GS-SLAM: Dense Visual SLAM with 3D Gaussian Splatting

Supplementary Material

Overview

This supplementary material complements our primary study, offering extended details and data to enhance the reproducibility of our GS-SLAM. It also includes supplementary evaluations and a range of qualitative findings that further support the conclusions presented in the main paper:

▷ ***Sec. 1****: Proofs of gradient computation, including the details of pose gradient derivation and depth gradient derivation.*

▷ ***Sec. 2****: Coarse-to-fine pose optimization algorithm, including pseudocode and visualization of how our algorithm worked.*

▷ ***Sec. 3****: Additional performance comparison, including the additional visualization on the Replica and TUM-RGBD dataset.*

▷ ***Sec. 4****: Additional ablation result, including the addition qualitative result on render, tracking and mapping performance.*

▷ ***Sec. 5****: Additional implementation details, including more details for reproducibility of our GS-SLAM.*

1. Proofs of Gradient Computation

To derive gradients for pose $\mathbf{P} = \{\mathbf{R}, \mathbf{t}\}$, recall that we store the pose in separate quaternion $\mathbf{q} \in \mathbb{R}^4$ and translation $\mathbf{t} \in \mathbb{R}^3$ vectors. To derive gradients for rotation, we recall the conversion from a unit quaternion $\mathbf{q} = [q_r, q_i, q_j, q_k]^T$ to rotation matrix $\mathbf{R}$:

$$\mathbf{R} = 2\begin{bmatrix} \frac{1}{2} - (q_j^2 + q_k^2) & (q_i q_j - q_r q_k) & (q_i q_k + q_r q_j) \\ (q_i q_j + q_r q_k) & \frac{1}{2} - (q_i^2 + q_k^2) & (q_j q_k - q_r q_i) \\ (q_i q_k - q_r q_j) & (q_j q_k + q_r q_i) & \frac{1}{2} - (q_i^2 + q_j^2) \end{bmatrix} \quad (14)$$

Also, we recall that the pose $\mathbf{P}$ is consisits of two part $\frac{\partial \mathbf{m}_i}{\partial \mathbf{P}}$ and $\frac{\partial \mathbf{\Sigma}'}{\partial \mathbf{P}}$ when ignore view-dependent color. For simplicity of the formula, we denote $\mathbf{X}_i = [x, y, z]^T$ in the camera coordinate as $\mathbf{X}_i^c = \mathbf{P}\mathbf{X}_i = [x^c, y^c, z^c]^T$.

Pose gradients back-propagation by $\frac{\partial \mathbf{m}_i}{\partial \mathbf{P}}$. We find the following gradients for translation $\mathbf{t}$ and $\frac{\partial \mathbf{X}_i^c}{\partial t}$:

$$\frac{\partial \mathbf{m}_i}{\partial \mathbf{t}} = \frac{\partial \mathbf{m}_i}{\partial \mathbf{X}_i^c} = \begin{bmatrix} \frac{f_x}{z^c} & 0 & -\frac{f_x x^c}{z^c} \\ 0 & \frac{f_y}{z^c} & -\frac{f_y y^c}{(z^c)^2} \end{bmatrix} \quad (15)$$

The gradients for rotation quaternion $\mathbf{q}$ is as follow:

$$\begin{aligned} \frac{\partial \mathbf{X}_i^c}{\partial q_r} &= 2\begin{bmatrix} 0 & -q_k y & q_j z \\ q_k x & 0 & -q_i z \\ -q_j x & q_i y & 0 \end{bmatrix}, \quad \frac{\partial \mathbf{X}_i^c}{\partial q_i} = 2\begin{bmatrix} 0 & q_j y & q_k z \\ q_j x & -2q_i y & -q_r z \\ -q_k x & q_r y & -2q_i z \end{bmatrix}, \\ \frac{\partial \mathbf{X}_i^c}{\partial q_j} &= 2\begin{bmatrix} -2q_j x & q_i y & q_r z \\ q_i x & 0 & q_k z \\ -q_r x & q_k y & -2q_j z \end{bmatrix}, \quad \frac{\partial \mathbf{X}_i^c}{\partial q_k} = 2\begin{bmatrix} -2q_k x & -q_r y & q_i z \\ q_r x & -2q_k y & q_j z \\ q_i x & q_j y & 0 \end{bmatrix} \end{aligned} \quad (16)$$

Pose gradients back-propagation by $\frac{\partial \mathbf{\Sigma}'}{\partial \mathbf{P}}$. We first derive the gradients for $\mathbf{E} = \mathbf{J}\mathbf{P}^{-1} = [e_{00}, e_{01}, e_{02}; e_{10}, e_{11}, e_{12}]$, where $\mathbf{J} = \frac{\partial \mathbf{m}_i}{\partial \mathbf{X}^c}$ is the Jacobian of $\mathbf{m}_i$ w.r.t. $\mathbf{X}^c$:

$$\frac{\partial \mathbf{\Sigma}'}{\partial \mathbf{P}} = \frac{\partial(\mathbf{J}\mathbf{P}^{-1}\mathbf{\Sigma}\mathbf{P}^{-T}\mathbf{J}^T)}{\partial \mathbf{P}} \quad (17)$$

Then, we back-propagate the gradient to $\mathbf{E}$:

$$\begin{aligned} \frac{\partial \text{vec}(\mathbf{\Sigma}')}{\partial e_{00}} &= \begin{bmatrix} 2e_{00}\Sigma_{00} + e_{01}\Sigma_{01} + e_{01}\Sigma_{10} + e_{02}\Sigma_{02} + e_{02}\Sigma_{20} \\ e_{10}\Sigma_{00} + e_{11}\Sigma_{01} + e_{12}\Sigma_{02} \\ e_{10}\Sigma_{00} + e_{11}\Sigma_{10} + e_{12}\Sigma_{20} \\ 0 \end{bmatrix}, \\ \frac{\partial \text{vec}(\mathbf{\Sigma}')}{\partial e_{01}} &= \begin{bmatrix} e_{00}\Sigma_{01} + e_{00}\Sigma_{10} + 2e_{01}\Sigma_{11} + e_{02}\Sigma_{12} + e_{02}\Sigma_{21} \\ e_{10}\Sigma_{10} + e_{11}\Sigma_{11} + e_{12}\Sigma_{12} \\ e_{10}\Sigma_{01} + e_{11}\Sigma_{11} + e_{12}\Sigma_{21} \\ 0 \end{bmatrix}, \\ \frac{\partial \text{vec}(\mathbf{\Sigma}')}{\partial e_{02}} &= \begin{bmatrix} e_{00}\Sigma_{02} + e_{00}\Sigma_{20} + e_{01}\Sigma_{12} + e_{01}\Sigma_{21} + 2e_{02}\Sigma_{22} \\ e_{10}\Sigma_{20} + e_{11}\Sigma_{21} + e_{12}\Sigma_{22} \\ e_{10}\Sigma_{02} + e_{11}\Sigma_{12} + e_{12}\Sigma_{22} \\ 0 \end{bmatrix}, \\ \frac{\partial \text{vec}(\mathbf{\Sigma}')}{\partial e_{10}} &= \begin{bmatrix} 0 \\ e_{00}\Sigma_{00} + e_{01}\Sigma_{10} + e_{02}\Sigma_{20} \\ e_{00}\Sigma_{00} + e_{01}\Sigma_{01} + e_{02}\Sigma_{02} \\ 2e_{10}\Sigma_{00} + e_{11}\Sigma_{01} + e_{11}\Sigma_{10} + e_{12}\Sigma_{02} + e_{12}\Sigma_{20} \end{bmatrix}, \\ \frac{\partial \text{vec}(\mathbf{\Sigma}')}{\partial e_{11}} &= \begin{bmatrix} 0 \\ e_{00}\Sigma_{01} + e_{01}\Sigma_{11} + e_{02}\Sigma_{21} \\ e_{00}\Sigma_{10} + e_{01}\Sigma_{11} + e_{02}\Sigma_{12} \\ e_{10}\Sigma_{01} + e_{10}\Sigma_{10} + 2e_{11}\Sigma_{11} + e_{12}\Sigma_{12} + e_{12}\Sigma_{21} \end{bmatrix}, \\ \frac{\partial \text{vec}(\mathbf{\Sigma}')}{\partial e_{12}} &= \begin{bmatrix} 0 \\ e_{00}\Sigma_{02} + e_{01}\Sigma_{12} + e_{02}\Sigma_{22} \\ e_{00}\Sigma_{20} + e_{01}\Sigma_{21} + e_{02}\Sigma_{22} \\ e_{10}\Sigma_{02} + e_{10}\Sigma_{20} + e_{11}\Sigma_{12} + e_{11}\Sigma_{21} + 2e_{12}\Sigma_{22} \end{bmatrix} \end{aligned} \quad (18)$$

In the mapping process, oversized 3D Gaussians are controlled through the delete, split, and clone process, so the magnitude of the covariance $\mathbf{\Sigma}$ is small enough. When back-propagate the gradients to pose $\mathbf{P}$, the intermediate term in Eq. (18) can be ignored.

Pose gradients back-propagation by depth supervision. Point-based depth alpha blending and color alpha blending share similarities; therefore, we implement the back-propagation of gradients for depth in the same manner for color:

$$\begin{aligned} \frac{\partial D}{\partial \alpha_i} &= d_i \prod_{j=1}^{i-1}(1-\alpha_j) - \sum_{k=i+1}^{n} d_k \alpha_k \prod_{j=1, j\neq i}^{k-1}(1-\alpha_j) \ , \\ \frac{\partial D}{\partial d_i} &= \alpha_i \prod_{j=1}^{i-1}(1-\alpha_j) \ , \end{aligned} \quad (19)$$

where n is the number of the 3D Gaussian splats that affect the pixel in the rasterization.

2. Coarse-to-fine Pose Optimization Algorithm

Our coarse-to-fine pose optimization algorithms are summarized in Algorithm 1.

Our algorithm is designed for splatting-based pose estimation, which uses α-blending on 3D Gaussians in strict

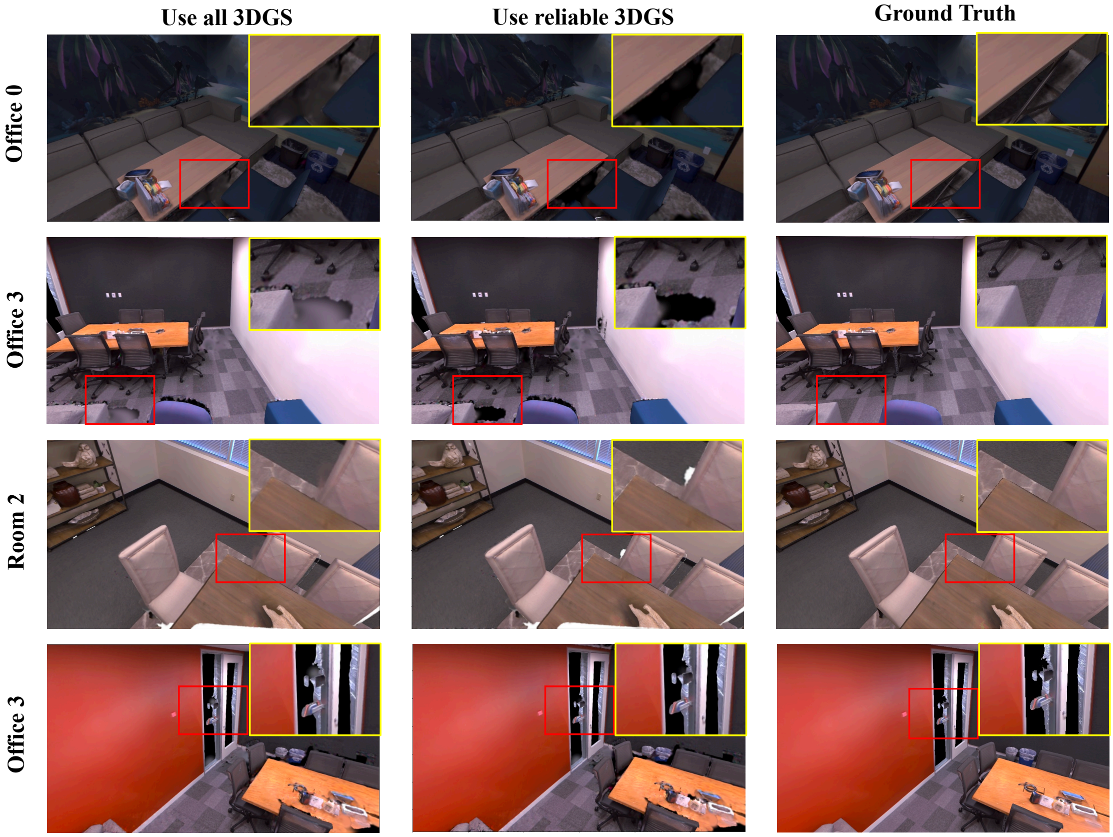

Figure 8. Visualization of the rasterized result before and after selecting reliable Gaussians. GS-SLAM coarse-to-fine strategy can effectively remove the unreliable 3D Gaussians to obtain more precise tracking results. In the enlarged figure, it can be seen that the abnormal optimized 3D Gaussians are removed. Note that the pixel without any 3D Gaussian will not affect the gradients of the pose. The background color in #office 0 and #office 3 is set to black, while in #room 0 is white.

front-to-back order rather than the volume rendering technique used by current NeRF-Based SLAM [11, 27, 35, 41, 48, 53, 55].

Visualization of the unreliable area removed in the fine stage is shown in Fig. 8. In the #Use all 3DGS column, we can see artifacts within the red bounding boxes resulting from improperly optimized positions in the past mapping process. These artifacts significantly impact the accuracy of pose estimation, as they introduce excessive optimization errors in the loss function. Our proposed reliable 3D Gaussians selection in coarse-to-fine strategy, as shown in the second column, filters the unreliable 3D Gaussians. The remaining background(black in #Office 0, #Office 3 and white in #Room 0) on the image plane would be ignored while back-propagating gradients to pose. Details about tracking and mapping settings can be seen in Sec. 5.

3. Additional Performance Comparison

Render performance on TUM-RGBD dataset. As shown in Fig. 11, we compare our method to current state-of-the-art NeRF-Based SLAM method CoSLAM [41], ES-LAM [11], Point-SLAM [27] and ground truth image. We

Algorithm 1 **Coarse-to-fine pose optimization**

W, H: width and height of the input images

```
$\mathbf{P}^{(0)} \leftarrow \mathbf{P}_{t-1}, \mathbf{P}_{t-2}$                ▷ Init Pose
$i_c, i_f \leftarrow 0$                ▷ Coarse and Fine Iteration Count
for $i_c < T_c$ do
    $\hat{I}_c \leftarrow$ Rasterize($\mathbf{G}$, $\mathbf{P}^{(i_c)}$, $\frac{1}{2}H$, $\frac{1}{2}W$)
                                        ▷ Coarse Render
    $L \leftarrow Loss(I_c, \hat{I}_c)$                ▷ Loss
    $\mathbf{P}^{(i_c+1)} \leftarrow$ Adam($\nabla L$)                ▷ Backprop & Step
    $i_c \leftarrow i_c + 1$
end for
$\mathbf{G}_{selected} \leftarrow$ ReliableGS($\mathbf{G}$, $\mathbf{P}^{(i_c+1)}, D, \varepsilon$)
                                        ▷ Select Reliable $\mathbf{G}$
for $i_f < T_f$ do
    $\hat{I}_f \leftarrow$ Rasterize($\mathbf{G}_{selected}$, $\mathbf{P}^{(i_c+i_f)}$, $H$, $W$)
                                        ▷ Fine Render
    $L \leftarrow Loss(I_f, \hat{I}_f)$                ▷ Loss
    $\mathbf{P}^{(i_c+i_f+1)} \leftarrow$ Adam($\nabla L$)                ▷ Backprop & Step
    $i_f \leftarrow i_f + 1$
end for
```

Table 9. Ablation study of tracking, mapping, and rendering performance on whole Replica dataset [31]. We present detailed ablation results on the entire Replica dataset to demonstrate the significant advantages of our proposed modules. Our method introduces a novel approach that combines coarse-to-fine pose estimation with an adaptive 3D Gaussian expansion strategy. This comprehensive methodology successfully increases the render, tracking, and mapping quality.

Method	Metric	Room 0	Room 1	Room 2	Office 0	Office 1	Office 2	Office 3	Office 4	Avg.
w/o delete in mapping	ATE [cm] ↓	0.58	0.63	0.57	0.76	**0.38**	0.61	0.62	0.87	0.63
	Depth L1 [cm] ↓	1.68	1.17	1.92	1.30	1.46	1.82	1.79	1.83	1.62
	Precision [%]↑	53.55	68.76	50.58	67.03	77.52	59.45	53.37	49.03	59.91
	Recall [%]↑	49.32	61.88	45.57	61.60	65.67	50.15	46.09	42.80	52.89
	F1 [%]↑	51.35	65.14	47.94	64.20	71.10	54.41	49.46	45.70	56.16
	PSNR [dB] ↑	31.22	**33.33**	**33.58**	38.17	39.97	30.77	**32.04**	34.86	34.24
	SSIM ↑	0.967	**0.975**	**0.977**	0.984	0.990	0.974	0.969	0.980	0.977
	LPIPS ↓	**0.094**	**0.075**	**0.086**	0.053	0.046	0.096	**0.100**	**0.080**	0.079
Coarse in tracking	ATE [cm] ↓	0.91	0.87	0.52	0.71	0.65	0.56	0.50	0.71	0.68
	Depth L1 [cm] ↓	1.48	0.94	1.47	0.84	0.97	1.52	1.58	1.28	1.26
	Precision [%]↑	59.68	70.51	62.66	83.11	**87.79**	66.85	61.34	66.55	69.81
	Recall [%]↑	57.54	64.98	57.58	76.36	74.58	58.25	54.07	59.01	62.80
	F1 %]↑	56.50	67.63	60.01	79.59	80.22	62.25	57.48	62.55	65.78
	PSNR [dB] ↑	29.13	32.08	33.12	38.62	40.69	32.02	32.02	35.05	34.09
	SSIM ↑	0.954	0.970	0.971	**0.986**	**0.993**	**0.978**	0.967	0.980	0.975
	LPIPS ↓	0.120	0.085	0.093	0.051	0.037	0.097	0.117	0.089	0.086
Fine in tracking	ATE [cm] ↓	0.49	0.82	5.59	0.69	0.57	**0.55**	**0.40**	0.74	1.23
	Depth L1 [cm] ↓	1.39	0.91	5.84	0.87	1.26	1.49	1.55	1.27	1.82
	Precision [%]↑	62.61	73.71	16.45	81.63	72.86	69.08	61.88	67.05	63.16
	Recall [%] ↑	59.18	67.68	15.53	75.53	64.77	59.73	54.40	59.45	57.03
	F1 [%] ↑	61.29	70.57	15.98	78.46	68.57	64.06	57.90	63.02	59.98
	PSNR [dB] ↑	30.84	33.247	27.25	38.41	40.46	32.13	32.03	35.18	33.69
	SSIM ↑	0.964	**0.975**	0.901	0.985	0.992	**0.978**	0.966	0.980	0.968
	LPIPS ↓	0.096	0.076	0.188	0.052	0.037	0.095	0.119	0.089	0.094
Our full	ATE [cm] ↓	**0.48**	**0.53**	**0.33**	**0.52**	0.41	0.59	0.46	**0.70**	**0.50**
	Depth L1 [cm] ↓	**1.31**	**0.82**	**1.26**	**0.81**	**0.96**	**1.41**	**1.53**	**1.08**	**1.15**
	Precision [%]↑	**64.58**	**83.11**	**70.13**	**83.43**	87.77	**70.91**	**63.18**	**68.88**	**74.09**
	Recall [%]↑	**61.29**	**76.83**	**63.84**	**76.90**	**76.15**	**61.63**	**62.91**	**61.50**	**67.63**
	F1 [%]↑	**62.89**	**79.85**	**66.84**	**80.03**	**81.55**	**65.95**	**59.17**	**64.98**	**70.16**
	PSNR [dB] ↑	**31.56**	32.86	32.59	**38.70**	**41.17**	**32.36**	32.03	**35.23**	**34.56**
	SSIM ↑	**0.968**	0.973	0.971	**0.986**	**0.993**	**0.978**	**0.970**	**0.981**	**0.978**
	LPIPS ↓	**0.094**	**0.075**	0.093	**0.050**	**0.033**	**0.009**	0.110	0.088	**0.069**

showcase the render quality of different methods using the final reconstructed environment model, presented in descending order from top to bottom. Our GS-SLAM provides the clearest results, particularly evident in the complex #fr3_office scene, displaying a higher richness in detail information, indicating its superiority in handling details and edges.

Render performance on Replica dataset. As shown in Fig. 9, our GS-SLAM method performs superior in all tested scenarios. It delivers renderings with clear, well-defined edges and richly textured surfaces that closely align with the ground truth. This suggests that GS-SLAM not only captures the geometric detail with high precision but also accurately reconstructs the textural information of the scenes. The method's ability to render sharp images even in regions with complex textures and lighting conditions underscores its potential for accurate 3D environment mapping.

4. Additional Ablation Results

More detailed ablation experiments. In Tab. 9, we present our ablation study that demonstrates the effectiveness of our proposed approach on the comprehensive Replica dataset. The experiment contrasts different module arrangements, including disabling our Gaussian delete method in mapping, using only coarse images in tracking, using fine images in tracking, and using our fully integrated method. The results distinctly highlight the superiority of our complete methodology, particularly evident in improved metrics across the board, from ATE to LPIPS. This confirms the benefits of our coarse-to-fine pose estimation and adaptive 3D Gaussian expansion strategy.

Table 10. Ablation of the depth supervision on Replica #Room0.

Setting	#Room0							
	ATE↓	Depth L1↓	Precision↑	Recall ↑	F1↑	PSNR↑	SSIM↑	LPIPS↓
w/o Depth	0.80	3.21	14.28	15.01	14.63	29.76	0.956	0.107
w/ Depth	**0.48**	**1.31**	**64.58**	**61.29**	**62.89**	**31.56**	**0.968**	**0.094**

Effect of adaptive 3D Gaussian expansion strategy. Fig. 6 demonstrates the reconstructed mesh with and without our adaptive 3D Gaussian expansion strategy. The comparison is based on the Replica dataset Room0 subsequence. The left side displays a more coherent surface mesh due to our expansion strategy, while the right side lacks this delete strategy and results in less accuracy in reconstruction.

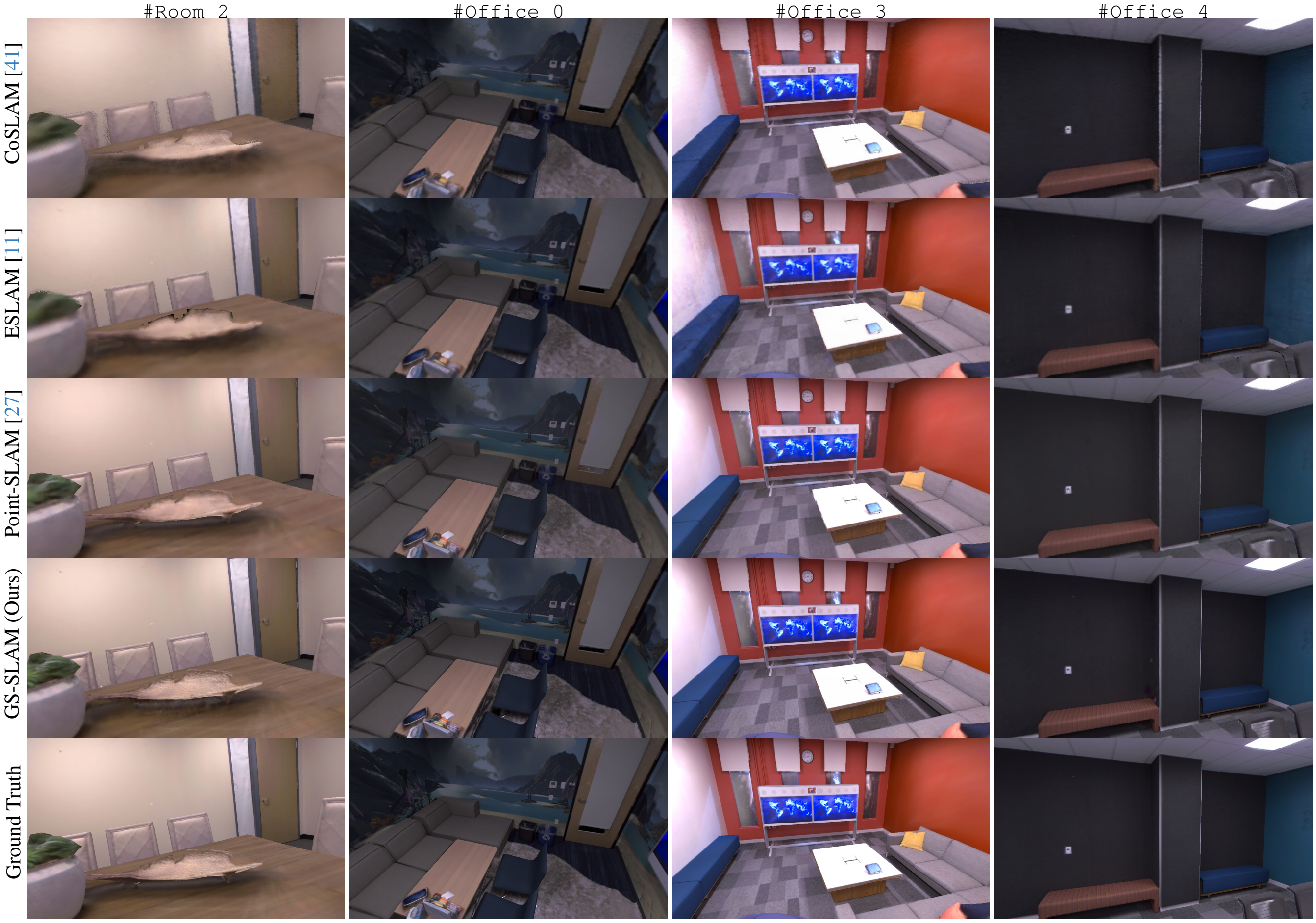

Figure 9. Additional render result comparison on Replica [31]. Our GS-SLAM can achieve more clear edges and better results in regions with rich texture than the previous SOTA methods.

Effect of depth supervision. Tab. 10 illustrates quantitative evaluation using depth supervision in mapping. In contrast to the original color-only supervision in [13], the depth supervision can significantly improve the tracking and mapping performance by providing accurate geometry constraints for the optimization. Our implementation with depth achieves better tracking ATE of 0.48, mapping precision of 64.58, and rendering PSNR of 31.56 compared with the implementation without depth supervision.

5. Additional Implementation Details

Mapping hyper-parameters. The 3D Gaussian representation and pose are trained using Adam optimizer with initialized position learning rate $lr_{\mathbf{X}_{init}} = 1.6e^{-5}$, final position learning rate $lr_{\mathbf{X}_{final}} = 1.7e^{-7}$, max attenuation steps 100. Other Spherical Harmonics coefficients learning rate $lr_{\mathbf{Y}}$, opacity learning rate lr_{Λ}, scaling learning rate $lr_{\mathbf{S}}$ and rotation learning rate $lr_{\mathbf{R}}$ set to $5e^{-4}$, $1e^{-2}$, $2e^{-4}$, $4e^{-5}$ respectively. In all experiments, we set the photometric loss weighting 0.8, geometric loss weighting 0.3, and keyframe window size $K = 10$. In the mapping process, we densify the 3D Gaussians every 10 iterations before the first 70 iterations in a total of 100 iterations. And the scale and grad threshold for clone or split is set to $0.02m$ and 0.002. For the stability of the optimization, first-order coefficients of spherical harmonics coefficients are only optimized in bundle adjustment. Note that we only optimize the camera pose in the latter half of the iterations due to the adverse impact of improper 3D Gaussians on optimization. Despite this, there are still negative optimizations for camera poses at some point. In addition, in all TUM-RGBD sequences and Replica `Office` subsequence, we set the background color to black, while in other `Room` subsequence, we set the background to white. In the Replica dataset, we use 10 iterations for tracking and 100 iterations for mapping with max keyframe interval $\mu_k = 30$, while in the challenging TUM RGB-D dataset, we use 30 iterations for tracking, with max keyframe interval $\mu_k = 5$.

Mapping mesh comparison method. We follow Point-SLAM and use TSDF-Fusion [4] to generate mesh from predicted pose and depth. We also evaluate map rendering quality in Sec. 4.3. That's because there is no direct way to get surface or mesh in 3DGS-based SLAM, as they do not represent scenes with density fields and can not directly generate mesh via marching cube. GS-SLAM achieves comparable map reconstruction results, better tracking accuracy, and higher FPS than Point-SLAM. Despite this, we explored generating mesh from 3DGS centers and Gaussian marching cube [36] in Fig. 10, but the results are not satisfactory.

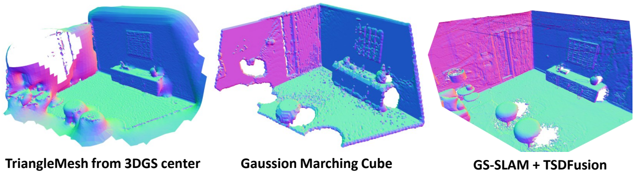

Figure 10. Generate mesh from 3DGS with different methods.

Tracking hyper-parameters. The pose is trained using FusedAdam optimizer with learning rate $lr_{\mathbf{t}} = 2e^{-4}$, $lr_{\mathbf{q}} = 5e^{-4}$, and photometric loss weighting 0.8, in the first five iterations we do the coarse pose estimation, while in the later iterations use the reliable 3D Gaussians to do the fine pose estimation. In addition, to exclude the pixel without proper color optimization. If the loss on the pixel is more than ten times the median loss, the pixel will be ignored.

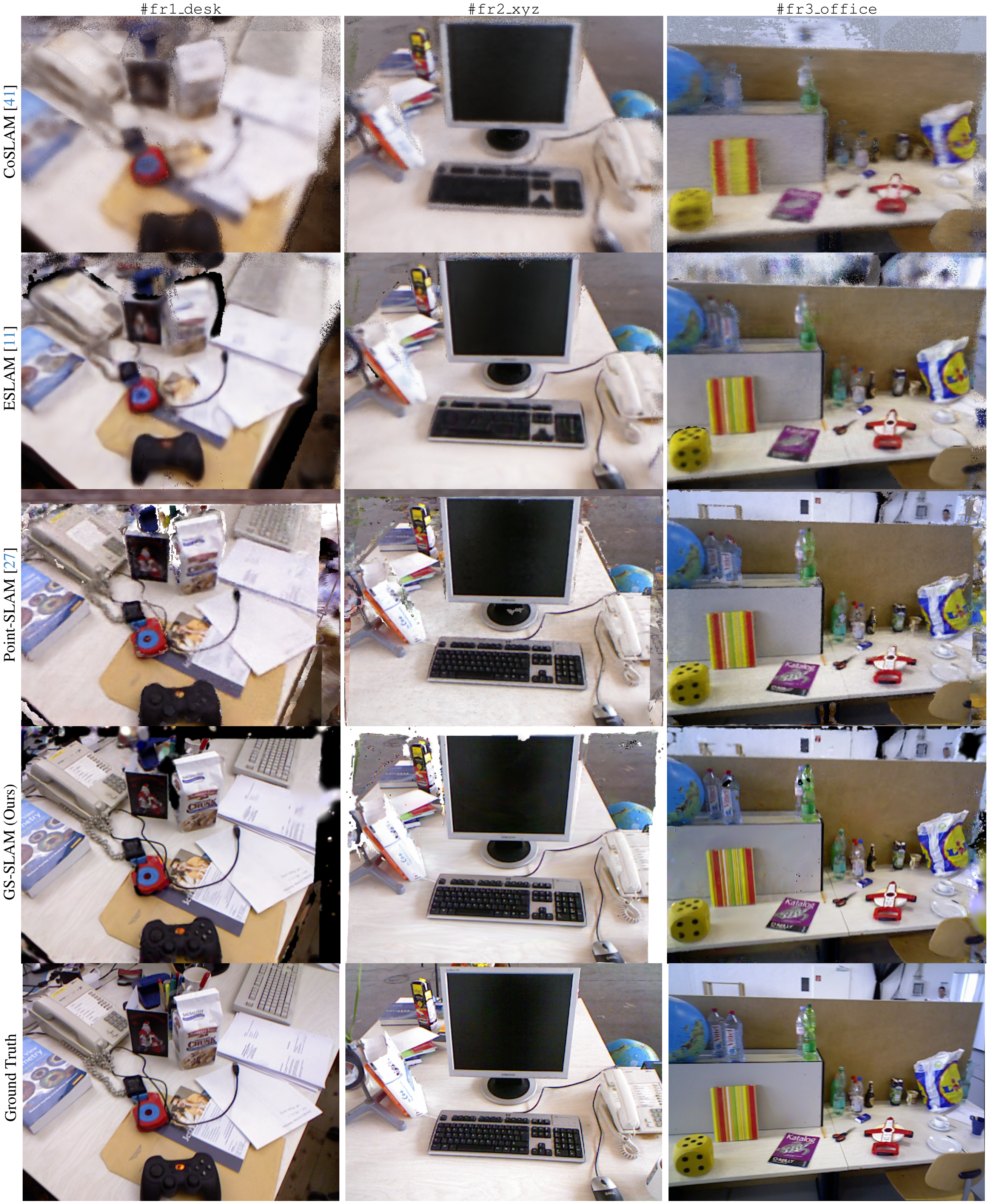

Figure 11. Additional render result comparison on TUM-RGBD [8]. Thanks to fast back-propagation of splatting in optimized 3D Gaussians, our GS-SLAM can reconstruct dense environment maps with richness and intricate details.

References

[1] Michael Bloesch, Jan Czarnowski, Ronald Clark, Stefan Leutenegger, and Andrew J. Davison. Codeslam - learning a compact, optimisable representation for dense visual slam. *CVPR*, pages 2560–2568, 2018. 2

[2] Guillaume Bresson, Zayed Alsayed, Li Yu, and Sébastien Glaser. Simultaneous localization and mapping: A survey of current trends in autonomous driving. *TIV*, 2:194–220, 2017. 6

[3] Zilong Chen, Feng Wang, and Huaping Liu. Text-to-3d using gaussian splatting. *ArXiv*, abs/2309.16585, 2023. 2

[4] Brian Curless and Marc Levoy. Volumetric method for building complex models from range images. In *SIGGRAPH*. ACM, 1996. 4

[5] Parth Rajesh Desai, Pooja Nikhil Desai, Komal Deepak Ajmera, and Khushbu Mehta. A review paper on oculus rift-a virtual reality headset. *ArXiv*, abs/1408.1173, 2014. 6

[6] Hugh F. Durrant-Whyte and Tim Bailey. Simultaneous localization and mapping: part i. *RAM*, 13:99–110, 2006. 1

[7] Christian Häne, Christopher Zach, Jongwoo Lim, Ananth Ranganathan, and Marc Pollefeys. Stereo depth map fusion for robot navigation. *IROS*, pages 1618–1625, 2011. 6

[8] Caner Hazirbas, Andreas Wiedemann, Robert Maier, Laura Leal-Taixé, and Daniel Cremers. Tum rgb-d scribble-based segmentation benchmark. https://github.com/tum-vision/rgbd_scribble_benchmark, 2018. 6

[9] Jiahui Huang, Shi-Sheng Huang, Haoxuan Song, and Shi-Min Hu. Di-fusion: Online implicit 3d reconstruction with deep priors. In *CVPR*, pages 8932–8941, 2021. 6

[10] Xudong Jiang, Lifeng Zhu, Jia Liu, and Aiguo Song. A slam-based 6dof controller with smooth auto-calibration for virtual reality. *TVC*, 39:3873 – 3886, 2022. 1

[11] Mohammad Mahdi Johari, Camilla Carta, and Franccois Fleuret. Eslam: Efficient dense slam system based on hybrid representation of signed distance fields. *CVPR*, 2023. 1, 2, 5, 6, 7, 8, 4

[12] Olaf Kähler, Victor Adrian Prisacariu, Julien P. C. Valentin, and David William Murray. Hierarchical voxel block hashing for efficient integration of depth images. *RAL*, 1:192–197, 2016. 1

[13] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *TOG*, 42(4), 2023. 2, 3, 7, 4

[14] Leonid Keselman and Martial Hebert. Approximate differentiable rendering with algebraic surfaces. In *ECCV*, 2022. 2

[15] Leonid Keselman and Martial Hebert. Flexible techniques for differentiable rendering with 3d gaussians. *arXiv preprint arXiv:2308.14737*, 2023. 2

[16] Georg S. W. Klein and David William Murray. Parallel tracking and mapping on a camera phone. *ISMAR*, pages 83–86, 2009. 2

[17] Jonathon Luiten, Georgios Kopanas, Bastian Leibe, and Deva Ramanan. Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. *ArXiv*, abs/2308.09713, 2023. 2

[18] Robert Maier, Raphael Schaller, and Daniel Cremers. Efficient online surface correction for real-time large-scale 3d reconstruction. *ArXiv*, abs/1709.03763, 2017. 1

[19] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020. 1

[20] Raul Mur-Artal and Juan D. Tardós. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *TRO*, 33:1255–1262, 2016. 1, 6

[21] Richard A. Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J. Davison, Pushmeet Kohli, Jamie Shotton, Steve Hodges, and Andrew William Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. *ISMAR*, pages 127–136, 2011. 1

[22] Richard A. Newcombe, S. Lovegrove, and Andrew J. Davison. Dtam: Dense tracking and mapping in real-time. *ICCV*, pages 2320–2327, 2011. 2

[23] Matthias Nießner, Michael Zollhöfer, Shahram Izadi, and Marc Stamminger. Real-time 3d reconstruction at scale using voxel hashing. *TOG*, 32:1 – 11, 2013. 1

[24] Delin Qu, Chi Yan, Dong Wang, Jie Yin, Dan Xu, Bin Zhao, and Xuelong Li. Implicit event-rgbd neural slam. *CVPR*, 2024. 1

[25] Gerhard Reitmayr, Tobias Langlotz, Daniel Wagner, Alessandro Mulloni, Gerhard Schall, Dieter Schmalstieg, and Qi Pan. Simultaneous localization and mapping for augmented reality. *ISUVR*, pages 5–8, 2010. 1

[26] Fabio Ruetz, Emili Hernández, Mark Pfeiffer, Helen Oleynikova, Mark Cox, Thomas Lowe, and Paulo Vinicius Koerich Borges. Ovpc mesh: 3d free-space representation for local ground vehicle navigation. *ICRA*, pages 8648–8654, 2018. 1

[27] Erik Sandström, Yue Li, Luc Van Gool, and Martin R. Oswald. Point-slam: Dense neural point cloud-based slam. In *ICCV*, 2023. 2, 5, 6, 7, 4

[28] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, 2016. 3

[29] Thomas Schöps, Torsten Sattler, and Marc Pollefeys. Bad slam: Bundle adjusted direct rgb-d slam. *CVPR*, pages 134–144, 2019. 2

[30] Thomas Schops, Torsten Sattler, and Marc Pollefeys. BAD SLAM: Bundle adjusted direct RGB-D SLAM. In *CVPR*, 2019. 6

[31] Julian Straub, Thomas Whelan, Lingni Ma, Yufan Chen, Erik Wijmans, Simon Green, Jakob J. Engel, Raul Mur-Artal, Carl Yuheng Ren, Shobhit Verma, Anton Clarkson, Ming Yan, Brian Budge, Yajie Yan, Xiaqing Pan, June Yon, Yuyang Zou, Kimberly Leon, Nigel Carter, Jesus Briales, Tyler Gillingham, Elias Mueggler, Luis Pesqueira, Manolis Savva, Dhruv Batra, Hauke Malte Strasdat, Renzo De Nardi, Michael Goesele, S. Lovegrove, and Richard A. Newcombe. The replica dataset: A digital replica of indoor spaces. *ArXiv*, abs/1906.05797, 2019. 5, 3, 4

[32] J. Stückler and Sven Behnke. Multi-resolution surfel maps for efficient dense 3d modeling and tracking. *JVCIR*, 25: 137–147, 2014. 1

[33] Jürgen Sturm, Nikolas Engelhard, Felix Endres, Wolfram Burgard, and Daniel Cremers. A benchmark for the evaluation of RGB-D SLAM systems. In *IROS*. IEEE/RSJ, 2012. 5, 6

[34] Edgar Sucar, Kentaro Wada, and Andrew J. Davison. Nodeslam: Neural object descriptors for multi-view shape reconstruction. *3DV*, pages 949–958, 2020. 2

[35] Edgar Sucar, Shikun Liu, Joseph Ortiz, and Andrew J. Davison. imap: Implicit mapping and positioning in real-time. *ICCV*, 2021. 1, 2, 6

[36] Jiaxiang Tang, Jiawei Ren, Hang Zhou, Ziwei Liu, and Gang Zeng. Dreamgaussian: Generative gaussian splatting for efficient 3d content creation. *ArXiv*, abs/2309.16653, 2023. 2, 5

[37] Zachary Teed and Jia Deng. Droid-slam: Deep visual slam for monocular, stereo, and rgb-d cameras. In *NIPS*, 2021. 2

[38] Andreas Langeland Teigen, Yeonsoo Park, Annette Stahl, and Rudolf Mester. Rgb-d mapping and tracking in a plenoxel radiance field. *ArXiv*, abs/2307.03404, 2023. 2

[39] Charalambos Theodorou, Vladan Velisavljevic, Vladimir Dyo, and Fredi Nonyelu. Visual slam algorithms and their application for ar, mapping, localization and wayfinding. *Array*, 15:100222, 2022. 1

[40] Angtian Wang, Peng Wang, Jian Sun, Adam Kortylewski, and Alan Yuille. Voge: a differentiable volume renderer using gaussian ellipsoids for analysis-by-synthesis. *arXiv preprint arXiv:2205.15401*, 2022. 2

[41] Hengyi Wang, Jingwen Wang, and Lourdes de Agapito. Co-slam: Joint coordinate and sparse parametric encodings for neural real-time slam. *CVPR*, 2023. 1, 5, 6, 8, 2, 4

[42] Kaixuan Wang, Fei Gao, and Shaojie Shen. Real-time scalable dense surfel mapping. *ICRA*, pages 6919–6925, 2019. 1

[43] Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. Image quality assessment: from error visibility to structural similarity. *TIP*, 13(4):600–612, 2004. 5

[44] Thomas Whelan, Michael Kaess, Maurice F. Fallon, Hordur Johannsson, John J. Leonard, and John B. McDonald. Kintinuous: Spatially extended kinectfusion. In *AAAI*, 2012. 2

[45] Thomas Whelan, John McDonald, Michael Kaess, Maurice Fallon, Hordur Johannsson, and John J. Leonard. Kintinuous: Spatially extended kinectfusion. In *RSS Workshop on RGB-D*, 2012. 6

[46] Thomas Whelan, Stefan Leutenegger, Renato Salas-Moreno, Ben Glocker, and Andrew Davison. Elasticfusion: Dense slam without a pose graph. In *RSS*, 2015. 1, 6

[47] Guanjun Wu, Taoran Yi, Jiemin Fang, Lingxi Xie, Xiaopeng Zhang, Wei Wei, Wenyu Liu, Qi Tian, and Xinggang Wang. 4d gaussian splatting for real-time dynamic scene rendering. *ArXiv*, abs/2310.08528, 2023. 2

[48] 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. *ISMAR*, pages 499–507, 2022. 2, 5, 6, 7, 8

[49] Ziyi Yang, Xinyu Gao, Wenming Zhou, Shaohui Jiao, Yuqing Zhang, and Xiaogang Jin. Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. *ArXiv*, abs/2309.13101, 2023. 2

[50] Zeyu Yang, Hongye Yang, Zijie Pan, Xiatian Zhu, and Li Zhang. Real-time photorealistic dynamic scene representation and rendering with 4d gaussian splatting. *ArXiv*, abs/2310.10642, 2023. 2

[51] Taoran Yi, Jiemin Fang, Guanjun Wu, Lingxi Xie, Xiaopeng Zhang, Wenyu Liu, Qi Tian, and Xinggang Wang. Gaussiandreamer: Fast generation from text to 3d gaussian splatting with point cloud priors. *ArXiv*, abs/2310.08529, 2023. 2

[52] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, pages 586–595, 2018. 5

[53] Youmin Zhang, Fabio Tosi, Stefano Mattoccia, and Matteo Poggi. Go-slam: Global optimization for consistent 3d instant reconstruction. In *ICCV*, 2023. 2

[54] Shuaifeng Zhi, Michael Bloesch, Stefan Leutenegger, and Andrew J. Davison. Scenecode: Monocular dense semantic reconstruction using learned encoded scene representations. *CVPR*, pages 11768–11777, 2019. 2

[55] 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. *CVPR*, 2021. 1, 2, 5, 6, 7, 8

[56] Wojciech Zielonka, Timur M. Bagautdinov, Shunsuke Saito, Michael Zollhofer, Justus Thies, and Javier Romero. Drivable 3d gaussian avatars. 2023. 2

[57] Matthias Zwicker, Hanspeter Pfister, Jeroen van Baar, and Markus H. Gross. Ewa volume splatting. *Proceedings Visualization, 2001. VIS '01.*, pages 29–538, 2001. 3