DyGS-SLAM: Real-Time Accurate Localization and Gaussian Reconstruction for Dynamic Scenes

Supplementary Material

1. Details

Sec.3.1 Dynamic Feature Detection

In Sec.3.1, the weight parameters for geometric and appearance information are set to $\alpha = 0.5$ and $\beta = 0.5$, respectively. We use the Interquartile Range (IQR) method to determine the reprojection constraint threshold, the reprojection depth residual threshold e_{th}, and the scoring threshold Φ_{th}. Specifically, we compute the reprojection error, depth residual, or score values for all feature points outside the potential moving object boxes, sort them in ascending order, and calculate the first quartile Q_1 and third quartile Q_3 of the sorted values. Based on this, the threshold is defined as:

$$th = Q_3 + k \cdot (Q_3 - Q_1), \tag{1}$$

where the adjustment parameter k is set to 2.

Sec.3.2 Dynamic Object Box Correction

In Sec.3.2, the candidate point set $Y = \{y_1, y_2, \ldots, y_m\}$ is defined as the set of feature points within a 50-pixel range around the detection box. To determine whether a candidate point y_j belongs to the dynamic object in the detection box, a threshold T is set based on the probability density distribution of foreground points. The probability density $p(x)$ of foreground points is estimated using a Gaussian Mixture Model (GMM), assuming an approximately normal distribution.

Specifically, the mean μ_p and standard deviation σ_p of the probability density of foreground points are calculated as:

$$\mu_p = \frac{1}{n}\sum_{i=1}^{n} p(x_i), \quad \sigma_p = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(p(x_i) - \mu_p)^2}, \tag{2}$$

where n represents the number of foreground points. Based on the statistical characteristics of the probability density distribution, the threshold is defined as:

$$T = \mu_p - k \cdot \sigma_p, \tag{3}$$

where the adjustment parameter k is set to 2 to moderately reduce sensitivity to outliers. Candidate points y_j with probability density $p(y_j)$ exceeding the threshold T are considered part of the dynamic object and subsequently contribute to the correction of the detection box.

Sec.3.4 Gaussian Mapping and Adaptive Feature Densification

1) Gaussian Mapping: The formal definition of a 3D Gaussian point is given as:

$$G(x \mid \mu, \Sigma) = e^{-\frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)}, \tag{4}$$

where $\mu \in \mathbb{R}^3$ represents the spatial mean, and $\Sigma \in \mathbb{R}^{3\times3}$ denotes the covariance matrix. To ensure the validity of the matrix during optimization, the covariance matrix Σ is decomposed into a scale matrix S and a rotation matrix R as follows:

$$\Sigma = RSS^\top R^\top. \tag{5}$$

During rendering, the 3D Gaussian points are projected onto a 2D plane. Using the intrinsic matrix K and the extrinsic matrix T, the 2D mean μ' and covariance Σ' are defined as:

$$\mu' = K[\mu, 1]^\top, \quad \Sigma' = JT\Sigma T^\top J^\top, \tag{6}$$

where J is the Jacobian matrix approximating the affine transformation of the projection. Each Gaussian point is associated with an opacity value σ and a view-dependent color c, which is determined by a set of spherical harmonic coefficients. The pixel color C is computed by performing alpha compositing on the sorted 2D Gaussian points, proceeding from front to back:

$$C = \sum_{i \in N} T_i G_i \left(u \mid \mu', \Sigma'\right) \sigma_i c_i, \tag{7}$$

where:

$$T_i = \prod_{j=1}^{i-1} \left(1 - G_j \left(u \mid \mu', \Sigma'\right) \sigma_j\right). \tag{8}$$

This equation represents the blending of transparency and color for different Gaussian points during rendering to generate the final 2D image.

2) Adaptive Feature Densification: The image is divided into 16×16 blocks, with the adjustment weights set to $w_G = w_V = 0.5$. The maximum number of sampling points is set to $n_{\max} = 15$, and the adjustment parameter is set to $\beta = 0.5$.

Other Details

Our system is fully implemented in C++ and CUDA. The object detection algorithm used is the TensorRT version of

YOLOX [1], with the YOLOX-s weight model. In our system, the semantic thread and the tracking thread run in parallel.

The number of features extracted per image is set to 2000.

2. Additional Results

2.1. Qualitative Evaluation of Localization Performance

Figs. 1 and 2 show the ATE of our method on selected sequences from the TUM [14] and BONN [12] datasets. Experimental results demonstrate that our method effectively overcomes dynamic disturbances to achieve accurate camera localization.

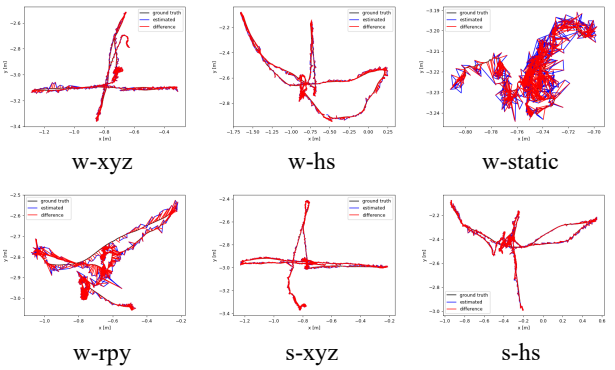

Figure 1. The ATE of our method on selected sequences from the TUM dataset.

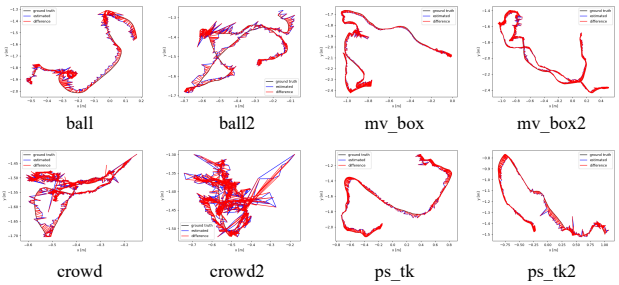

Figure 2. The ATE of our method on selected sequences from the BONN dataset.

2.2. Evaluation of Reconstruction Performance in Dynamic Scenes

2.2.1. Qualitative Evaluation

Figs. 4 and 5 illustrate the visual comparisons of rendered images produced by our method and those generated by PhotoSLAM [3], SplaTAM [6], and MonoGS [10] on other sequences from the TUM RGB-D dataset [14] and BONN RGB-D Dynamic dataset [12]. Several existing SLAM methods for dynamic scenes based on NeRF [11] or 3DGS [7] have not been open-sourced. We reproduced their results based on the methods demonstrated in the NID-SLAM [18] paper, and the visual comparisons of rendered images are shown in Fig. 6. For DDN-SLAM [9], DN-SLAM [13], Rodyn-SLAM [4], and DGS-SLAM [8], we compared the rendered images from the same viewpoints as those presented in their original papers, as shown in Figs. 7 and 3, where blank areas indicate sequences for which the respective papers did not provide results. The experimental results demonstrate that our algorithm can reconstruct a long-term consistent map containing only static backgrounds under dynamic interference conditions, showing significant advantages in image clarity and artifact reduction.

2.2.2. Quantitative Evaluation

Following RoDyn-SLAM [4] and DG-SLAM [17], we evaluate the reconstruction quality of static maps in dynamic scenes using three metrics: (i) accuracy (cm), (ii) completeness (cm), and (iii) completion rate (the percentage of points within a 5 cm threshold). The quantitative results in Tab. 1 demonstrate the superiority of our algorithm in reconstruction quality.

Method	Acc.↓	Comp.↓	Comp. Ratio↑
Co-SLAM [15]	18.06	60.86	16.27
ESLAM [5]	40.94	73.03	34.93
RoDyn-SLAM [4]	11.86	**13.71**	40.12
DG-SLAM [17]	<u>8.06</u>	15.46	<u>43.67</u>
Ours	**7.33**	<u>14.52</u>	**45.78**

Table 1. The average reconstruction quality across the ball, ball2, ps_tk, ps_tk2, and mv_box2 sequences of the BONN dataset. The best values are in bold, and the second-best are underlined.

Method	PSNR↑		SSIM↑		LPIPS↓	
	TUM	BONN	TUM	BONN	TUM	BONN
SplaTAM [6]	15.32	18.18	0.62	0.73	0.42	<u>0.25</u>
DGS [8]	<u>21.12</u>	–	<u>0.82</u>	–	<u>0.16</u>	–
Gassidy [16]	–	<u>24.24</u>	–	<u>0.78</u>	–	0.32
Ours	**22.15**	**26.33**	**0.89**	**0.86**	**0.15**	**0.22**

Table 2. Evaluation of mapping quality based on novel view synthesis results for BONN dataset (ball, ball2, ps_trk, ps_trk2, mv_box2) and TUM dataset (w_xyz, w_hs, w_static).

As shown in Tab. 2, our method maintains high map quality even under the influence of dynamic objects. It is important to note that the datasets used lack explicit ground truth benchmarks (i.e., the ground truth images contain unremoved dynamic objects, while the rendered images represent results with dynamic objects removed). The comparison is based on input images and the results produced by

different methods; therefore, filtering out dynamic objects leads to relatively lower values for these metrics.

2.3. Evaluation of Reconstruction Performance in Static Scenes

Figs. 8, 9, and 10 show the visual comparisons of rendered images produced by our method and PhotoSLAM [3] on the TUM dataset sequence `fr3_long_office_household`, and the ICL-NUIM dataset [2] sequences `lr/kt2` and `of/kt3`, respectively. In this comparative experiment, the number of features extracted per image was set to 1000. The experimental results indicate that the proposed densification method based on image texture complexity significantly improves the reconstruction of scenes with uneven lighting, low texture, or smooth surfaces.

2.4. Real-Time Performance and Resource Utilization

Tab. 3 presents the running frame rate, rendering frame rate, GPU memory usage, and reconstruction model size of our system across different sequences, fully demonstrating its great potential for practical applications.

Seq	Running FPS ↑	Rendering FPS ↑	GPU Memory ↓	Model Size (MB)
w_hs	33.12	786.28	3.09	7.8
w_xyz	32.49	729.93	2.96	6.2
ball2	32.12	1112.58	2.67	3.8
mov_box	34.49	925.32	2.76	4.2
crow3	34.78	1051.13	3.12	2.8
syn	36.43	1381.85	2.87	0.6

Table 3. Running frame rate, rendering frame rate, GPU memory usage, and reconstructed model size of our system across different sequences.

#Features	1000	2000	3000	4000	5000
Running FPS ↑	36.85	33.90	29.82	25.97	20.57

Table 4. The relationship between the number of Gaussian features and system running speed. The values presented correspond to the averages across all sequences in Tab. 3.

As shown in Tab. 4, the system's running speed exhibits a negative correlation with the number of Gaussian features. The rendering FPS, GPU memory usage, and model size remain nearly unchanged as the number of Gaussian features increases from 1000 to 5000, and thus are not listed due to space constraints. Notably, our experiments reveal that a higher number of Gaussian features does not necessarily lead to better reconstruction and rendering quality. This is attributed to the image feature extraction process and our densification strategy, which ensure that the feature density aligns with the complexity of the environmental textures. The rendering results presented in this paper correspond to feature counts of 2000/1000, achieving SOTA performance.

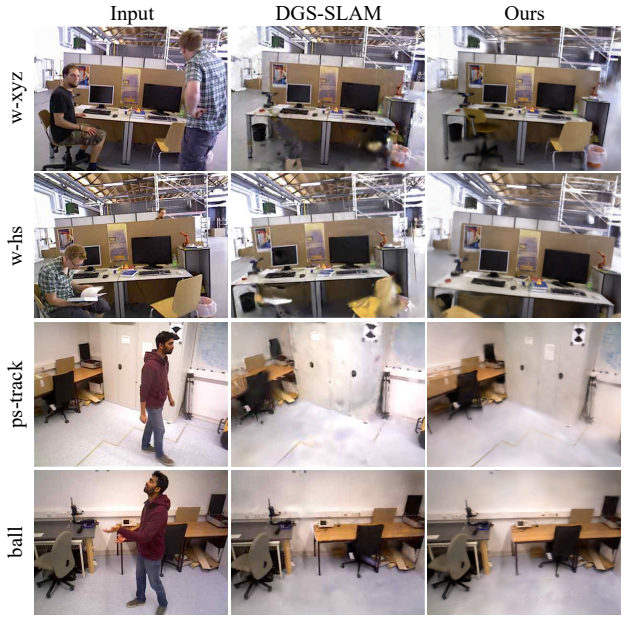

Figure 3. Visual Comparison of Rendered Images with DGS-SLAM.

References

[1] Zheng Ge, Songtao Liu, Feng Wang, Zeming Li, and Jian Sun. Yolox: Exceeding yolo series in 2021. *arXiv preprint arXiv:2107.08430*, 2021. 2

[2] Ankur Handa, Thomas Whelan, John McDonald, and Andrew J Davison. A benchmark for rgb-d visual odometry, 3d reconstruction and slam. In *2014 IEEE international conference on Robotics and automation (ICRA)*, pages 1524–1531. IEEE, 2014. 3

[3] Huajian Huang, Longwei Li, Hui Cheng, and Sai-Kit Yeung. Photo-slam: Real-time simultaneous localization and photo-realistic mapping for monocular stereo and rgb-d cameras. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 21584–21593, 2024. 2, 3

[4] Haochen Jiang, Yueming Xu, Kejie Li, Jianfeng Feng, and Li Zhang. Rodyn-slam: Robust dynamic dense rgb-d slam with neural radiance fields. *IEEE Robotics and Automation Letters*, 2024. 2

[5] Mohammad Mahdi Johari, Camilla Carta, and François Fleuret. Eslam: Efficient dense slam system based on hybrid representation of signed distance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 17408–17419, 2023. 2

Figure 4. Visual comparison of rendered images on the TUM datasets.

[6] 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*, pages 21357–21366, 2024. 2

[7] Bernhard Kerbl, Georgios Kopanas, Thomas Leimkühler, and George Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Trans. Graph.*, 42(4):139–1, 2023. 2

[8] Mangyu Kong, Jaewon Lee, Seongwon Lee, and Euntai Kim. Dgs-slam: Gaussian splatting slam in dynamic environment. *arXiv preprint arXiv:2411.10722*, 2024. 2

[9] Mingrui Li, Zhetao Guo, Tianchen Deng, Yiming Zhou, Yuxiang Ren, and Hongyu Wang. Ddn-slam: Real time dense dynamic neural implicit slam. *IEEE Robotics and Automation Letters*, 2025. 2

[10] Hidenobu Matsuki, Riku Murai, Paul HJ Kelly, and Andrew J Davison. Gaussian splatting slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18039–18048, 2024. 2

[11] 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. *Communications of the ACM*, 65(1):99–106, 2021. 2

[12] Emanuele Palazzolo, Jens Behley, Philipp Lottes, Philippe Giguere, and Cyrill Stachniss. Refusion: 3d reconstruction in dynamic environments for rgb-d cameras exploiting residuals. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7855–7862. IEEE, 2019. 2

[13] Chenyu Ruan, Qiuyu Zang, Kehua Zhang, and Kai Huang. Dn-slam: A visual slam with orb features and nerf mapping in dynamic environments. *IEEE Sensors Journal*, 2023. 2

[14] Jürgen Sturm, Nikolas Engelhard, Felix Endres, Wolfram Burgard, and Daniel Cremers. A benchmark for the evaluation of rgb-d slam systems. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 573–580. IEEE, 2012. 2

[15] Hengyi Wang, Jingwen Wang, and Lourdes Agapito. Co-slam: Joint coordinate and sparse parametric encodings for neural real-time slam. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13293–13302, 2023. 2

[16] Long Wen, Shixin Li, Yu Zhang, Yuhong Huang, Jianjie Lin, Fengjunjie Pan, Zhenshan Bing, and Alois Knoll. Gassidy: Gaussian splatting slam in dynamic environments. *arXiv preprint arXiv:2411.15476*, 2024. 2

[17] Yueming Xu, Haochen Jiang, Zhongyang Xiao, Jianfeng Feng, and Li Zhang. Dg-slam: Robust dynamic gaussian splatting slam with hybrid pose optimization. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. 2

[18] Ziheng Xu, Jianwei Niu, Qingfeng Li, Tao Ren, and Chen Chen. Nid-slam: Neural implicit representation-based rgb-d slam in dynamic environments. *arXiv preprint arXiv:2401.01189*, 2024. 2

Figure 5. Visual comparison of rendered images on the BONN datasets.

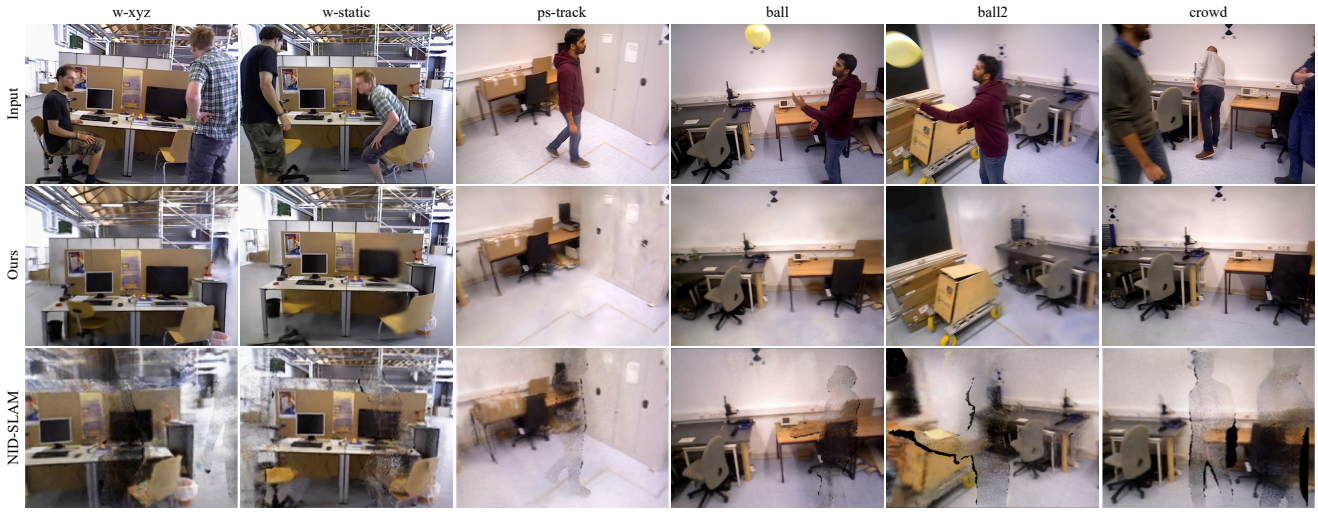

Figure 6. The visual comparisons of rendered images with NID-SLAM. The results of NID-SLAM were reproduced based on the methods described in its paper.

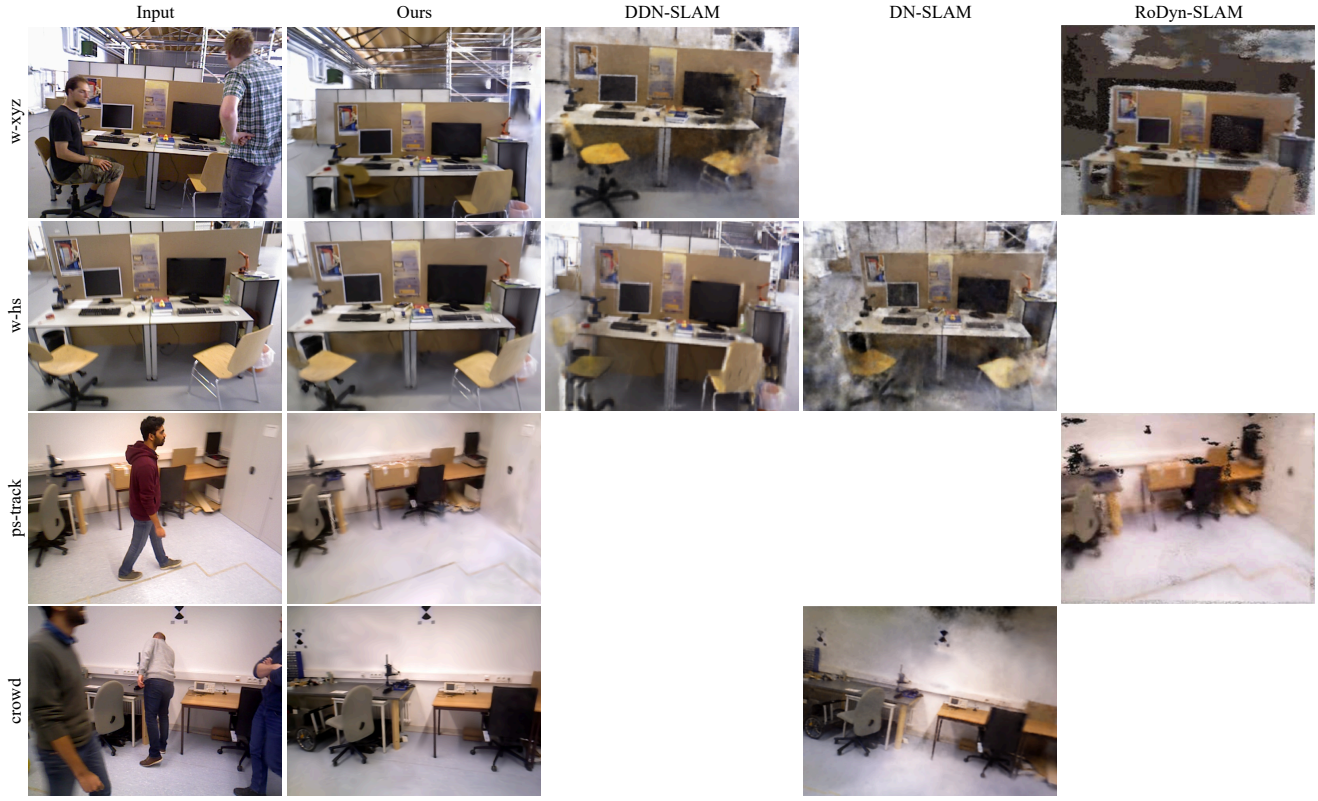

Figure 7. The visual comparisons of rendered images with other SLAM methods for dynamic scenes based on NeRF or 3DGS. Blank areas indicate sequences for which the respective papers did not provide results.

Figure 8. The visual comparisons of rendered images on the TUM dataset sequence `fr3_long_office_household`.

Figure 9. The visual comparisons of rendered images on the ICL-NUIM dataset sequence `lr/kt2`.

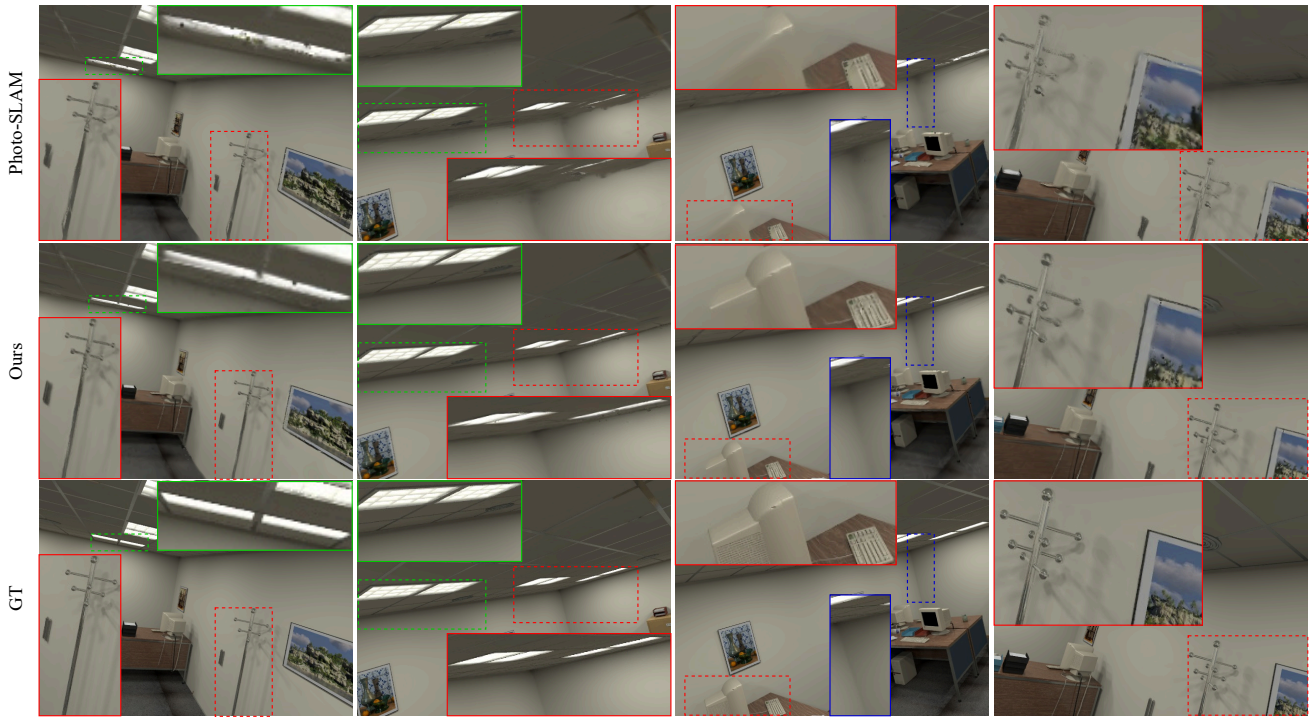

Figure 10. The visual comparisons of rendered images on the ICL-NUIM dataset sequence `of/kt3`.