






Supplementary material for *I*²-SLAM: Inverting Imaging Process for Robust Photorealistic Dense SLAM

Gwangtak Bae^{*1}, Changwoon Choi^{*1}, Hyeongjun Heo¹,
Sang Min Kim¹, and Young Min Kim^{†1,2}

¹ Dept. of Electrical and Computer Engineering, Seoul National University

² INMC & IPAI, Seoul National University

A Implementation Details

RGB-SLAM We reimplement NeRF-SLAM [8] with torch-ngp [11]. Our model takes the same inputs as in NeRF-SLAM [8]. We utilize camera poses, depth estimates, and uncertainty maps of keyframes from NeRF-SLAM. We sequentially add keyframes to the training set. The learning rate for camera tracking is $1e-4$. The learning rate for the tone-mapping module is $1e-4$. We utilize camera preconditioning [6] for virtual camera pose optimization. We use photometric errors instead of 2D projections to make camera poses to change in motion-blur direction.

RGBD-SLAM For ScanNet [3], we follow the experimental setup of SplatAM [5], except for unpadding edges. We unpad 24 pixels in color images and 12 pixels in depth images to ignore black regions caused by the calibration process. *I*²-SLAM parameterizes virtual camera poses with center poses and velocities. We use the same learning rate for the center pose and translational velocity as SplatAM, while the learning rate for the rotational velocity is set to one-tenth. The learning rate for the tone-mapping module is $1e-2$. The learning rate for the trajectory regularizer is $1e6$. *I*²-SLAM optimizes monochromatic WB for ScanNet. For the synthetic dataset, *I*²-SLAM optimizes monochromatic WB and monochromatic CRF. We initialize virtual camera poses by interpolating between the current frame and the previous frame by a factor of $1/10$. The learning rate for trajectory regularizer is $1e4$. The learning rate for camera tracking is $5e-5$. The mapping interval is every five frames. We use 30 iterations for mapping and 200 iterations for tracking.

B Dataset

Synthetic dataset We create a new synthetic dataset that simultaneously considers autoexposure and motion blur with Blender’s [1] Cycles engine. We use three

^{*} Authors contributed equally to this work.

[†] Young Min Kim is the corresponding author.

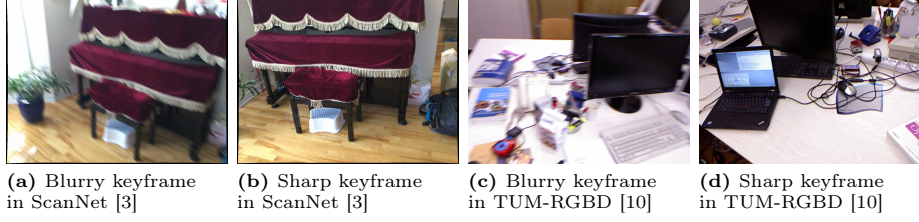


Fig. 1: Examples of sharp keyframes and blurry keyframes in real-world datasets.

Blender scenes with trajectories manually set for each scene pretending a human is capturing the scene. At first, we render HDR images along the trajectory without motion blur for ground truth images. Then we render LDR images from the HDR images based on the classical Reinhard tone-mapping [7]. We define our tone-mapping function following HDR-NeRF [4]:

$$I_{LDR} = 255((\Delta t I_{HDR}/(\Delta t I_{HDR} + 1))^{1/2.2}), \quad (1)$$

where I_{LDR} and I_{HDR} are the pixel value of LDR and HDR images, $\Delta t = 2^{EV}$ is the exposure time and EV is the exposure value. We set EV for each frame by converting the HDR RGB image to gray values and mapping the LDR output of the median gray value to $255/2$. Finally, we blur the LDR images using Blender’s built-in motion blur function. We set the shutter curve as a square function. The shutter value SV_i of the i th frame is decided by the exposure as follows:

$$SV_i = a2^{EV_i} + b, \quad (2)$$

where a and b are the predefined parameters per scene and trajectory. Specifically, we set a and b by forcing the minimum and maximum shutter values to be 0.1 and 1 for each trajectory in the scene for our dataset.

Real-world dataset We use part of the sequences as input instead of the entire sequences for experiments in Scannet (0~300 in 0024-01, 0~300 in 0031-00, 1000~1400 in 0736-00, and 0~500 in 0785-00). As real-world datasets are heavily affected by motion blur, it is unlikely to evaluate the image quality with the entire input as a reference. Therefore, we manually annotate sharp images from input image sequences to evaluate rendering quality in real-world datasets. 9.5% of input frames from ScanNet [3] and 29.7% from TUM-RGBD [10] are selected for evaluation. We select the entire frame of `fr2/xyz` of TUM-RGBD for evaluation since the corresponding scene was carefully captured to avoid motion blur. Figure 1 depicts examples of selected sharp and blurry images.

C Trajectory Regularization

In this section, we explain our trajectory loss terms (Eq. 16 and 17 in the main manuscript) in more detail, especially describing the relationship between the global scale parameter a and the frame rate of input video.

Table 1: Rendering quality comparison against RGBD-SLAM baselines on ScanNet [3]. I^2 -SLAM in this table represents our RGBD-SLAM model which is incorporated into SplaTAM [5].

Methods	Metrics	ScanNet [3]			
		0024-01	0031-00	0736-00	0785-00
Co-SLAM [12]	PSNR	17.34	21.05	16.38	19.89
	SSIM	0.570	0.645	0.525	0.647
	LPIPS	0.542	0.473	0.651	0.618
Point-SLAM [9]	PSNR	17.67	21.09	16.21	20.71
	SSIM	0.533	0.605	0.508	0.677
	LPIPS	0.439	0.417	0.508	0.483
SplaTAM [5]	PSNR	21.60	24.64	24.50	19.63
	SSIM	0.786	0.773	0.847	0.719
	LPIPS	0.236	0.275	0.182	0.340
I^2 -SLAM	PSNR	23.39	26.89	24.07	26.40
	SSIM	0.780	0.796	0.828	0.762
	LPIPS	0.180	0.236	0.175	0.238

The trajectory loss guides the start and end camera poses to be located on the global trajectory. Here, global trajectory stands for the trajectory defined by the middle camera position of each frames. Specifically, we define trajectory loss as a L2-norm between start and end camera poses and the one point on linear interpolant of \mathbf{t}^i and \mathbf{t}^{i-1} . Also, the interpolating parameter should be determined by the timestamp of start and end poses.

Also, when we assume the linear motion during the exposure time, $t^i - t^{i-1} = \frac{1}{f}$, where $t^i = \frac{t_s^i + t_e^i}{2}$ and f is the frequency of the input video samples. Also, $t^i - t_s^i = \frac{\Delta t^i}{2}$. Therefore, the translation vector of the start position $\mathbf{t}(t_s^i)$ should be located on LERP($\mathbf{t}^{i-1}, \mathbf{t}^i, 1 - \frac{f\Delta t^i}{2}$). Now, we can find the relationship between the global scale parameter a and the video frame rate f , $a = \frac{f}{2}$. Same analogy holds for the end camera location and rotation vectors.

D Additional Experimental Results

Comparison with more baselines We further compare I^2 -SLAM with two widely used neural RGBD-SLAM methods, Co-SLAM [12] and Point-SLAM [9]. In Tab. 1, Both two methods show a lack of rendering performance in ScanNet’s all scenes except scene 0785-00. As depicted in Fig. 4, scene 0785-00 contains severe appearance changes which can cause inaccurate mapping on homogeneous surfaces. SplaTAM shows inferior performance than other baselines. However, by

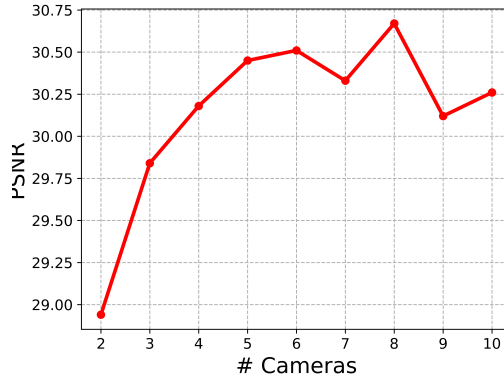


Fig. 2: PSNR variation over the number of virtual cameras

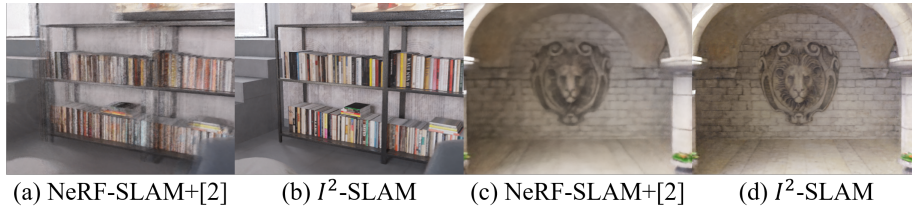


Fig. 3: Renderings of RGB-SLAM with image enhancements.

integrating our method into SplaTAM, it outperforms other methods with a large margin.

Qualitative results We conduct more qualitative comparison in Figs. 4 to 7. Figure 4 demonstrates qualitative results of our RGBD-SLAM model in ScanNet. I^2 -SLAM renders sharp images even when blurry frames are used for mapping. In the scene 0785-00, I^2 -SLAM successfully reflects extreme appearance changes. Figures 5 and 7 show that our method reconstructs small objects, *e.g.*, a pen and a scissors, and areas with complex texture. The effect of our tone-mapper can be seen through the removal of artifacts in Fig. 6.

Effects of the number of virtual cameras We use five virtual cameras to approximately simulate motion blur. We conduct an ablation study on the number of virtual cameras. Figure 2 shows results with our RGB-SLAM model in *Italian-flat-1*. With an increasing number of cameras, the performance sees an upward trend. Considering the trade-off between performance and runtime, we use five virtual cameras.

Table 2: Results of RGB-SLAM with enhanced image inputs.

Methods	TUM-RGBD				Synthetic			
	ATE ↓	PSNR ↑	SSIM ↑	LPIPS ↓	ATE ↓	PSNR ↑	SSIM ↑	LPIPS ↓
NeRF-SLAM	3.21	26.89	0.817	0.227	2.47	27.41	0.828	0.310
NeRF-SLAM+ [2]	3.27	26.61	0.825	0.202	3.77	27.65	0.820	0.237
NeRF-SLAM+ [2]+TM	2.80	26.22	0.814	0.215	3.48	27.13	0.809	0.254
I^2 -SLAM	1.28	29.40	0.861	0.151	1.48	29.59	0.878	0.256

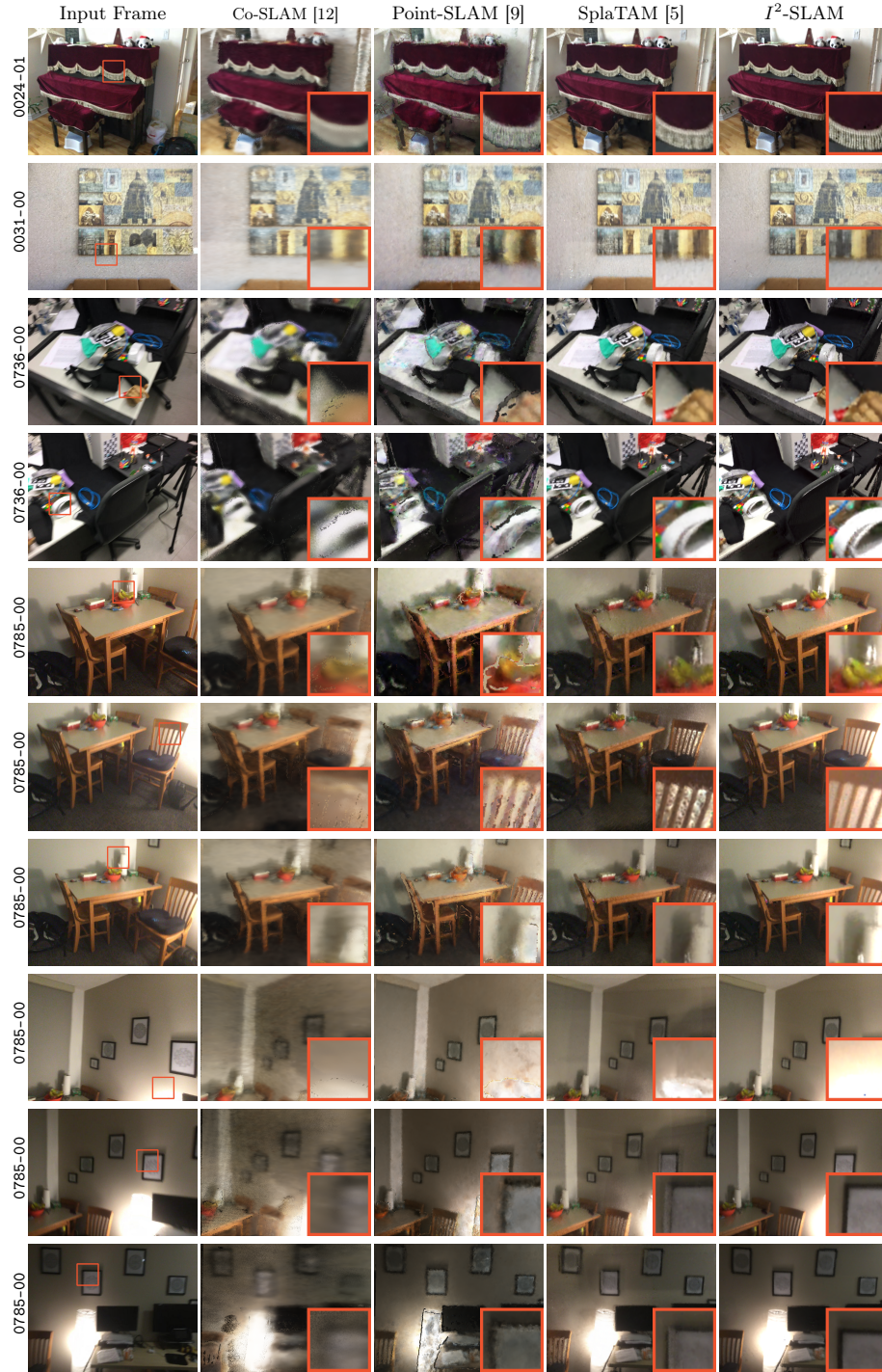
E Comparison with SLAM using 2D Image Deblurring

We conduct comparisons with the RGB-SLAM with enhanced image inputs. We first deblur input frames with NAFNet [2] and run NeRF-SLAM with deblurred images. In Tab. 2 and Fig. 3, even with the deblurred inputs, there is no significant improvement in tracking and rendering quality. While integrating our tone mapper (TM) enhances trajectory accuracy, it still shows lower performance compared to I^2 -SLAM. This is attributed to the multi-view inconsistency in image deblurring process.

References

1. Blender, O.C.: Blender - a 3D modelling and rendering package. Blender Foundation, Stichting Blender Foundation, Amsterdam (2018), <http://www.blender.org>
2. Chen, L., Chu, X., Zhang, X., Sun, J.: Simple baselines for image restoration. In: European conference on computer vision. pp. 17–33. Springer (2022)
3. Dai, A., Chang, A.X., Savva, M., Halber, M., Funkhouser, T., Nießner, M.: Scannet: Richly-annotated 3d reconstructions of indoor scenes. In: Proc. Computer Vision and Pattern Recognition (CVPR), IEEE (2017)
4. Huang, X., Zhang, Q., Feng, Y., Li, H., Wang, X., Wang, Q.: Hdr-nerf: High dynamic range neural radiance fields. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 18398–18408 (2022)
5. 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)
6. Park, K., Henzler, P., Mildenhall, B., Barron, J.T., Martin-Brualla, R.: Camp: Camera preconditioning for neural radiance fields. ACM Transactions on Graphics (TOG) **42**(6), 1–11 (2023)
7. Reinhard, E., Stark, M., Shirley, P., Ferwerda, J.: Photographic tone reproduction for digital images. ACM Trans. Graph. **21**(3), 267–276 (jul 2002). <https://doi.org/10.1145/566654.566575>
8. Rosinol, A., Leonard, J.J., Carlone, L.: Nerf-slam: Real-time dense monocular slam with neural radiance fields. In: 2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). pp. 3437–3444. IEEE (2023)
9. 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)

10. 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)
11. Tang, J.: Torch-ngp: a pytorch implementation of instant-ngp (2022), <https://github.com/ashawkey/torch-ngp>
12. Wang, H., Wang, J., Agapito, L.: 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. pp. 13293–13302 (2023)



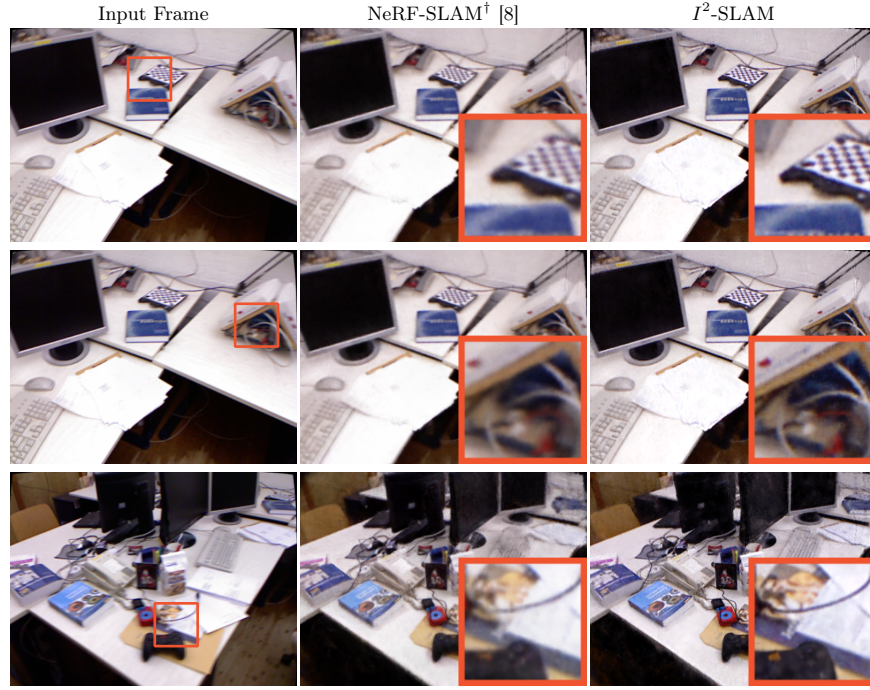


Fig. 5: Additional qualitative results in **fr1/desk** of TUM-RGBD [10]

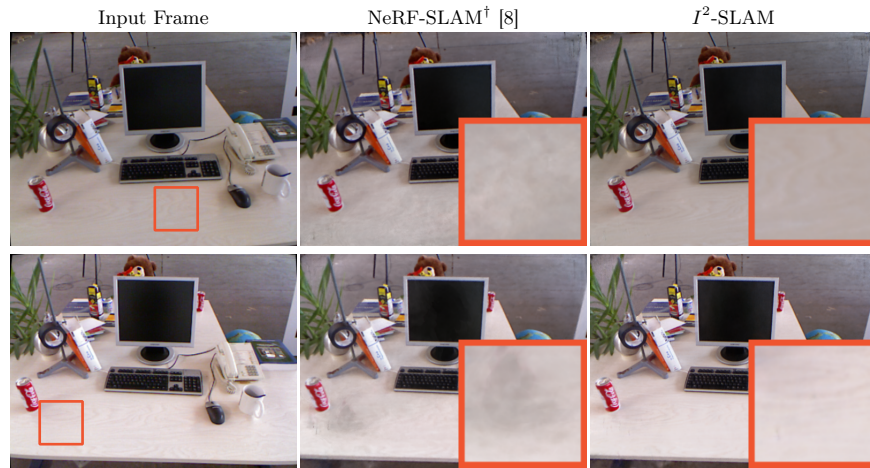


Fig. 6: Additional qualitative results in **fr2/xyz** of TUM-RGBD [10]

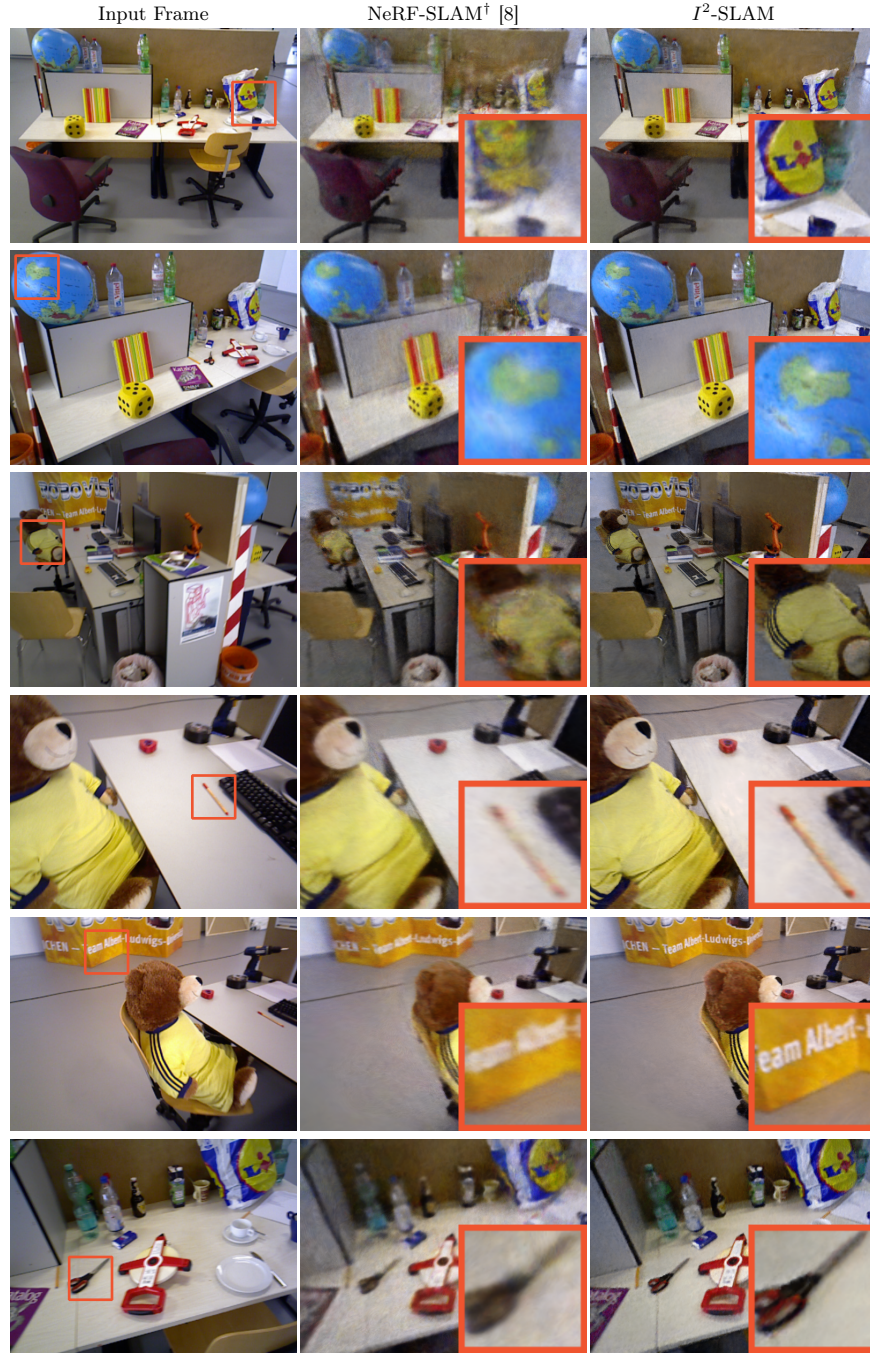


Fig. 7: Additional qualitative results in fr3/office of TUM-RGBD [10]