Supplementary Meterial of MonoGS++: Fast and Accurate Monocular RGB Gaussian SLAM

BMVC 2024 Submission # 133

1 Implementation Details

For visual odometry, we randomly sample 128 patches per frame. The initialization process utilizes the first 8 key frames, while the local optimization window is set to 15 frames. The most recent 3 frames are consistently used as keyframes, with the fourth-to-last keyframe being marginalized if the optical flow magnitude between the fifth-to-last keyframe and the third-to-last keyframe is less than 15 pixels.

In the context of 3D Gaussian mapping, the learning rate for the Gaussian center starts at 1e-4 and gradually decreases to 1e-6. The learning rates for opacity, scale, rotation, and features are set to 0.05, 0.001, 0.001, and 0.0025, respectively. For each scene in the Replica dataset [5], we perform a total of 10,000 optimization iterations. Densification begins after 500 iterations and continues until 9,000 iterations, occurring every 200 iterations. Following the 3D Gaussian Splatting methodology [2], opacity is reset every 3,000 iterations. The threshold τ for dynamic 3D Gaussian insertion is defined as the 25th percentile of the mean distances to each point's three nearest neighbors. In clarity-enhancing Gaussian densification, the split threshold is set at 0.00025 of the total image area.

2 Comparison with Baselines

We selected Point-SLAM [4], SplaTAM [1], and MonoGS [3] as our baselines. Using their open-source code, we reproduced PointSLAM [4], SplaTAM [1], and MonoGS [3], ensuring consistent hardware settings with our method. The results are presented in Table 1 and Table 2 of the main paper. Below, we outline the key differences between our approach and these existing methods.

Point-SLAM [4] diverges from previous dense neural SLAM methods that depend on feature grids (dense grid or hash grid). Instead, it decodes colors and occupancies from point clouds back-projected from input depth maps. In contrast, our approach is a 3D Gaussian Splatting (GS)-based method that does not utilize depth information.

Both SplaTAM [1] and MonoGS [3] are also 3D GS-based methods. However, they differ from our approach as they jointly optimize the camera poses and 3D Gaussians by minimizing rendering loss. Our method, on the other hand, employs a patch-based visual

© 2024. The copyright of this document resides with its authors.
It may be distributed unchanged freely in print or electronic forms.

odometry to estimate camera poses, which enhances efficiency and accuracy. Furthermore, SplaTAM [1] relies on depth sensors to initialize 3D Gaussians. Although MonoGS [3] can operate with monocular RGB input, our experiments demonstrate that our method surpasses it in both camera tracking accuracy and rendering quality. MonoGS lacks robustness across different modes, while our method maintains high performance and achieves consistent results with both monocular RGB and RGB-D inputs.

3 More Results

3.1 More ablation studies

In addition to the ablation studies presented in the main text, we conducted additional experiments to demonstrate the effectiveness of our proposed modules.

Effectiveness of Planar Regularization. We present the loss curves for variations with and without planar regularization term in Figure 1, running on Replica/room0. The loss values for the configurations incorporating planar regularization are consistently lower than those without, indicating that the planar regularization term enhances the convergence of the optimization process.

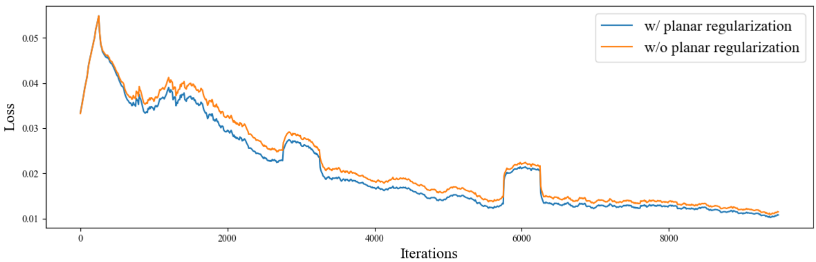

Figure 1: Loss curve of our method with and without planar regularization.

Effectiveness of Clarity-Enhancing Densification. As shown in Figure 2, we present a visual comparison of results with and without the Clarity-Enhancing Densification module. Without this module, the rendered image lacks detail and appears blurry, as highlighted by the blue and green rectangles. For a more detailed examination, please zoom in on the highlighted areas.

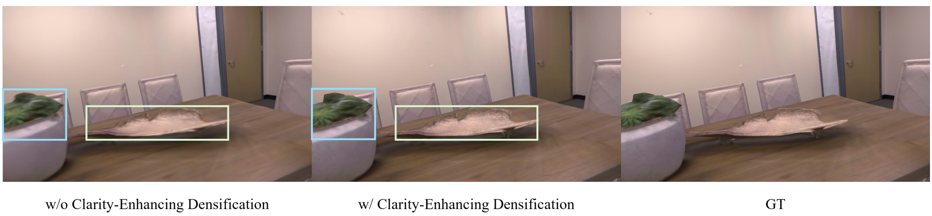

Figure 2: Visual comparison results of with and without Clarity-Enhancing Densification.

3.2 Memory Analysis

We report the peak values of GPU memory consumption in Table 1, comparing our method with the baselines SplaTAM [1] and MonoGS [3] when running on Replica/room0. SplaTAM and MonoGS require 11.34 GB and 14.96 GB of GPU memory, respectively, whereas our method consumes only 7.91 GB. This significantly lower GPU memory consumption demonstrates the memory efficiency of our approach compared to the baselines.

Method	SplaTAM [1]	MonoGS [3]	Ours
GPU Memory (MB)	11611	15315	8104

Table 1: **GPU memory consumption comparison.** We compare GPU memory consumption with baselines SplaTAM [1] and MonoGS [3] on Replica/room0.

3.3 Extended experiments using RGB-D as input

Our method is well compatible with scenarios using RGB-D as input and achieves better accuracy compared to using RGB input.

Method. Implementing a version of our method that utilizes RGB-D input is straightforward. Firstly, for the visual odometry component, we initialize the inverse depths of patches using the input depths instead of random sampling. Secondly, in the 3D Gaussian mapping process, in addition to incorporating new 3D Gaussians derived from patches optimized by visual odometry, we also add new 3D Gaussians from randomly downsampled points back-projected from input depth images every 50 frames. Furthermore, we introduce depth supervision for the optimization of the 3D Gaussian map, specifically by minimizing the difference between the rendered depth images and the input depth images, for the *i*-th frame, the depth loss term is defined as:

$$\mathcal{L}_{depth} = \|\hat{D}_i - D_i\|_1, \tag{1}$$

and the final objective function is changed to:

$$\mathcal{L} = \lambda_{color} \cdot \mathcal{L}_{color} + \lambda_{reg} \cdot \mathcal{L}_{reg} + \lambda_{depth} \cdot \mathcal{L}_{depth}, \tag{2}$$

where the λ_{depth} is the weight of $\mathcal{L}_{depth}$.

Experimental results. As shown in Table 2, we conducted experiments using both monocular RGB and RGB-D inputs on the Replica [5] and TUM-RGBD [6] datasets, each with three sequences. The numerical results reveal that MonoGS [3] performs better in RGB-D mode than in monocular RGB mode on the Replica dataset but performs worse on the TUM-RGBD dataset. This indicates a lack of robustness in MonoGS across different modes. In contrast, our method demonstrates greater robustness, achieving comparable results in both monocular RGB and RGB-D modes. Additionally, when using RGB-D as input, our method outperforms MonoGS in both rendering quality and tracking accuracy.

3.4 More Visualization Results

We show more visualization results in Figure 3 and Figure 4.

Method	Modality	Metric	room2	office2	office4	fr1/desk2	fr2/xyz	fr3/office
MonoGS	RGB	PSNR[dB]↑	31.82	27.01	27.29	14.06	22.06	23.02
		SSIM↑	0.92	0.88	0.90	0.50	0.72	0.78
		LPIPS↓	0.16	0.26	0.25	0.62	0.27	0.32
		ATE-MSE (cm)↓	6.53	20.89	43.85	79.45	4.31	1.85
MonoGS	RGB-D	PSNR[dB]↑	37.49	36.24	37.06	8.90	12.46	15.95
		SSIM↑	0.96	**0.96**	0.95	0.31	0.71	0.46
		LPIPS↓	0.075	**0.078**	0.099	0.71	0.30	0.74
		ATE-MSE (cm)↓	0.31	**0.31**	3.2	90.92	1.47	104.88
Ours	RGB	PSNR[dB]↑	37.01	36.11	37.28	20.64	26.52	25.08
		SSIM↑	0.96	0.95	**0.96**	0.77	0.86	0.85
		LPIPS↓	0.077	0.090	0.086	0.29	0.13	0.18
		ATE-MSE (cm)↓	0.22	0.42	0.42	5.18	**0.38**	0.36
Ours	RGB-D	PSNR[dB]↑	**38.25**	**36.43**	**38.11**	**21.70**	**27.08**	**25.79**
		SSIM↑	**0.97**	**0.96**	**0.96**	**0.79**	**0.87**	**0.86**
		LPIPS↓	**0.075**	0.081	**0.084**	**0.252**	**0.113**	**0.169**
		ATE-MSE (cm)↓	**0.19**	0.32	**0.40**	**4.66**	0.42	**0.31**

Table 2: **Comparison with MonoGS [3].** We conduct experiments both taking monocular RGB and RGB-D as input on Replica [5] and TUM-RGBD [6] datasets, each consisting of 3 sequences. The best results are shown in **bold**, and the second best results are underlined.

4 Limitations and Future Works

While our method has demonstrated significant effectiveness, several limitations need to be addressed to improve its applicability in more challenging environments. Currently, the approach may struggle with scenes that involve significant motion blur or dynamic objects. Future research will focus on enhancing the robustness and adaptability of our method to better handle these complex scenarios. Additionally, to develop a more practical and comprehensive SLAM system, future work will focus on integrating loop closing, map reusing, and re-localization capabilities.

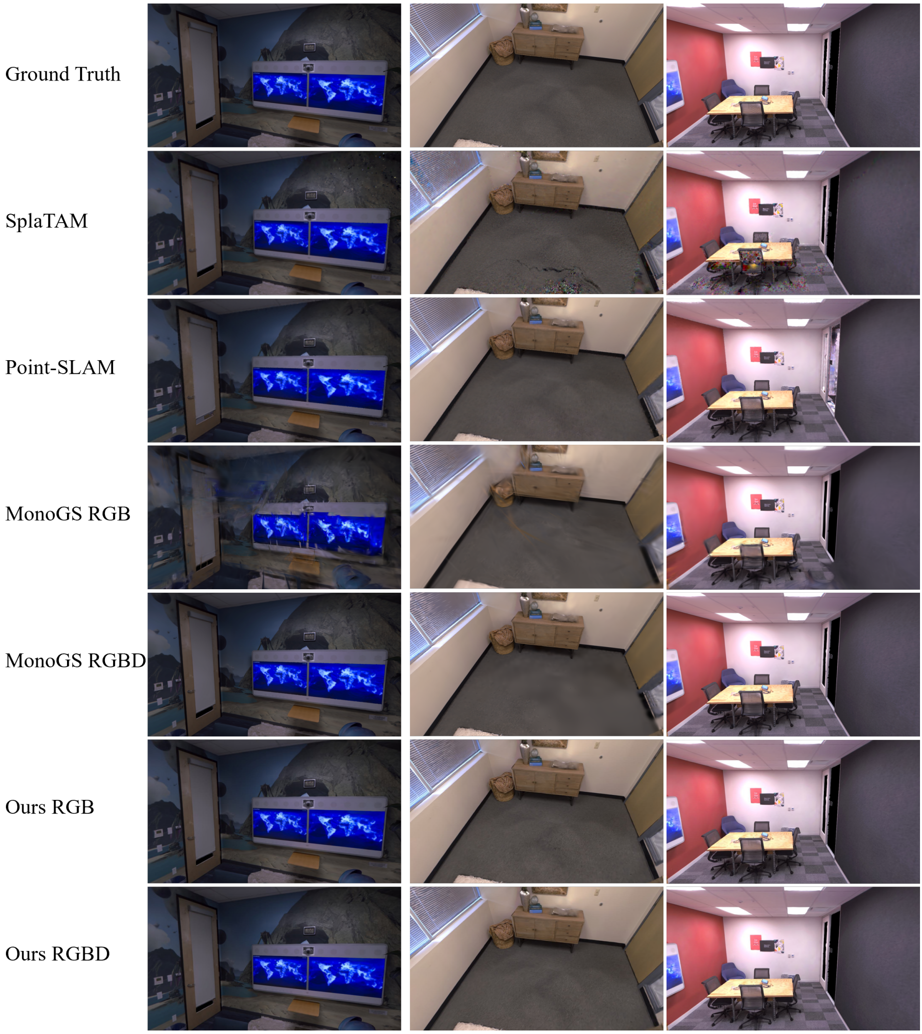

Figure 3: Rendering samples on Replica dataset.

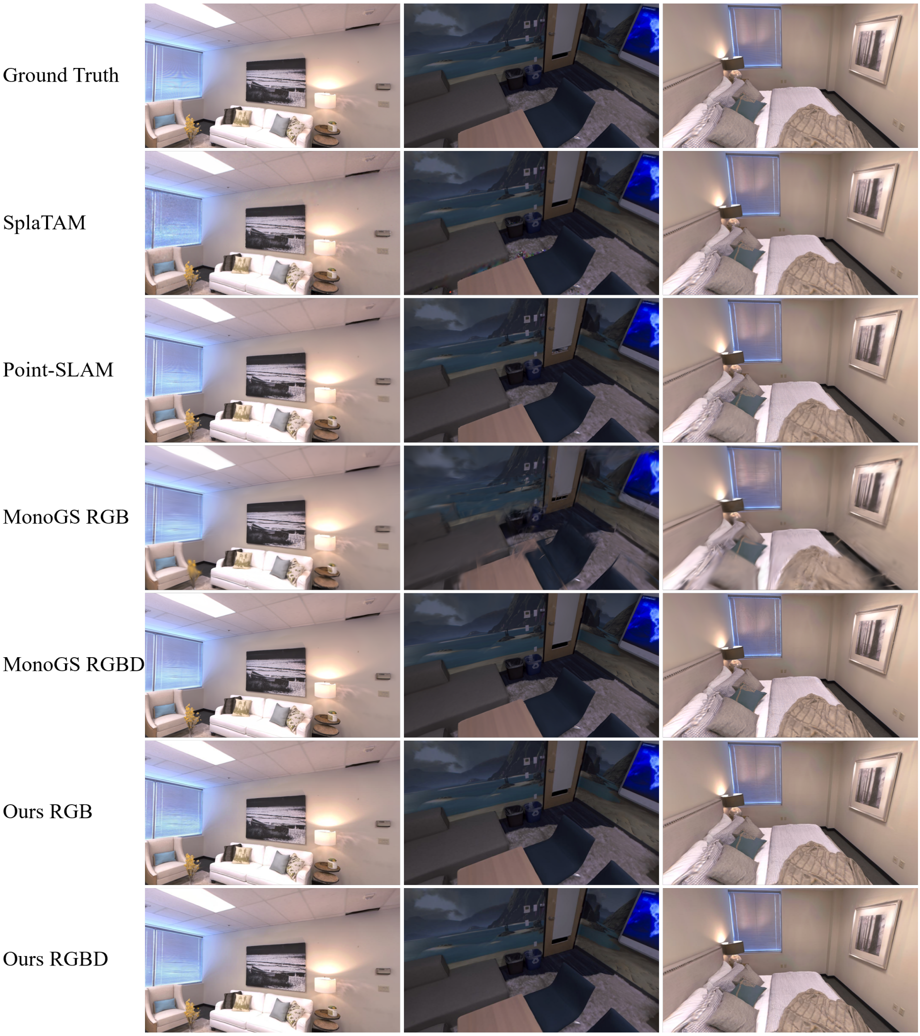

Figure 4: Rendering samples on Replica dataset.

References

[1] Nikhil Keetha, Jay Karhade, Krishna Murthy Jatavallabhula, Gengshan Yang, Sebastian Scherer, Deva Ramanan, and Jonathon Luiten. Splatam: Splat, track & map 3d gaussians for dense rgb-d slam. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2024.

[2] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. ACM Transactions on Graphics, 42(4), July 2023. URL https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/.

[3] Hidenobu Matsuki, Riku Murai, Paul H. J. Kelly, and Andrew J. Davison. Gaussian Splatting SLAM. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2024.

[4] Erik Sandström, Yue Li, Luc Van Gool, and Martin R. Oswald. Point-slam: Dense neural point cloud-based slam. In Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2023.

[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] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of rgb-d slam systems. In Proc. of the International Conference on Intelligent Robot Systems (IROS), Oct. 2012.