PNeRFLoc: Visual Localization with Point-based Neural Radiance Fields -Supplementary Material-

Boming Zhao[1], **Luwei Yang**, **Mao Mao**[1], **Hujun Bao**[1], **Zhaopeng Cui**[1*]

[1]State Key Lab of CAD & CG, Zhejiang University

In this supplementary material, we first discuss the implementation details in Sec. A. Subsequently, in Sec. B we will explain how to train a scene-specific PointNeRF for reconstruction. Additionally, we will delve into more details of pose refinement with efficient rendering-based optimization in Sec. C. Then, in Sec. D, we demonstrate the effectiveness of our efficient rendering-based optimization. Finally, we provide additional ablation studies and confirm the necessity of using our scene-agnostic localization feature extractor in Sec. E.

A Implementation Details

In this section, we introduce the parameter settings when training PNeRFLoc on three different datasets. Our model operates on a single NVIDIA GeForce RTX 3090 GPU.

- **7Scenes (Shotton et al. 2013)** For each scene in the 7Scenes, we select one image from every ten reference images to train our model. For each scene, we train 30k iterations. The resolution of each image is 640×480.
- **Replica (Straub et al. 2019)** For each scene in the Replica, we select one image from every ten reference images to train our model. For each scene, we train 30k iterations. The resolution of each image is 1200×680.
- **Cambridge Landmarks (Kendall, Grimes, and Cipolla 2015)** For the "shop" scene, we use all reference images as training data, and for other scenes, we select one image from every two reference images to train our model. During the training process, if the training images come from the same sequence, we use the same appearance embedding code. For each scene, we train 150k iterations. The resolution of each image is 1024×576. Additionally, as the query images and reference images are taken at different times under varying lighting conditions, we need to learn the appearance embedding code of the query image sequences. Therefore we randomly select one query image from each sequence as a representative and then, we render an image by using the camera pose provided by structure-based localization and minimize the residual between the rendering image and the query image by optimizing the appearance embedding code. It should be noted that we optimize all representative images together for 2k iterations.

We also compared the efficiency of our method and the SOTA method PixLoc on the Replica dataset. The PixLoc (Sarlin et al. 2021) runs at 0.25Hz with a 3.5GB GPU memory cost. Our method is comparable and runs at 0.14Hz with a 12GB GPU memory cost without any optimization. Note that our system can be improved with implementation optimization using CUDA.

B Scene-Specific PointNeRF Reconstruction

The radiance of a pixel can be determined by casting a ray through the pixel, sampling M shading points at positions $\{x_i | i = 1, ..., M\}$ along the ray, and then accumulating radiance by utilizing volume density (see Sec. 3.1 in the main paper). Specifically, given a training view image I, we first sample N rays across the image, and on each of these rays, we sample M shading points. To ensure these shading points are primarily concentrated on the surface of the scene, we rely on the sampling method provided by PointNeRF that shading points are collected only in the areas where the emitted rays are close to the point cloud. Following this, we apply scene-specific feature adaptation (see Sec. 3.2 in the main paper) to convert the features of K nearest neural point neighbors of the shading points into learned point-wise neural features. These learned point-wise neural features of the neighboring points are then input into Multi-Layer Perceptrons (MLPs) to learn the density and RGB values for each point. Moreover, we employ standard inverse distance weighting to aggregate the neural density and color from these K neighboring points to obtain a single density and color of each shading point. Finally, we calculate the color of each ray using Eq. 1 in the main paper and train the NeRF model by minimizing the rendering loss function (see Eq. 5 in the main paper).

C More Details on Pose Refinement

We depict the detailed process of pose refinement in Fig. C. First, we obtain the visual reference image pose $(\mathbf{R}, \mathbf{t})$ of the given query image through structure-based localization and render the visual reference image under this pose with a learned PNeRFLoc model. We then sample points (red points) on the visual reference image and back-project them into 3D space through the rendering depth map. In the first

[*]Corresponding author.

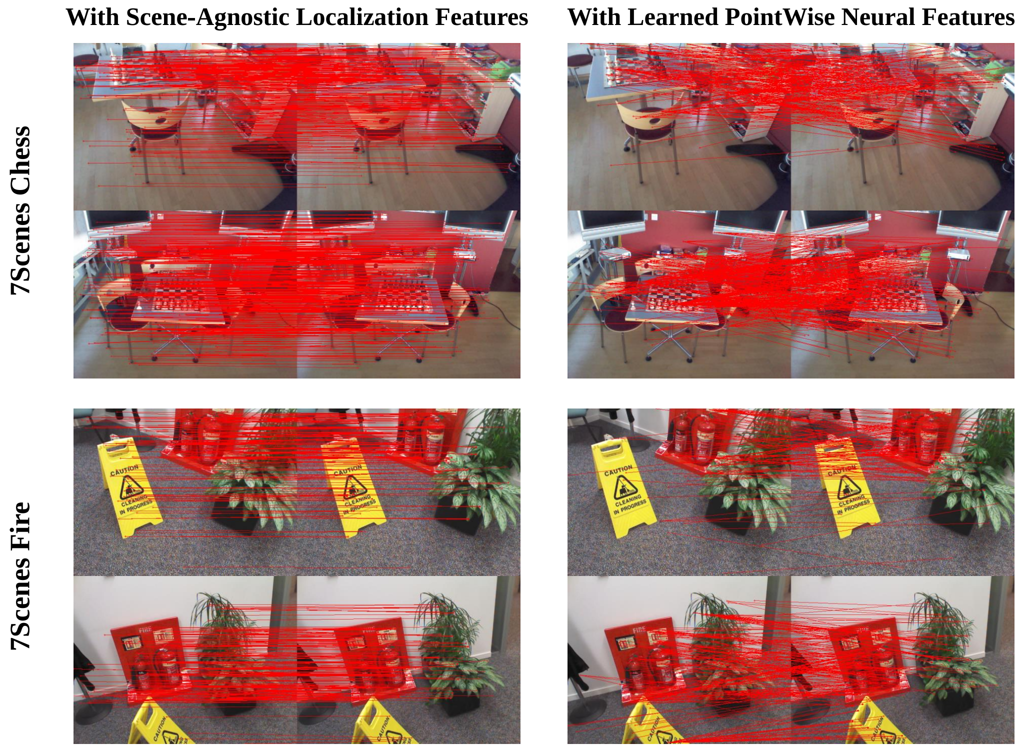
With Scene-Agnostic Localization Features
With Learned PointWise Neural Features
7Scenes Chess
7Scenes Fire

Figure A: **Feature matching results.** We compared the matching results with different features. Specifically, we extract 800 key points from the query image. The features extracted on the left images are scene-agnostic localization features, while those on the right are learned pointwise neural features. We then match 2D key points with the 3D points in the point cloud by computing the cosine similarity and project the corresponding 3D points onto the query image using the ground truth camera poses. Finally, we illustrate the matching relationships and demonstrate the necessity of using a localization feature extractor instead of the learned pointwise neural features for feature matching.

Table A: **Ablation of matching 2D-3D points with different features.** We apply the PnP algorithm with the matching results obtained from different features. We also report the median translation/rotation errors (meters/degrees) and the best results are highlighted in **bold**.

Config.	7Scenes	
	Chess	Fire
w/ Learned Pointwise Neural Features	3.19 / 134	4.49 / 124
w/ Scene-Agnostic Localization Features	**0.03 / 1.21**	**0.03 / 1.47**

step of optimization, we take $(\mathbf{R}, \mathbf{t})$ as the initial pose of the query image and project the 3D points onto the query image (red points). Due to the imprecise pose, there is a misalignment between the sampled points on the reference image and the projected points on the query image. Therefore, we can minimize misalignment by optimizing the camera pose $(\mathbf{R}, \mathbf{t})$. Finally, we obtained the optimized accurate pose $(\mathbf{R}', \mathbf{t}')$, and the projected points are represented as purple points, which align with the sampled points on the reference. In this way, we don't need to render a new image for each step of optimization and avoid the backpropagation through the networks for better convergence.

D Effectiveness of the Proposed Efficient Rendering-based Optimization

In the experiment of our main paper (Sec. 3.3 in the main paper), we have shown the effectiveness of the proposed efficient rendering-based optimization given the initial pose estimated from the structure-based localization. One may be curious about whether the proposed method can handle the initial poses with large errors (Also mentioned in Sec. 1 in the main paper).

To illustrate this, we evaluated our method on the Replica dataset using a randomly generated pose with significant viewpoint changes, where the translation/rotation error is 0.90m/25.6°. We set the total number of optimization itera-

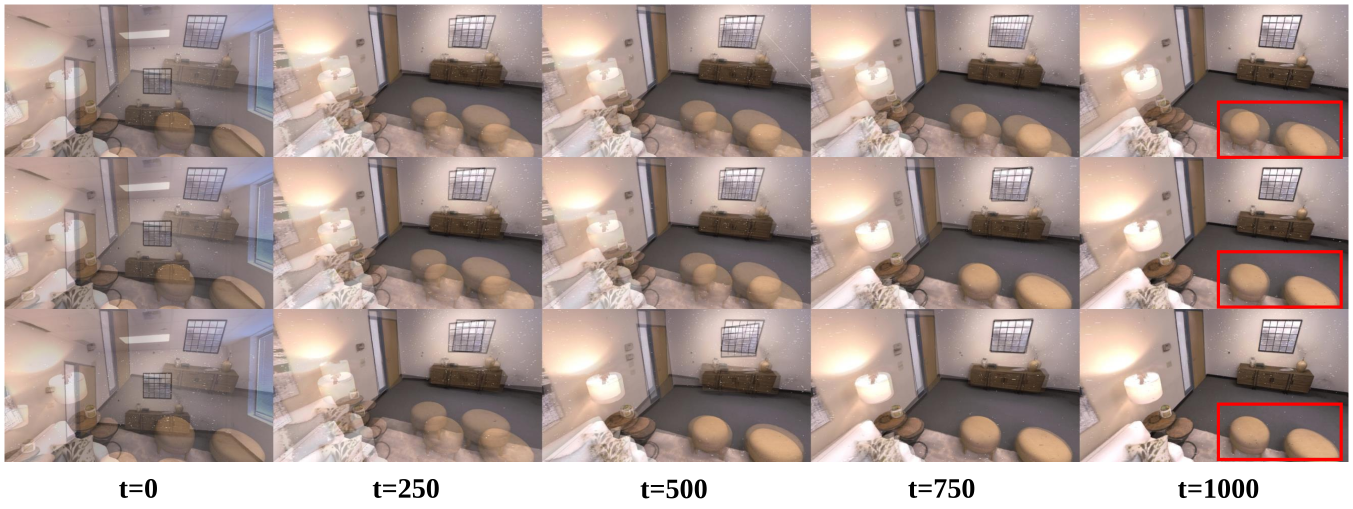

Figure B: **Visualization of the rendering-based optimization given the initial pose with a large error.** We visualize the rendered images based on the estimated pose at step t of the optimization and the query image together. From top to bottom, it shows the results with one, two, and four rendered images.

Config.	4%	3%	2%	1%
room0	0.06 m / 1.13°	0.04 m / 1.02°	0.03 m / 0.98°	0.02 m / 0.75°
room1	0.48 m / 7.53°	0.25 m / 4.32°	0.15 m / 2.52°	0.08 m / 1.70°
room2	0.12m / 3.05°	0.12 m / 2.32°	0.06 m / 1.32°	0.03 m / 1.24°
office0	0.10 m / 2.08°	0.06 m / 1.39 °	0.04 m / 0.82°	0.01 m / 0.59°
office1	0.08 m / 2.91°	0.04 m / 1.85°	0.02 m / 0.95°	0.01 m / 0.69°
office2	0.04 m / 0.92°	0.04 m / 0.85°	0.03 m / 0.80°	0.01 m / 0.71°
office3	0.06 m / 1.28°	0.04 m / 0.91°	0.04 m / 0.96°	0.02 m / 0.74°
office4	0.05 m/ 1.02°	0.04 m / 0.85°	0.03 m/ 0.66°	0.02 m/ 0.59°

Table B: **Robustness against different depth errors on Replica datasets.** We report median translation/rotation errors. The "n%" means adding a Gaussian noise with the mean value of $\pm n\% * depth$ to the GT depth.

tions at 1k and compared the outcomes of rendering the visual reference image once, twice, and four times respectively. The different optimization results are shown in Fig. B. The first row displays the results of rendering a single reference image, and at 1000 steps, the rendering image shows significant misalignment with the query image and the translation/rotation error between the optimized pose and the Ground Truth pose is 0.34m/4.54°. The second row demonstrates the results of rendering the reference image twice, meaning we render the visual reference image again at 500 steps based on the estimated pose from the previous optimization. We can see that the estimated pose with two rendered images is better but there is still slight misalignment at 1000 steps with the translation/rotation error of 0.11m/1.57°. The last row shows the results of rendering the reference image four times, that is, rendering the visual reference image at 250, 500, and 750 steps. In this case, the final rendered image almost aligns with the query image, with a translation/rotation error of 0.03m/0.42°, achieving precise localization results. Our experiments demonstrate the effectiveness of our efficient rendering-based optimization which can optimize an initial pose with large errors via iteratively rendered images. This experiment demonstrates that even when there are large view changes between the query image and the visual reference image, our method can still effectively optimize the pose. We attribute this to our proposed efficient rendering-based optimization method that uses warping loss, which by avoiding backpropagation through the networks provides greater stability and is less prone to falling into incorrect local minima, and more accurate optimization results can be obtained by iteratively rendering the visual reference image with updated poses for several times. However, in most cases in our experiment, the initial poses provided by structure-based localization are not far away from the GT poses. Therefore, in practice, we only render the visual reference image once for efficiency.

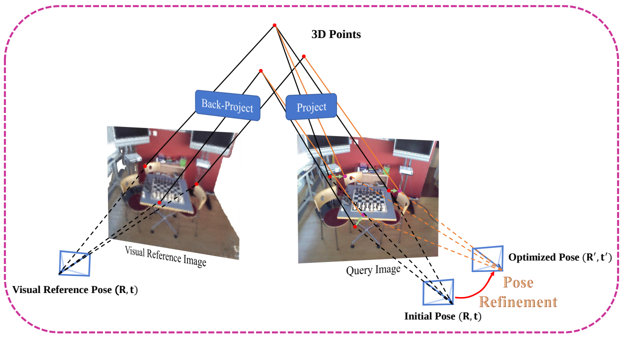
3D Points
Back-Project
Project
Visual Reference Image
Query Image
Optimized Pose $(\mathbf{R}', \mathbf{t}')$
Pose Refinement
Visual Reference Pose $(\mathbf{R}, \mathbf{t})$
Initial Pose $(\mathbf{R}, \mathbf{t})$

Figure C: **Pose refinement with efficient rendering-based optimization.** We back-project the sample points on the visual reference image into the world coordinate system and project them on the query with the initial pose $(\mathbf{R}, \mathbf{t})$. Then We can optimize the camera pose by aligning the pixels on the rendering image and the query image. To be noted, our sampled points are very dense in practice (90k points).

E More ablation studies

E.1 Ablation Studies on Scene-Agnostic Feature Extractor

We conduct experiments to demonstrate the importance of our scene-agnostic feature extractor. We compared directly learned pointwise neural features and scene-agnostic localization features (R2D2 (Revaud et al. 2019)) for feature matching on the 7Scenes dataset with our points filter strategy (see Sec. 3.3 in the main paper). As shown in Fig A, due to the lack of distinctiveness in learned pointwise neural features, they cannot be used as robust descriptors, leading to many incorrect correspondences. On the other hand, using well-studied deep features for localization performed excellently, with most matches being correct. We also report structure-based localization median translation/rotation errors(meters/degrees) in Table A. We can see that if the learned pointwise neural features are used for feature matching, it will result in significant errors in the structure-based localization. This demonstrates the necessity of choosing the scene-agnostic localization features instead of the learned pointwise neural features for feature matching.

E.2 Robustness against Noisy Depth

To evaluate the robustness of our method against noisy depth input, we add random noises to the depth image on the Replica Dataset. From Table B, we can see that our method is robust to noisy depth and it still works reasonably well when the noise level is up to 4%. This is also demonstrated by our experiment on the real-world 7Scenes dataset where the depth is also noisy.

References

Kendall, A.; Grimes, M.; and Cipolla, R. 2015. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *Proceedings of the IEEE international conference on computer vision*, 2938–2946.

Revaud, J.; Weinzaepfel, P.; de Souza, C. R.; and Humenberger, M. 2019. R2D2: Repeatable and Reliable Detector and Descriptor. In *NeurIPS*.

Sarlin, P.-E.; Unagar, A.; Larsson, M.; Germain, H.; Toft, C.; Larsson, V.; Pollefeys, M.; Lepetit, V.; Hammarstrand, L.; Kahl, F.; et al. 2021. Back to the feature: Learning robust camera localization from pixels to pose. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 3247–3257.

Shotton, J.; Glocker, B.; Zach, C.; Izadi, S.; Criminisi, A.; and Fitzgibbon, A. 2013. Scene coordinate regression forests for camera relocalization in RGB-D images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2930–2937.

Straub, J.; Whelan, T.; Ma, L.; Chen, Y.; Wijmans, E.; Green, S.; Engel, J. J.; Mur-Artal, R.; Ren, C.; Verma, S.; et al. 2019. The Replica dataset: A digital replica of indoor spaces. *arXiv preprint arXiv:1906.05797*.