Nerfels: Renderable Neural Codes for Improved Camera Pose Estimation - Supplementary Material

Gil Avraham[1,*], Julian Straub[3], Tianwei Shen[3], Tsun-Yi Yang[3], Hugo Germain[2], Chris Sweeney[3], Vasileios Balntas[3], David Novotny[4], Daniel DeTone[3], Richard Newcombe[3]
Monash University[1], École des Ponts[2], Facebook Reality Labs[3], Facebook AI Research[4]
gil.avraham@monash.edu, hugo.germain@enpc.fr
{jstraub, tianweishen, tsunyi, sweeneychris, vassileios, dnovotny, ddetone, newcombe @fb.com}

1. Algorithm Outline

Algorithm 1 A full algorithmic outline of proposed system

OFFLINE

$\{\mathcal{I}_i, \mathcal{P}_{i,N}, c_i\}_{i=1}^{N_c} \leftarrow$ Extract triplets of images poses and codes ($c_i \sim \mathcal{N}(0, \frac{1}{d_c})$) from a scene

$\mathcal{D}_{NR}(P_N, c) \leftarrow$ Optimise $\mathcal{L}_D$ to recover $\hat{\Theta}, \hat{\mathcal{C}}$

ONLINE

RelativePoseEstimator(I_R, I_Q):

Extract keypoints and features
$\boldsymbol{k}_R, \boldsymbol{d}_R \leftarrow keypointFeatureExtractor(I_R)$
$\boldsymbol{k}_Q, \boldsymbol{d}_Q \leftarrow keypointFeatureExtractor(I_Q)$
Match keypoints and retain the matches
$\boldsymbol{k}_R, \boldsymbol{k}_Q \leftarrow Matcher(\boldsymbol{k}_R, \boldsymbol{d}_R, \boldsymbol{k}_Q, \boldsymbol{d}_Q)$
# Extract reference Nerfels)
for $i = 1$ to $|\boldsymbol{k}_R|$ **do**
  $\hat{P}_{N,i}, \hat{c}_i \leftarrow \arg\min_{P_{N,i}, c_i} \mathcal{L}_{inv}(P_{N,i}, c_i, k_i)$
end for
# Solve relative pose
$\hat{P} \leftarrow \arg\min_P \mathcal{L}_{PnP+Photo}(\boldsymbol{k}_R, \boldsymbol{k}_Q, (\hat{\boldsymbol{P}}_N, \hat{\boldsymbol{c}}))$
return $\hat{P}$

2. Cumulative Error Curves

We provide cumulative error curves corresponding to the feature extractors from Table 1 (SIFT, D2Net, Superpoint). These curves add additional detail to the evaluation performed in the main paper, which selected operating points of 0.25, 0.5, and 1.0 meters along the x-axis. In each of the three cases, the addition of the photometric term into the optimisation provided by Nerfel rendering improves across all thresholds. Additionally, as the difficulty of the pose estimation increases, the gap widens.

*Performed while interning at Facebook Reality Labs

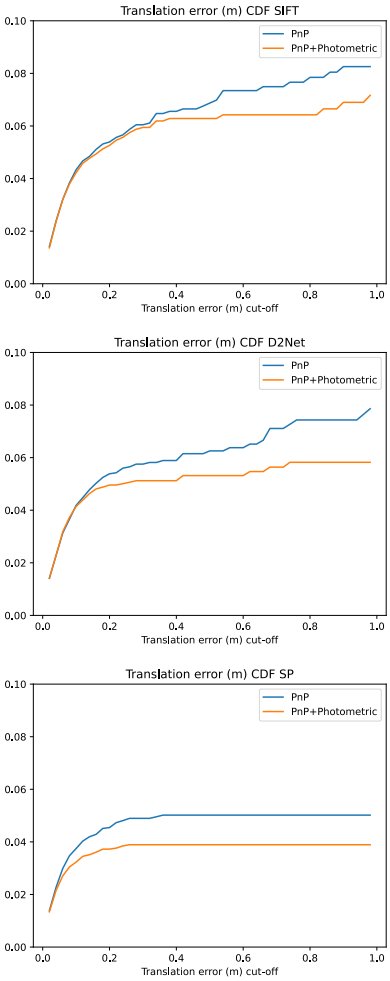


Figure 1. **Cumulative Error Plots for Various Local Features**. Cumulative error plots for (top) SIFT (middle) D2Net and (bottom) SuperPoint are shown. The blue line shows vanilla PnP, and the orange line shows the joint optimization of PnP + Photometric error, which is enabled by Nerfels. As the difficulty of problem increases from left to right on the x-axis, the benefit is larger.

3. Implementation Details

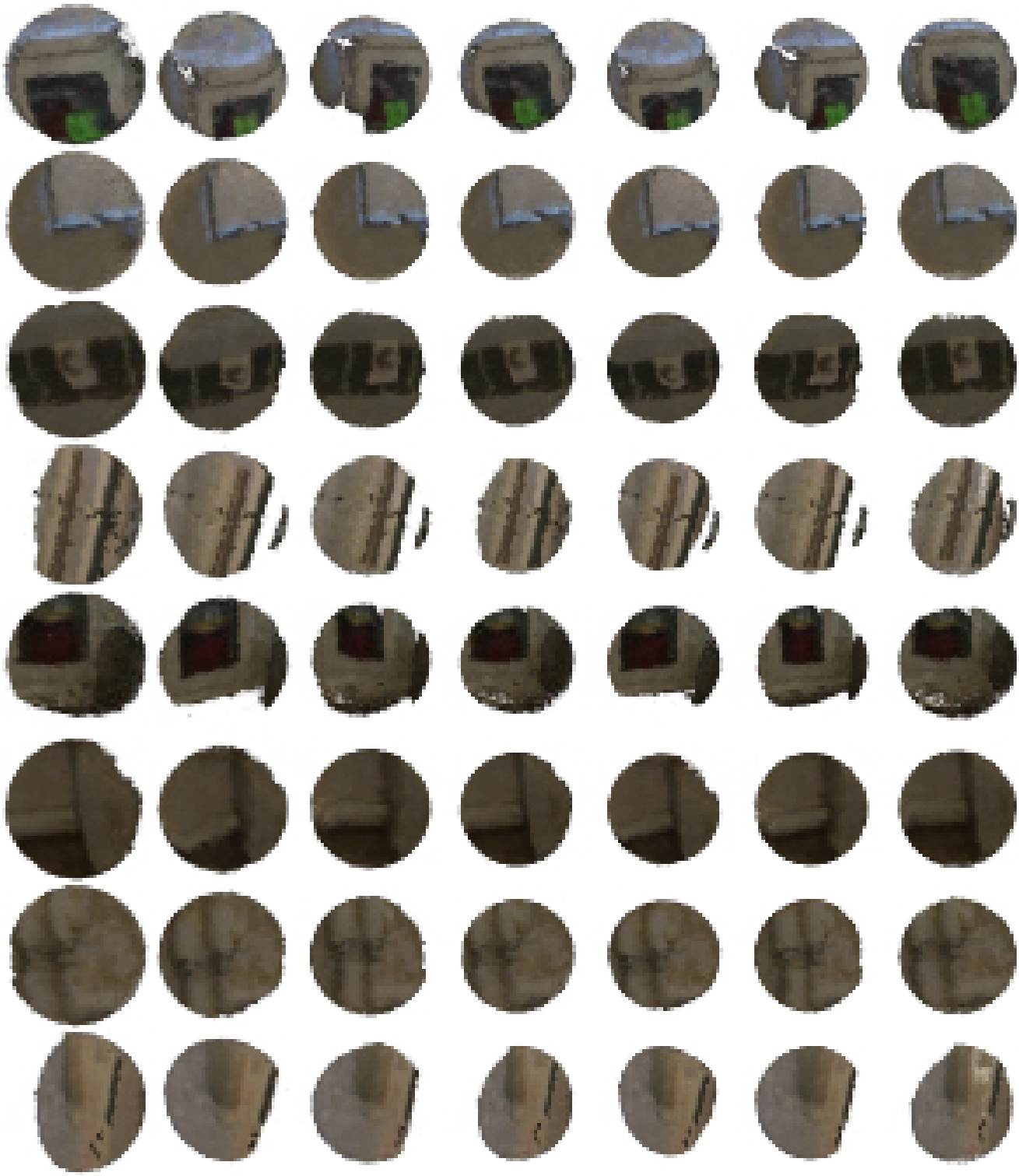
Network Architecture For constructing the neural rendering decoder, we follow a similar network architecture as done in [2] with the additional Nerfel code $d_c = 64$, concatenated to the encodings of the queried points. The queried points and view directions are encoded using the positional encodings with $\mathcal{L}_x = 10, \mathcal{L}_\phi = 4$. The point encodings are fed to an 8 layer FC network with 256 hidden dimension with point encodings being skipped and concatenated to the features every 3 layers. This is followed by concatenating the view positional encodings with a single FC layer and the outputs result from σ head and an RGB ($\hat{r}$) head.

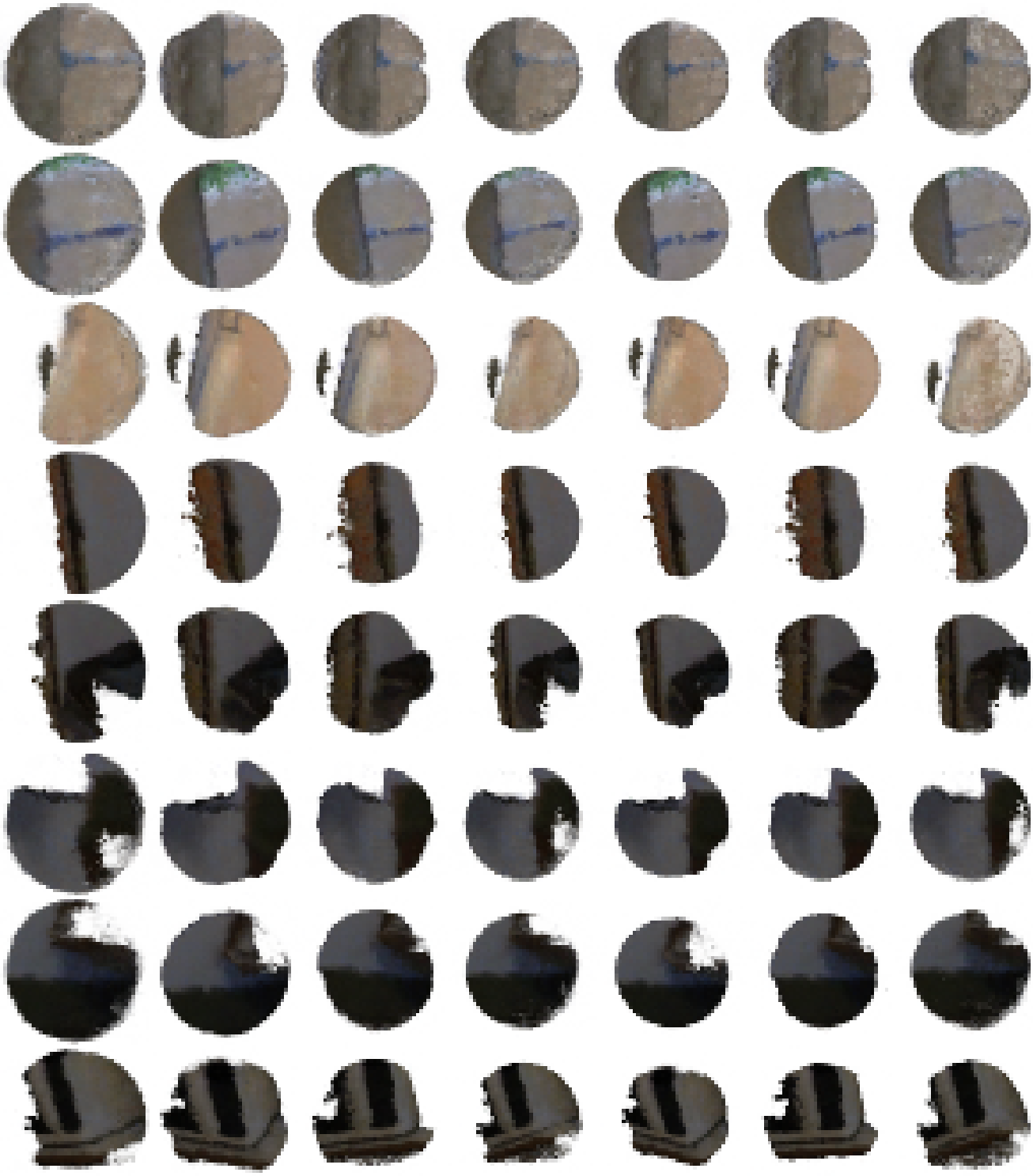
Training Settings The decoder D_{NR} was trained over $800k$ iterations using Adam optimiser [1] with a learning rate of $5e^{-4}$ and a decay factor of 0.1 annealed through the training iterations. The size of an image in the decoder training phase is 80×60, and the number of rays batched in each iteration is set to 1024. The number of coarse rays sampled during training is 64 followed by 128 fine rays using the stratified sampling approach. Within the volume rendering function a 0.2 STD of noise is used for perturbing the radiance field. When collecting the Nerfel codes (Section 4), the radius of a Nerfel sphere $r_s = 0.3$ in metric measurement and given a set of keypoints $\mathcal{K}$ over an entire scene the threshold t_f is adaptively chosen so that $N_c \approx 0.2|\mathcal{K}|$.

Evaluation Settings For both synthetic and real datasets, image pairs were chosen with overlap of $[0.4, 0.8]$ to simulate wide-baseline scenarios. The overlap value was computed by symmetrically taking the reference frame and reprojecting it's 3D points onto the query image and and vice versa. We check the percentage of the points that fall within the opposing frame and the average of both is the overlap value.

4. Nerfel Samples

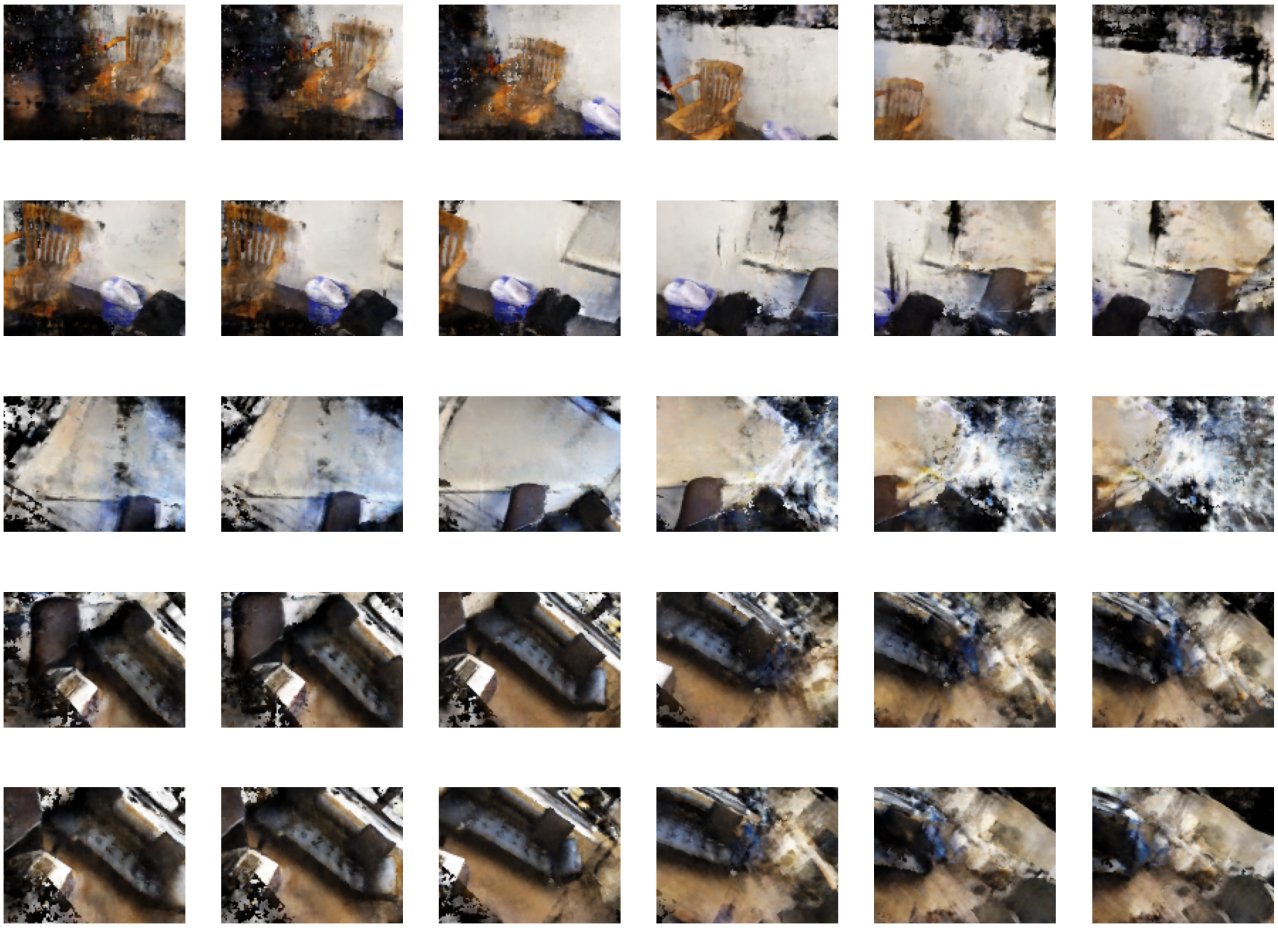
In the following we provide qualitative samples of Nerfels with frontal rotation poses.





5. Nerf Full-Scene Rendering Samples

Here we provide 5 different locations in 4 full-scenes that are implicitly represented using a Nerf network. For each location the camera was perturbed to show the surroundings. From the renderings it can be seen that attempting to model a full-scene using a Nerf with a similar capacity to that used for Nerfels results in low quality scene representation. This causes the pose-estimation optimisation converge to incorrect solutions.



References

[1] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 2

[2] 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 *European Conference on Computer Vision*, pages 405–421. Springer, 2020. 2