025. G2-Mapping: General Gaussian Mapping for Monocular, RGB-D, and LiDAR-Inertial-Visual Systems. *IEEE Transactions on Automation Science and Engineering*, 1–1.

Douze, M.; Guzhva, A.; Deng, C.; Johnson, J.; Szilvasy, G.; Mazaré, P.-E.; Lomeli, M.; Hosseini, L.; and Jégou, H. 2024. The faiss library. *arXiv preprint arXiv:2401.08281*.

Hu, J.; Mao, M.; Bao, H.; Zhang, G.; and Cui, Z. 2023. Cp-slam: Collaborative neural point-based slam system. *Advances in Neural Information Processing Systems*, 36: 39429–39442.

Huang, B.; Yu, Z.; Chen, A.; Geiger, A.; and Gao, S. 2024a. 2D Gaussian Splatting for Geometrically Accurate Radiance Fields. In *SIGGRAPH 2024 Conference Papers*. Association for Computing Machinery.

Huang, H.; Li, L.; Hui, C.; and Yeung, S.-K. 2024b. 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*.

Keetha, N.; Karhade, J.; Jatavallabhula, K. M.; Yang, G.; Scherer, S.; Ramanan, D.; and Luiten, J. 2024. SplaTAM: Splat, Track and Map 3D Gaussians for Dense RGB-D SLAM. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Kerbl, B.; Kopanas, G.; Leimkühler, T.; and Drettakis, G. 2023. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. *ACM Transactions on Graphics*, 42(4).

Kümmerle, R.; Grisetti, G.; Strasdat, H.; Konolige, K.; and Burgard, W. 2011. G2o: A general framework for graph optimization. In *2011 IEEE international conference on robotics and automation*, 3607–3613. IEEE.

Lajoie, P.-Y.; and Beltrame, G. 2023. Swarm-slam: Sparse decentralized collaborative simultaneous localization and mapping framework for multi-robot systems. *IEEE Robotics and Automation Letters*, 9(1): 475–482.

Lajoie, P.-Y.; Ramtoula, B.; Chang, Y.; Carlone, L.; and Beltrame, G. 2020. DOOR-SLAM: Distributed, online, and outlier resilient SLAM for robotic teams. *IEEE Robotics and Automation Letters*, 5(2): 1656–1663.

Matsuki, H.; Murai, R.; Kelly, P. H. J.; and Davison, A. J. 2024. Gaussian Splatting SLAM. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Mildenhall, B.; Srinivasan, P. P.; Tancik, M.; Barron, J. T.; Ramamoorthi, R.; and Ng, R. 2020. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. In *ECCV*.

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

Pan, X.; Charron, N.; Yang, Y.; Peters, S.; Whelan, T.; Kong, C.; Parkhi, O.; Newcombe, R.; and Ren, Y. C. 2023. Aria digital twin: A new benchmark dataset for egocentric 3d machine perception. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 20133–20143.

Park, J.; Zhou, Q.-Y.; and Koltun, V. 2017. Colored point cloud registration revisited. In *Proceedings of the IEEE international conference on computer vision*, 143–152.

Peng, Z.; Shao, T.; Yong, L.; Zhou, J.; Yang, Y.; Wang, J.; and Zhou, K. 2024. RTG-SLAM: Real-time 3D Reconstruction at Scale using Gaussian Splatting. *ACM Transactions on Graphics*.

Schmuck, P.; and Chli, M. 2019. CCM-SLAM: Robust and efficient centralized collaborative monocular simultaneous localization and mapping for robotic teams. *Journal of Field Robotics*, 36(4): 763–781.

Straub, J.; Whelan, T.; Ma, L.; Chen, Y.; Wijmans, E.; Green, S.; Engel, J. J.; Mur-Artal, R.; Ren, C.; Verma, S.; Clarkson, A.; Yan, M.; Budge, B.; Yan, Y.; Pan, X.; Yon, J.; Zou, Y.; Leon, K.; Carter, N.; Briales, J.; Gillingham, T.; Mueggler, E.; Pesqueira, L.; Savva, M.; Batra, D.; Strasdat, H. M.; Nardi, R. D.; Goesele, M.; Lovegrove, S.; and Newcombe, R. 2019. The Replica Dataset: A Digital Replica of Indoor Spaces. *arXiv preprint arXiv:1906.05797*.

Tian, Y.; Chang, Y.; Arias, F. H.; Nieto-Granda, C.; How, J. P.; and Carlone, L. 2022. Kimera-multi: Robust, distributed, dense metric-semantic slam for multi-robot systems. *IEEE Transactions on Robotics*, 38(4).

Wang, Z.; Bovik, A.; Sheikh, H.; and Simoncelli, E. 2004. Image quality assessment: from error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4): 600–612.

Yan, C.; Qu, D.; Xu, D.; Zhao, B.; Wang, Z.; Wang, D.; and Li, X. 2024. GS-SLAM: Dense Visual SLAM with 3D Gaussian Splatting. In *CVPR*.

Yu, Z.; Sattler, T.; and Geiger, A. 2024. Gaussian Opacity Fields: Efficient Adaptive Surface Reconstruction in Unbounded Scenes. *ACM Transactions on Graphics*.

Yugay, V.; Gevers, T.; and Oswald, M. R. 2024. MAGiC-SLAM: Multi-Agent Gaussian Globally Consistent SLAM. *arXiv preprint arXiv:2411.16785*.

Zhang, B.; Fang, C.; Shrestha, R.; Liang, Y.; Long, X.; and Tan, P. 2024. RaDe-GS: Rasterizing Depth in Gaussian Splatting. arXiv:2406.01467.

Zhang, R.; Isola, P.; Efros, A. A.; Shechtman, E.; and Wang, O. 2018. The Unreasonable Effectiveness of Deep Features as a Perceptual Metric. *CoRR*, abs/1801.03924.

Zhu, L.; Li, Y.; Sandstrom, E.; Huang, S.; Schindler, K.; and Armeni, I. 2024. LoopSplat: Loop Closure by Registering 3D Gaussian Splats. arXiv:2408.10154.

Zhu, Z.; Peng, S.; Larsson, V.; Xu, W.; Bao, H.; Cui, Z.; Oswald, M. R.; and Pollefeys, M. 2022. NICE-SLAM: Neural Implicit Scalable Encoding for SLAM. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.