Supplementary Materials for: RGBD GS-ICP SLAM

Seongbo Ha, Jiung Yeon, and Hyeonwoo Yu

Sungkyunkwan University, Suwon, South Korea
{sobo3607,wcr12st,hwyu}@skku.edu

This supplementary material includes the following sections:

A. Further Implementation Details

Our G-ICP [9] module is implemented based on the VGICP [6], which is implemented in C++ and parallel computation for faster processing. The C++ implementation of G-ICP is wrapped using pybind11 to make it accessible from Python. The SLAM system is structured in parallel to operate with two processes, tracking and mapping, using PyTorch multiprocessing.

A.1. Tracking Process

G-ICP Before generating the source point cloud from the current depth image, we downsample it by 1/10 in the Replica [10] dataset and by 1/5 in the TUM-RGBD [11] dataset. These values are chosen based on the resolution of the images. The covariances of the source point cloud are computed within the G-ICP module by finding the 10 nearest points to each point to make source Gaussians from the source point cloud. Source Gaussians are aligned with the target Gaussians existing in the 3D GS map to estimate the current camera pose, while omitting those target Gaussians with low opacity (below 0.05 in Replica and 0.09 in TUM). During the aligning procedure, correspondences between source and target Gaussians are determined based on Euclidean distances, excluding those with distances greater than 2cm in Replica and 3cm in TUM.

Keyframe Selection and Adding Target Gaussians We select the current frame as a keyframe when the proportion of source Gaussians corresponding with target Gaussians

falls below specific thresholds set to 70% in Replica, and 81% in TUM. Correspondences between the source and target Gaussians are inherited from G-ICP, and those with distances exceeding 0.05cm in Replica, and 0.1cm in TUM are filtered out. We incorporate Gaussians from the keyframe into the map as new target Gaussians, excluding those that overlap with corresponding target Gaussians already present in the map. To achieve this, we use a threshold of 0.005 cm in Replica and 0.1 cm in TUM.

A.2. Mapping Process

As a result of the tracking procedure using G-ICP, new primitives are added to the 3D GS map with the appropriate initial state to represent the scene accurately. The newly added primitives inherit their initial values from the scale-aligned source Gaussians, along with the color values from the original image and an opacity of 0.1. In the mapping process, we further optimize the Gaussians $\boldsymbol{G}$ and their color set $\boldsymbol{\mathcal{C}}$ and opacity set $\boldsymbol{O}$ to characterize the scene more precisely, as outlined below:

$$\boldsymbol{\mathcal{X}}^*, \boldsymbol{\mathcal{C}}^*, \boldsymbol{H}^*, \boldsymbol{O}^* = \underset{\boldsymbol{\mathcal{X}}, \boldsymbol{\mathcal{C}}, \boldsymbol{H}, \boldsymbol{O}}{\operatorname{argmin}} \lambda_{I_1} \boldsymbol{\mathcal{L}}_1 \left(I, I_{gt}\right) + \lambda_{I_2} \boldsymbol{\mathcal{L}}_{D-SSIM} \left(I, I_{gt}\right) + \lambda_D \boldsymbol{\mathcal{L}}_1 \left(D, D_{gt}\right)$$

Each pixel of RGB (I_p) and depth images (D_p) is synthesized by blending $\mathcal{N}$ Gaussians overlapping it as the followings:

$$I_p = \sum_{i \in \mathcal{N}} \mathbf{c}_i \alpha_i \prod_{j=1}^{i-1} \left(1 - \alpha_j\right), \; D_p = \sum_{i \in \mathcal{N}} z_i \alpha_i \prod_{j=1}^{i-1} \left(1 - \alpha_j\right)$$

We consider Gaussians as view-independent primitives, thus excluding spherical harmonics that enable Gaussians to represent view-dependent color. This omission enhances the speed of both rendering and optimizing Gaussians. We optimize the 3D GS map on the selected keyframes and mapping-only keyframes from the tracking process. Upon adding a new keyframe or mapping-only keyframe, we utilize it once, then for subsequent optimizing iterations, we randomly select from a set of keyframes and mapping-only keyframes. The 3D GS map is optimized using RGBD images with their original resolution. Every 200 iterations, we prune Gaussians with a maximum scale larger than 0.25m in Replica and 1.0m in TUM, or with opacity less than 0.005. We utilize the Adam optimizer for optimizing Gaussians, setting the following learning rates: 0.000004 for position, 0.001 for rotation, 0.0025 for color, 0.05 for opacity, and 0.005 for scale.

B. Experimental Details

Baseline Methods We compare the proposed method with SLAM approaches utilizing NeRF [7] and those utilizing 3DGS [5]. Among NeRF-based methods, we compare our method with Point-SLAM [8], as it is the state-of-the-art (SOTA) method for NeRF-based approaches. Additionally, we evaluate our method against older NeRF-based methods, NICE-SLAM [13] and Orbeez-SLAM [2]. We compare our approach with

GS-SLAM [12], Photo-SLAM [3], SplaTAM [4], as 3DGS-based methods. For tracking accuracy, we also evaluated ORB-SLAM3 [1], the base method of Photo-SLAM.

Evaluation Settings To evaluate tracking accuracy, we first align the estimated trajectory with the ground truth trajectory, and then calculate the translation error between them. We evaluated the rendering quality of every image in the datasets. Because the proposed method conducts mapping solely on keyframes and mapping-only keyframes, rather than on every frame, we demonstrate the novel and trained view rendering performance of our method by evaluating it in this manner.

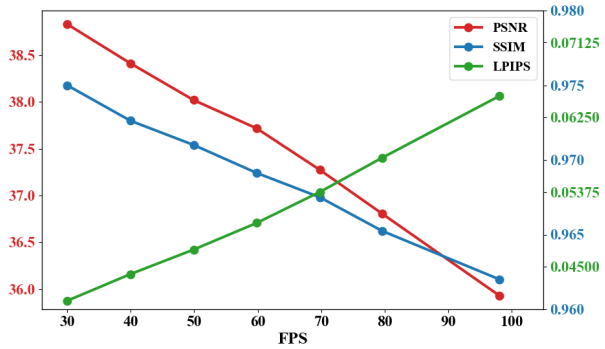

Fig. 1: Map Quality According to System Speed. Results represent the average performance across 8 scenes in Replica.

C. Additional Experimental Results

C.1. Map Quality According to System Speed

Tab. 1 presents the map quality and tracking accuracy of our method with respect to system FPS. We compare the map quality across scenarios where the system FPS is limited to 5, 10, 15, 20, 30, 40, 50, 60, 70, 80, and where no limitations are imposed. We found that the best performance is achieved at around 15FPS with robust tracking across all FPS settings. And even in situations of fast system speed, there is no significant degradation in map quality. This result underscores the ability to rapidly reconstruct the map, facilitated by providing the covariance computed in G-ICP and the color values from the original image as initial values for the 3D GS map primitives.

Methods that perform tracking based on rendering loss rely on differences between the rendered images in the reconstructed map and the images observed at the current time to estimate the camera pose. Therefore, most portion of the rendered image must be synthesized from sufficiently optimized regions of the map. This implies that the majority of the currently observed area is presumed to be adequately optimized. This assumption places constraints on system speed, and degrades tracking performance when the

camera moves rapidly and encounters new areas. In contrast, the proposed method incorporates target Gaussians into the map with information about the surrounding space, enabling their use for tracking even before they are fully optimized through the mapping process. As a result, our method demonstrates stable tracking accuracy even in scenarios where the system speed is fast, indicating that the map representing the observed area may not be fully optimized yet. This highlights the robust tracking performance of our method, even when the camera moves rapidly or when the observed images contain numerous new areas.

C.2. Geometric Quality of the Map

Our method exhibits an average depth L1 error/maximum depth of 0.030m/5.5m and 0.118m/8.5m across all scenes in Replica and TUM. The ATE RMSEs are 0.002m and 0.024m for Replica and TUM, indicating the impact of the geometric quality on the tracking accuracy. A 3DGS+SLAM work, GS-SLAM [12] reported their depth L1 error as 0.012m/5.5m in Replica. Since our system does not involve bundle adjustment for system speed, resulting in relatively lower depth quality.

C.3. Further Analysis of Advantages of the Connection between G-ICP and 3DGS

Our method achieves efficiency in both parts, G-ICP and 3DGS. Since we use scan-to-map matching for robust tracking, the number of target points continues to increase. Calculating the covariances of these points is computationally expensive, so we utilize the parameters existing in the 3DGS map. Using this method, we found that the system FPS increases from 97.97 to 103.61, and shows the same tracking accuracy. In the aspect of 3DGS, the covariances computed in G-ICP are utilized as the initial values for new Gaussians in the 3DGS map to minimize optimization iterations. While we validated this approach in the paper, we further assessed the efficiency aspect by comparing scenarios where the covariances from G-ICP are not used (case 1), and they are used but scale aligning is not applied (case 2). On Replica office4, at 30FPS, the proposed method achieved 38.75 PSNR, but in case 1, even after an additional 1000s of training, the maximum PSNR was 27.52. In case 2, after an additional 75s of training, it showed similar PSNR to the proposed system.

C.4. Qualitative Results

In this section, we showcase the qualitative results to visually demonstrate our method's capability of reconstructing maps in high-fidelity quality. The results of our methods are obtained under the condition of limiting the system speed to 30 FPS. We compare the results of our method with those of SplaTAM and Point-SLAM on Replica and TUM datasets. Despite the system speed of the proposed method being over 90 times faster than Point-SLAM and SplaTAM, it still shows the highest map quality.

Fig. 2: Novel View Rendering Results on Replica room1.

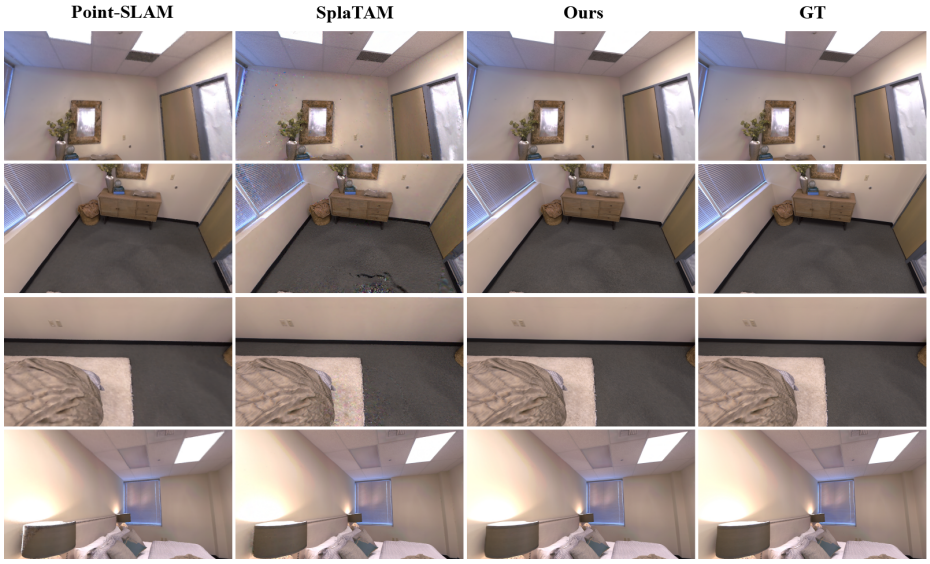

Fig. 3: Rendering Comparison on Replica room1.

Table 1: Map Quality According to System Speed on Replica.

Condition	Metrics	R0	R1	R2	Of0	Of1	Of2	Of3	Of4	Avg.
Not limited	PSNR [dB] ↑	32.20	35.36	34.42	40.31	40.75	33.85	34.08	36.47	35.93
	SSIM ↑	0.940	0.960	0.957	0.978	0.977	0.962	0.953	0.963	0.962
	LPIPS ↓	0.081	0.067	0.083	0.045	0.051	0.069	0.067	0.065	0.066
	ATE RMSE ↓	**0.15**	**0.16**	**0.10**	**0.17**	**0.12**	**0.16**	**0.16**	0.22	**0.16**
Limited to 80 FPS	PSNR [dB] ↑	33.27	35.67	35.79	40.92	41.75	35.07	34.75	37.23	36.81
	SSIM ↑	0.949	0.961	0.964	0.979	0.980	0.966	0.957	0.966	0.965
	LPIPS ↓	0.068	0.064	0.069	0.040	0.042	0.060	0.060	0.059	0.058
	ATE RMSE ↓	**0.15**	**0.16**	**0.10**	0.28	**0.12**	**0.16**	**0.16**	**0.20**	**0.16**
Limited to 70 FPS	PSNR [dB] ↑	33.64	36.11	36.45	41.57	42.30	35.26	35.42	37.44	37.28
	SSIM ↑	0.951	0.964	0.966	0.982	0.982	0.967	0.961	0.967	0.968
	LPIPS ↓	0.065	0.060	0.065	0.036	0.038	0.056	0.054	0.056	0.054
	ATE RMSE ↓	**0.15**	**0.16**	**0.10**	0.26	**0.12**	**0.16**	**0.16**	**0.20**	**0.16**
Limited to 60 FPS	PSNR [dB] ↑	34.10	36.63	36.90	42.06	42.59	35.75	35.84	37.89	37.72
	SSIM ↑	0.953	0.966	0.968	0.983	0.983	0.969	0.963	0.968	0.969
	LPIPS ↓	0.061	0.056	0.060	0.033	0.035	0.052	0.051	0.053	0.050
	ATE RMSE ↓	**0.15**	**0.16**	**0.10**	0.18	**0.12**	0.17	0.20	**0.20**	**0.16**
Limited to 50 FPS	PSNR [dB] ↑	34.56	36.88	37.50	42.24	42.57	36.04	36.21	38.15	38.02
	SSIM ↑	0.957	0.968	0.971	0.984	0.983	0.970	0.965	0.970	0.971
	LPIPS ↓	0.056	0.052	0.055	0.031	0.034	0.049	0.048	0.051	0.047
	ATE RMSE ↓	0.16	**0.16**	**0.10**	**0.17**	**0.12**	**0.16**	0.19	**0.20**	**0.16**
Limited to 40 FPS	PSNR [dB] ↑	35.00	37.34	38.04	42.67	43.08	36.26	36.41	38.49	38.41
	SSIM ↑	0.960	0.969	0.973	0.985	0.984	0.972	0.967	0.971	0.973
	LPIPS ↓	0.052	0.049	0.052	0.028	0.031	0.047	0.045	0.049	0.044
	ATE RMSE ↓	**0.15**	**0.16**	0.11	0.18	**0.12**	**0.16**	0.19	0.21	**0.16**
Limited to 30 FPS	PSNR [dB] ↑	35.37	37.80	38.50	43.13	43.26	36.93	36.90	38.75	38.83
	SSIM ↑	0.963	0.971	0.975	**0.986**	**0.985**	0.974	0.969	0.973	0.975
	LPIPS ↓	0.048	0.045	0.048	0.026	0.029	0.043	0.042	0.045	0.041
	ATE RMSE ↓	**0.15**	**0.16**	0.11	0.18	**0.12**	0.17	**0.16**	0.21	**0.16**
Limited to 20 FPS	PSNR [dB] ↑	35.92	38.34	39.03	43.36	43.68	37.24	37.21	39.17	39.24
	SSIM ↑	0.966	0.973	0.976	**0.986**	**0.985**	0.975	0.970	0.974	**0.976**
	LPIPS ↓	0.045	0.042	0.044	0.024	0.028	0.040	0.038	0.041	0.038
	ATE RMSE ↓	**0.15**	**0.16**	**0.10**	0.19	**0.12**	0.17	0.18	0.21	**0.16**
Limited to 15 FPS	PSNR [dB] ↑	36.11	**38.41**	**39.18**	43.43	43.73	**37.52**	**37.37**	**39.41**	**39.40**
	SSIM ↑	0.967	**0.974**	**0.977**	**0.986**	**0.985**	**0.976**	0.971	0.974	**0.976**
	LPIPS ↓	0.043	**0.040**	**0.042**	0.024	0.028	**0.037**	**0.037**	**0.040**	**0.036**
	ATE RMSE ↓	**0.15**	**0.16**	0.11	0.19	0.13	0.17	0.18	0.21	**0.16**
Limited to 10 FPS	PSNR [dB] ↑	**36.47**	38.29	39.13	**43.52**	**43.81**	37.30	37.31	39.31	39.39
	SSIM ↑	**0.969**	**0.974**	**0.977**	**0.986**	**0.985**	0.974	0.971	**0.975**	**0.976**
	LPIPS ↓	**0.040**	0.041	**0.042**	**0.023**	**0.027**	0.043	0.038	0.041	0.037
	ATE RMSE ↓	0.16	**0.16**	0.11	0.20	0.13	0.17	0.18	0.21	0.17
Limited to 5 FPS	PSNR [dB] ↑	36.39	38.16	38.71	43.20	43.64	35.43	34.94	39.16	38.70
	SSIM ↑	**0.969**	**0.974**	0.976	0.985	**0.985**	0.973	**0.973**	0.974	**0.976**
	LPIPS ↓	0.043	**0.040**	0.045	0.030	0.028	0.047	0.051	0.046	0.041
	ATE RMSE ↓	0.16	0.18	0.11	0.18	0.16	0.18	0.19	0.21	0.17

Fig. 4: Novel View Rendering Results on Replica room2.

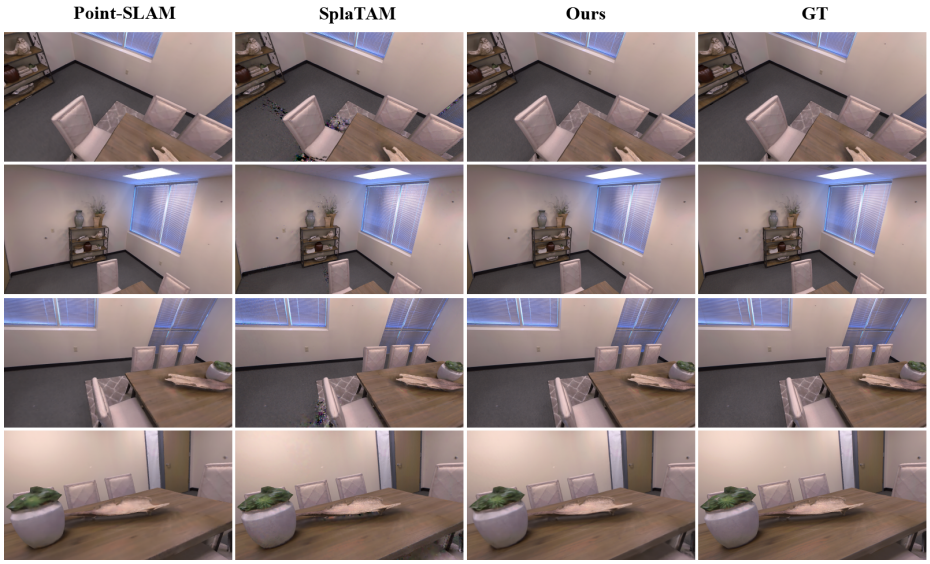

Fig. 5: Rendering Comparison on Replica room2.

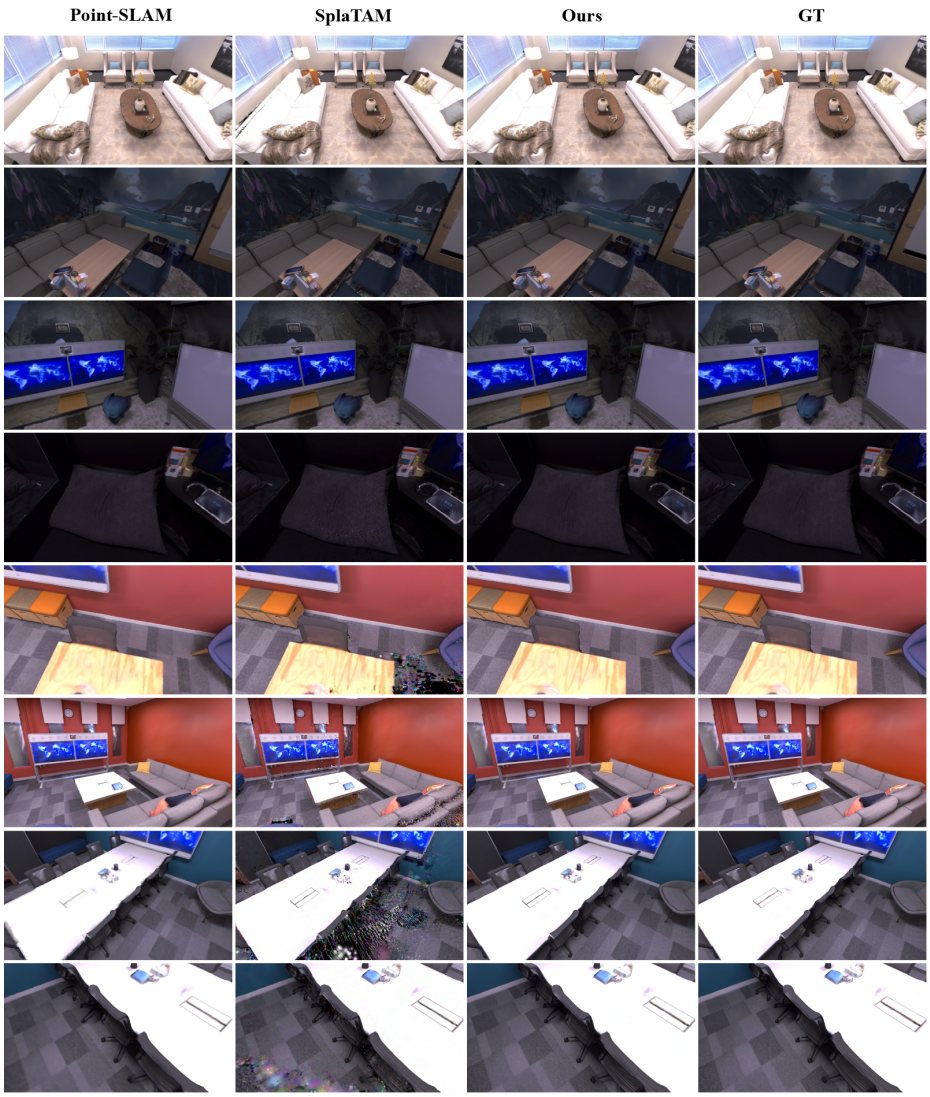

Fig. 6: Rendering Comparison on Replica.

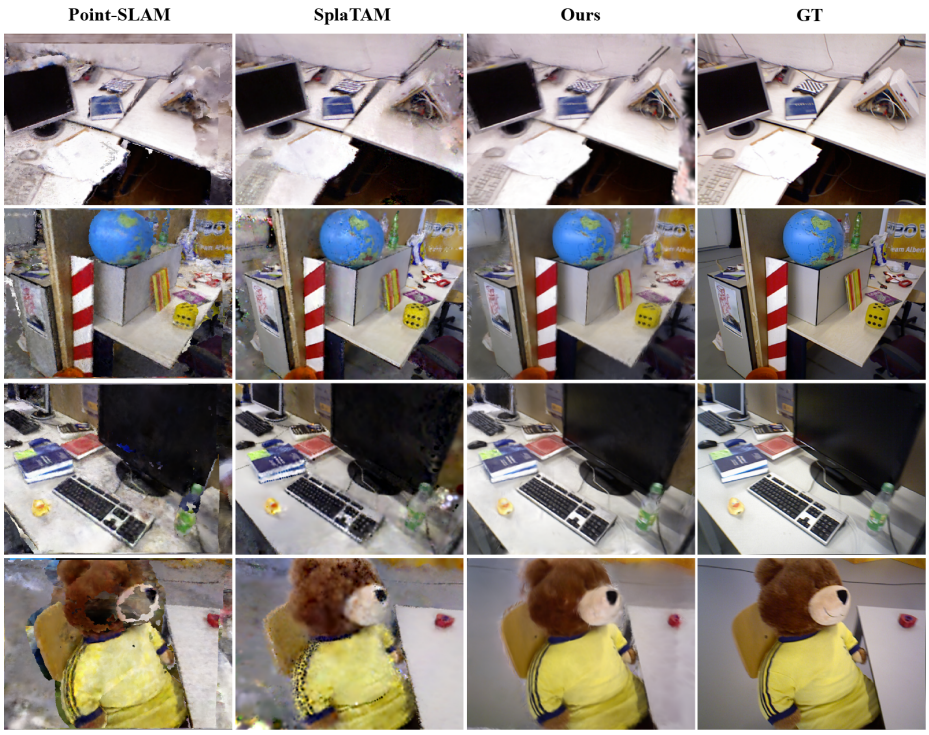

Fig. 7: Rendering Comparison on TUM-RGBD.

References

1. Campos, C., Elvira, R., Rodríguez, J.J.G., Montiel, J.M., Tardós, J.D.: Orb-slam3: An accurate open-source library for visual, visual–inertial, and multimap slam. IEEE Transactions on Robotics **37**(6), 1874–1890 (2021)
2. Chung, C.M., Tseng, Y.C., Hsu, Y.C., Shi, X.Q., Hua, Y.H., Yeh, J.F., Chen, W.C., Chen, Y.T., Hsu, W.H.: Orbeez-slam: A real-time monocular visual slam with orb features and nerf-realized mapping. In: 2023 IEEE International Conference on Robotics and Automation (ICRA). pp. 9400–9406. IEEE (2023)
3. Huang, H., Li, L., Cheng, H., Yeung, S.K.: Photo-slam: Real-time simultaneous localization and photorealistic mapping for monocular, stereo, and rgb-d cameras. arXiv preprint arXiv:2311.16728 (2023)
4. Keetha, N., Karhade, J., Jatavallabhula, K.M., Yang, G., Scherer, S., Ramanan, D., Luiten, J.: Splatam: Splat, track & map 3d gaussians for dense rgb-d slam. arXiv preprint arXiv:2312.02126 (2023)
5. Kerbl, B., Kopanas, G., Leimkühler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. ACM Transactions on Graphics **42**(4) (2023)
6. Koide, K., Yokozuka, M., Oishi, S., Banno, A.: Voxelized gicp for fast and accurate 3d point cloud registration. In: 2021 IEEE International Conference on Robotics and Automation (ICRA). pp. 11054–11059. IEEE (2021)
7. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. Communications of the ACM **65**(1), 99–106 (2021)
8. Sandström, E., Li, Y., Van Gool, L., Oswald, M.R.: Point-slam: Dense neural point cloud-based slam. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 18433–18444 (2023)
9. Segal, A., Haehnel, D., Thrun, S.: Generalized-icp. In: Robotics: science and systems. vol. 2, p. 435. Seattle, WA (2009)
10. Straub, J., Whelan, T., Ma, L., Chen, Y., Wijmans, E., Green, S., Engel, J.J., Mur-Artal, R., Ren, C., Verma, S., et al.: The replica dataset: A digital replica of indoor spaces. arXiv preprint arXiv:1906.05797 (2019)
11. Sturm, J., Engelhard, N., Endres, F., Burgard, W., Cremers, D.: A benchmark for the evaluation of rgb-d slam systems. In: 2012 IEEE/RSJ international conference on intelligent robots and systems. pp. 573–580. IEEE (2012)
12. Yan, C., Qu, D., Wang, D., Xu, D., Wang, Z., Zhao, B., Li, X.: Gs-slam: Dense visual slam with 3d gaussian splatting. arXiv preprint arXiv:2311.11700 (2023)
13. Zhu, Z., Peng, S., Larsson, V., Xu, W., Bao, H., Cui, Z., Oswald, M.R., Pollefeys, M.: Nice-slam: Neural implicit scalable encoding for slam. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 12786–12796 (2022)