CP-SLAM: Collaborative Neural Point-based SLAM Supplementary Material

Jiarui Hu[1], Mao Mao[1], Hujun Bao[1], Guofeng Zhang[1], Zhaopeng Cui[1]*

[1]State Key Lab of CAD&CG, Zhejiang University

In this supplementary material, we first describe more details about how to decode origin neighbor features to the density and radiance in Section. A. Next, we provide additional experimental details in Section. B, including depth mask, hyperparameter settings, and PGO implementation. In Section. C, we comprehensively investigate the applicability of the centralized learning strategy in collaborative neural SLAM, providing further evidence for the rationality of our proposed two-stage learning strategy. In Section. D, we provide more discussion on the point cloud filtering strategy for sub-map fusion and global map refinement. We added details about the dataset generation and tracking evaluation in Section. E. Finally, we provide a comprehensive analysis of additional experiments in Section. F.

A Feature Decoding

Three lightweight MLPs C, G, U are used in the proposed system for feature transfer and predicting meaningful density and radiance. The entire decoding process is visualized in Fig. 1. The 3-dimensional relative displacement and the original 32-dimensional neural point feature are extended to 45 and 96 dimensions respectively through positional encoding (Eq. 1) with an order of 7 and 1. MLP C has one hidden layer with 256 neurons followed by $LeakyRelu$ activation. We found that $LeakyRelu$ activation, instead of $Relu$, can enhance training stability and speed convergence. Considering the complexity of the radiance field, we set 2 hidden layers in the radiance decoder U with 128 neurons. Benefitting from neural point representation and concentrative sampling strategy, we can obtain a high-quality depth map, as shown in Fig. 2, with very fewer training iterations by employing MLP G containing one hidden layer with 256 neurons.

$$\gamma(p) = (\sin(2^0\pi p), \cos(2^0\pi p), ..., \sin(2^{L-1}\pi p), \cos(2^{L-1}\pi p)). \tag{1}$$

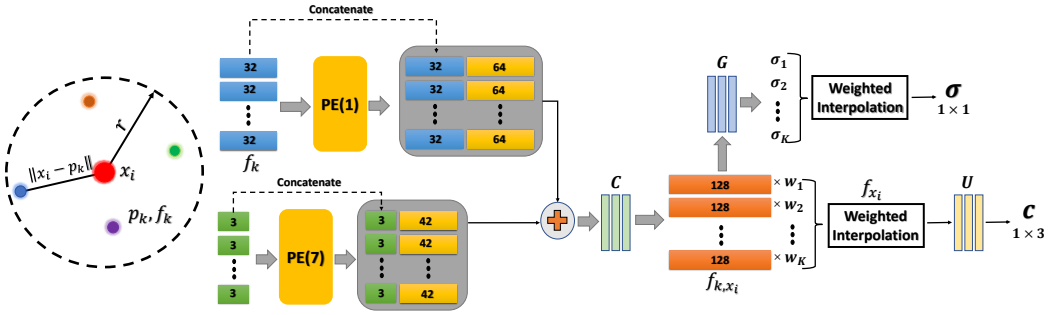


Figure 1: **Feature Decoding.** PE(N) represents the positional encoding with an order of N, and '$\bigoplus$' represents the concatenation of encoded relative displacement and neighbor features.

*Corresponding author.

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

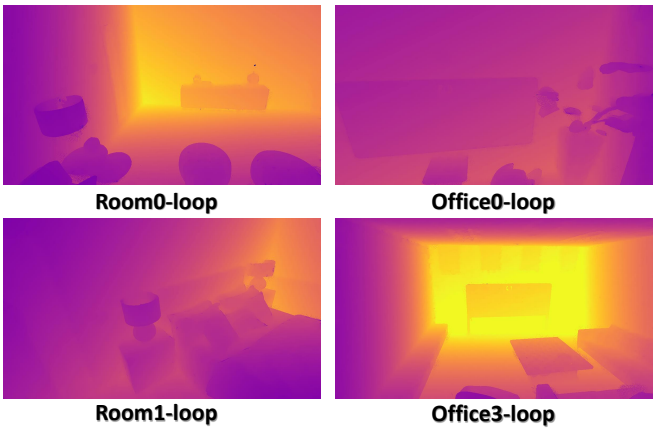


Figure 2: **Rendered Depth Map.** With a simple MLP G, neural point cloud field and concentrated sample points, it is possible to efficiently render precise geometry in only 500 optimization steps.

B More Implementation Details

$CP\text{-}SLAM$ is implemented using Python3.7 and Pytorch1.11. We use Adam optimizer with different learning rates in tracking and mapping. Specifically, we set learning rates to 0.0015, 0.003, and 0.005 for the pose, MLPs, and neural point features. We found that, in the mapping process, imposing a learning rate decay strategy as described in Eq. 2 for feature optimization is helpful to regress the correct neural field.

$$lr_{update} = lr_{init} * 0.1^{\frac{iter}{10000}}. \tag{2}$$

At the same time, in order to prevent a too low learning rate caused by long-term work, we reset the learning rate to the initial value and reuse this strategy prior to each mapping iteration. During pose estimation, we evaluate the uncertainty of each rendered pixel and exclude zero-depth and outlier pixels in the tracking loss function $L_{tracking}$. A pixel is considered an outlier if it satisfies the following condition:

$$0.1\mu \leq |D - \hat{D}| \leq 10\mu \ \ or \ \ Var \leq 2\nu, \tag{3}$$

where D, V represent the rendered depth and uncertainty of a pixel, $\hat{D}$ is the ground truth depth, and μ, ν denote median depth error and median uncertainty in a batch. In terms of PGO, we use the g2opy [3] library, an open-source and efficient framework for optimizing graph-based nonlinear error functions.

C Will the Centralized Learning Work?

As mentioned in Section 3.3 of our main paper, we have developed a novel two-stage learning strategy, i.e., distributed-to-centralized learning, for the proposed collaborative neural SLAM. One may be curious about whether the centralized learning works. To verify this, we have attempted to perform centralized learning from scratch. However, such a mechanism was proved to be remarkably ineffectual and even failed to learn the correct field at all during the initialization. In our analysis, this phenomenon can be attributed to the aliasing effect. In collaborative SLAM, all agents set their initial coordinate system as the identity system $\mathcal{I}$ located at $\mathcal{O}$ $(0, 0, 0)$. If centralized learning is performed on all sequences from scratch, it is equivalent to aliasing different neural fields in the coordinate system $\{\mathcal{I}, \mathcal{O}\}$. The messy and mutually interfering density and radiance distribution make it impossible to regress the correct neural field.

D Point Cloud Filtering Strategy

In our proposed system, the processing for the point cloud, such as sub-map fusion or global map refinement, will inevitably lead to point cloud redundancy in local regions. Therefore, we will perform grid-based filtering in 3D space. Taking one cube as an example, as shown in Fig. 3, only the

nearest neural point to the center of this cube is retained, which is enough to represent latent spatial information in this cube. The size of cubes ρ directly determines the sparsity of the neural point field. Through extensive experiments, we have found that the optimal cube size should be slightly smaller than the search radius r, which can avoid the empty neighbor point and achieve a good trade-off between accuracy and efficiency. In our experiments, we set $\rho = 0.14m$ and $r = 0.15m$.

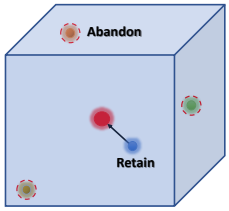


Figure 3: **Neural Point Filtering.** Non-maximum suppression based on the distance to the cube center.

E Dataset Generation and Trajectory Evaluation

In order to fully demonstrate the capabilities of each module in our $CP\text{-}SLAM$ system, we customize single-agent trajectories with loop closure (Room-0-loop, Room-1-loop, Office-0-loop, Office-3-loop) and collaborative trajectories (Apartment-0, Apartment-1, Apartment-2, Office-0-C) in Blender. Then, in the Replica SDK [5], we render depth and color maps along the customized trajectories, in which the camera intrinsics, image resolution, and depth scale remained the same as NICE-SLAM [7]. In addition, all the methods mentioned in this paper for trajectory assessment, except for CCM-SLAM [4], are RGB-D-based. For the camera trajectories generated by CCM-SLAM, we align them with the Ground Truth camera trajectory using Sim(3) Umeyama alignment in the EVO [1] tool. As for the camera trajectories produced by other methods, we align them with the Ground Truth camera trajectory by aligning the origin. Trajectory alignment is crucial for proper drift and loop closure evaluation. To be specific, after aligning the initial poses, we calculate the Absolute Trajectory Error (ATE) for each pose and compute the RMSE, Mean, and Median values.

F Additional Experiments

F.1 Completion Metric Evaluation with the Culling Strategy

We have introduced the culling strategy of ESLAM [2] into the completion metric evaluation, and the results are listed in Table. 1 and Table. 2. As shown, our method achieves state-of-the-art performance in terms of completion benefiting from high accuracy, while performing on par with the SOTA method (Vox-Fusion [6]) in terms of completion ratio, which validates the effectiveness of our method for the single-agent SLAM.

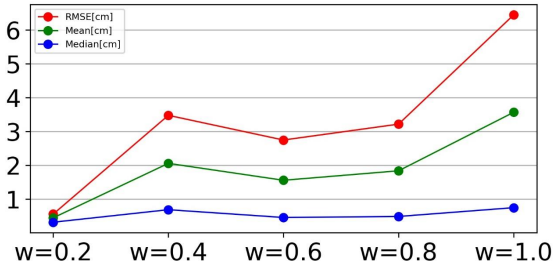


Figure 4: **Color Loss Ablation in Room0-loop.**

Method	Office0-loop	Office3-loop	Room0-loop	Room1-loop
NICE-SLAM	1.69	2.22	1.74	1.73
Vox-Fusion	1.11	1.51	1.32	1.06
Ours	**1.04**	**1.47**	**1.21**	**1.01**

Table 1: **Completion [cm] (↓) Metric.** The culling strategy is adopted in the completion evaluation. It can be observed that our method achieves state-of-the-art performance.

Method	Office0-loop	Office3-loop	Room0-loop	Room1-loop
NICE-SLAM	97.22	94.82	98.14	97.98
Vox-Fusion	**99.69**	**98.87**	**99.35**	**99.84**
Ours	99.45	98.34	99.185	99.70

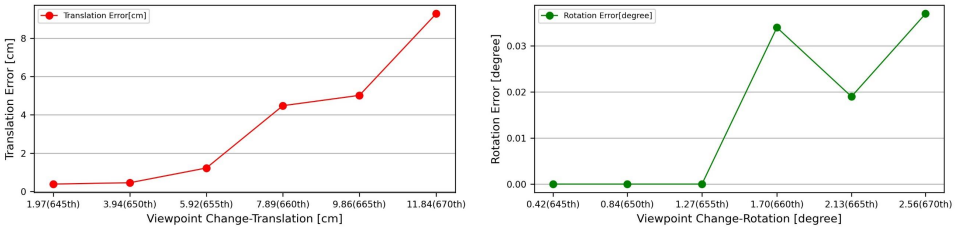
Table 2: **Completion Ratio [<5cm, %] (↑) Metric.** The culling strategy is adopted in the completion ratio evaluation. It can be observed that our method performs better than NICE-SLAM and is on par with Vox-Fusion.

F.2 Color Loss Weight

To further explore the strong non-convexity of the color map, we evaluate the tracking performance at the setting of increasing color loss weights. It can be observed that, in Fig. 4, as the weight of color loss w increases from 0.2 to 1, pose error also rises consistently. This confirms our point in Section.3.1.

F.3 Viewpoint Change

As pointed out in Section.5 in our paper, the rendering-based optimization is limited in the face of large viewpoint changes, which remains a bottleneck for other existing works, such as NICE-SLAM [7] and Vox-Fusion [6]. We conduct an ablation study on viewpoint difference in Fig. 5. As the inter-frame interval increases, translation and rotation errors gradually increase.



(a) (b)

Figure 5: **Viewpoint Change Ablation in Room0-loop.** (a) Translation error grows with larger viewpoint differences. (b) Rotation error grows with larger viewpoint differences.

References

[1] Michael Grupp. evo: Python package for the evaluation of odometry and slam. `https://github.com/MichaelGrupp/evo`, 2017.

[2] 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 international conference on Computer Vision and Pattern Recognition (CVPR)*, 2023.

[3] Rainer Kümmerle, Giorgio Grisetti, Hauke Strasdat, Kurt Konolige, and Wolfram Burgard. g 2 o: A general framework for graph optimization. In *2011 IEEE International Conference on Robotics and Automation*, pages 3607–3613. IEEE, 2011.

[4] Patrik Schmuck and Margarita Chli. Ccm-slam: Robust and efficient centralized collaborative monocular simultaneous localization and mapping for robotic teams. *Journal of Field Robotics*, 36(4):763–781, 2019.

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

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

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