

DFNet: Enhance Absolute Pose Regression with Direct Feature Matching

Shuai Chen, Xinghui Li, Zirui Wang, and Victor A. Prisacariu

Active Vision Lab, University of Oxford

Abstract. We introduce a camera relocalization pipeline that combines absolute pose regression (APR) and direct feature matching. By incorporating exposure-adaptive novel view synthesis, our method successfully addresses photometric distortions in outdoor environments that existing photometric-based methods fail to handle. With domain-invariant feature matching, our solution improves pose regression accuracy using semi-supervised learning on unlabeled data. In particular, the pipeline consists of two components: Novel View Synthesizer and DFNet. The former synthesizes novel views compensating for changes in exposure and the latter regresses camera poses and extracts robust features that close the domain gap between real images and synthetic ones. Furthermore, we introduce an online synthetic data generation scheme. We show that these approaches effectively enhance camera pose estimation both in indoor and outdoor scenes. Hence, our method achieves a state-of-the-art accuracy by outperforming existing single-image APR methods by as much as 56%, comparable to 3D structure-based methods.¹

Keywords: Absolute Pose Regression, Feature Matching, NeRF

1 Introduction

Estimating the position and orientation of cameras from images is essential in many applications, including virtual reality, augmented reality, and autonomous driving. While the problem can be approached via a geometric pipeline consisting of image retrieval, feature extraction and matching, and a robust Perspective-n-Points (PnP) algorithm, many challenges remain, such as invariance to appearance or the selection of the best set of method hyperparameters.

Learning-based methods have been used in traditional pipelines to improve robustness and accuracy, e.g. by generating neural network (NN)-based feature descriptors [7,16,17,25,26], combining feature extraction and matching into one network [30], or incorporating differentiable outlier filtering modules [1,2,3]. Although deep 3D-based solutions have demonstrated favorable results, many prerequisites often remain, such as the need for an accurate 3D model of the scene and manual hyperparameter tuning of the remaining classical components.

The alternative end-to-end NN-based approach, termed absolute pose regression (APR), directly regresses the absolute pose of the camera from input images

¹ The code is available in <https://code.active.vision>.

[14] without requiring prior knowledge about the 3D structure of the neighboring environment. Compared with deep 3D-based methods, APR methods can achieve at least one magnitude faster running speeds at the cost of inferior accuracy and longer training time. Although follow-up works such as MapNet [4] and Kendall *et al.* [13] attempt to improve APR methods by adding various constraints such as relative pose and scene geometry reprojection, a noticeable gap remains between APR and 3D-based methods.

Recently, Direct-PN [5] achieved state-of-the-art (SOTA) accuracy in indoor localization tasks among existing single-frame APR methods. As well as being supervised by ground-truth poses, the network directly matches the input image and a NeRF-rendered image at the predicted pose. However, it has two major limitations: (a) direct matching is very sensitive to photometric inconsistency, as images with different exposures could produce a high photometric error even from the same camera pose, which reduces the viability of photometric direct matching in environments with large photometric distortions, such as outdoor scenes; (b) there is a domain gap between real and rendered images caused by poor rendering quality or changes in content and appearance of the query scene.

In order to address these limitations, we propose a novel relocalization pipeline that combines APR and direct feature matching. First, we introduce a histogram-assisted variant of NeRF, which learns to control synthetic appearance via histograms of luminance information. This significantly reduces the gap between real and synthetic image appearance. Second, we propose a network *DFNet* that extracts domain invariant features and regresses camera poses, trained using a contrastive loss with a customized mining method. Matching these features instead of direct pixels colors boosts the performance of the direct dense matching further. Third, we improve generalizability by (i) applying a cheap Random View Synthesis (RVS) strategy to efficiently generate a synthetic training set by rendering novel views from randomly generated pseudo training poses and (ii) allow the use of unlabeled data. We show that our method outperforms existing single-frame APR methods by as much as 56% on both indoor 7-Scenes and outdoor Cambridge datasets. We summarize our main contributions as follows:

1. We introduce a direct feature matching method that offers better robustness than the prior photometric matching formulation, and devise a network *DFNet* that can effectively bridge the feature-level domain gap between real and synthetic images.
2. We introduce a histogram-assisted NeRF, which can scale the direct matching approach to scenes with large photometric distortions, *e.g.*, outdoor environments, and provide more accurate rendering appearance to unseen real data.
3. We show that a simpler synthetic data generation strategy such as RVS can improve pose regression performance.

2 Related Work

Absolute Pose Regression Absolute pose regression aims to directly regress the 6-DOF camera pose from an image using Convolutional Neural Networks.

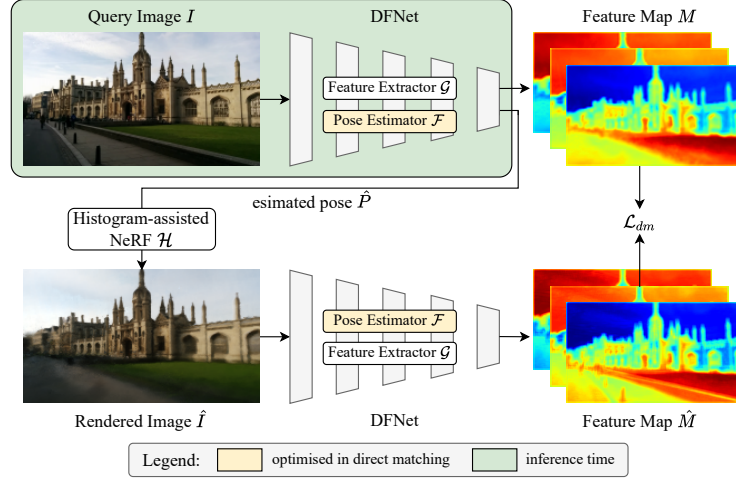


Fig. 1. Overview of the direct feature matching pipeline. Given an input image I , a pose regressor \mathcal{F} estimates a camera pose \hat{P} , from which a luminance prior NVS system \mathcal{H} renders a synthetic image \hat{I} . Domain invariant features of M and \hat{M} are extracted using a feature extractor \mathcal{G} , supplying a feature-metric direct matching signal \mathcal{L}_{dm} to optimize the pose regressor.

The first practice in this area is introduced by PoseNet [14], which is a GoogLeNet-backbone network appended with an MLP regressor. Successors of PoseNet propose several variations in network architectures, such as adding LSTM layers [36], adapting an encoder-decoder backbone [19], splitting the network into position and orientation branches [38], or incorporating attentions using transformers [29,28]. Other methods propose different strategies to train APR. Bayesian PoseNet [15] inserts Monte Carlo dropout to a Bayesian CNN that estimates pose with uncertainty. Kendall *et al.* [13] proposes to balance the translation and rotation loss at training using learnable weights and reprojection error. MapNet [4] trains the network using both absolute pose loss and relative pose loss but can infer in a single-frame manner. Direct-PoseNet (Direct-PN) [5] adapts additional photometric loss by comparing the query image with NeRF synthesis on the predicted pose.

Semi-supervised Learning in APR Several APR methods explore semi-supervised learning with additional images without ground-truth pose annotation to improve pose regression performance. To the best of our knowledge, MapNet+ [4] and MapNet+PGO [4] are the pioneers to train APR on unlabeled video sequences using external VO algorithms [8,9]. Direct-PN+ [5] finetune on unlabeled data from arbitrary viewpoints solely based on its direct matching formulation. While the direct matching idea from Direct-PN+ inspires our proposed method, we focus on training in the feature space. Our solution can scale to scenes with large photometric distortion, where the previous method fails.

Novel View Synthesis in APR Novel View Synthesis (NVS) can be beneficial to the visual relocalization task. For example, NVS can expand training space by generating extra synthetic data. Purkait *et al.* [24] propose a method to generate realistic synthetic training data for pose regression leveraging the 3D map and feature correspondences. LENS [22] deploys a NeRF-W [18] model to sample the scene boundaries and synthesize virtual views with uniformly generated virtual camera poses. However, Purkait *et al.* rely on a pre-computed reconstructed 3D map. LENS is limited by its costly offline computation efficiency and the lack of compensation to the domain gap between synthetic and real images, i.e., dynamic objects or artifacts. Another direction is to embed NVS into the pose estimation process. InLoc [32] verifies the predicted pose with view synthesis. Ng et al. [23] combine a multi-view stereo (MVS) model with a relative pose regressor (RPR). iNeRF [39], Wang *et al.* [37], and Direct-PN [5] utilize an inverted NeRF to optimize the camera pose. Our paper is the first to incorporate both strategies yet have major differences from the above methods. 1) we introduce an NVS method that can adapt to real exposure change in view synthesis. 2) we address the domain adaptation problem between the actual camera footage with synthetic images. 3) our synthetic data generation strategy is comparatively less constrained and can be deployed efficiently in online training.

3 Method

We illustrate our proposed direct feature matching pipeline in Fig. 1, which contains two primary components: 1) the DFNet network, which, given an input image I , uses a pose estimator \mathcal{F} to predict a 6-DoF camera pose and a feature extractor \mathcal{G} to compute a feature map M , and 2) a histogram-assisted NeRF \mathcal{H} , which compensates for high exposure fluctuation by providing luminance control when rendering a novel view given an arbitrary pose.

Training the direct feature matching pipeline can be split into two stages, (i) DFNet and the histogram-assisted NeRF, and (ii) direct feature matching. In stage one, we train the NVS module \mathcal{H} like a standard NeRF, and the DFNet with a loss term \mathcal{L}_{DFNet} in Eq. (5). In stage two, fixing the histogram-assisted NeRF and the feature extractor \mathcal{G} , we further optimize the main pose estimation module \mathcal{F} via a direct feature matching signal between feature maps extracted from the real image and its synthetic counterpart \hat{I} , which is rendered from the predicted pose \hat{P} of image I via the NVS module \mathcal{H} . At test time, only the pose estimator \mathcal{F} is required given the query image, which ensures a rapid inference.

This section is organized as follows: the DFNet pipeline is detailed in Section 3.1, followed by a showcase of our histogram-assisted NeRF \mathcal{H} in Section 3.2. To further boost the pose estimation accuracy, an efficient Random View Synthesis (RVS) training strategy is introduced in 3.3.

3.1 Direct Feature Matching For Pose Estimation

This section aims to introduce: 1) the design of our main network DFNet, 2) the direct feature matching formulation that boosts pose estimation performance in

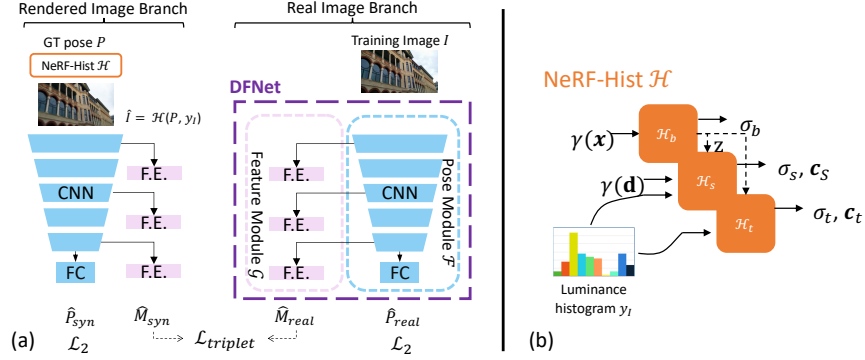


Fig. 2. (a) The training scheme for DFNet to close the domain gap between real images and rendered images. (b) The histogram-assisted NeRF architecture.

a semi-supervised training manner, and 3) the contrastive-training scheme that closes the domain gap between real images and synthetic images.

DFNet Structure The DFNet in our pipeline consists of two networks, a pose estimator \mathcal{F} and a feature extractor \mathcal{G} . The pose estimator \mathcal{F} in our DFNet is similar to an ordinary PoseNet, which predicts a 6-DoF camera pose $\hat{P} = \mathcal{F}(I)$ for an input image I , and can be supervised by an L_1 or L_2 loss between the pose estimation \hat{P} and its ground truth pose P .

The feature extractor \mathcal{G} in our DFNet takes as input feature maps extracted from various convolutional blocks in the pose estimator and pushes them through a few convolutional blocks, producing the final feature maps $M = \mathcal{G}(I)$, which are the key ingredients during feature-metric direct matching.

Two key properties of the feature extractor \mathcal{G} that we seek to learn are 1) domain invariance, i.e., being invariant to the domain of real images and the domain of synthetic images and 2) transformation sensitive, i.e., being sensitive to the image difference that is caused by geometry transformations. With these properties learned, our feature extractor can extract domain-invariant features during feature-metric direct matching while preserving geometry-sensitive information for pose learning. We detail the way to train the DFNet in the *Closing the Domain Gap* section.

Direct Feature Matching Direct matching in APR was first introduced by Direct-PN [5], which minimizes the photometric difference between a real image I and a synthetic image \hat{I} rendered from the estimated pose \hat{P} of the real image I . Ideally, if the predicted pose \hat{P} is close to its ground truth pose P , and the novel view renderer produces realistic images, the rendered image \hat{I} should be indistinguishable from the real image.

In practice, we found the photometric-based supervision signal could be noisy in direct matching, when part of scene content changes. For example, random cars and pedestrians may appear through time or the NeRF rendering quality is imperfect. Therefore, we propose to measure the distance between images in feature space instead of in photometric space, given that the deep features are usually more robust to appearance changes and imperfect renderings.

Specifically, for an input image I and its pose estimation $\hat{P} = \mathcal{F}(I)$, a synthetic image $\hat{I} = \mathcal{H}(\hat{P}, \mathbf{y}_I)$ can be rendered using the pose estimation \hat{P} and the histogram embedding \mathbf{y}_I of the input image I . We then extract the feature map $M \in \mathbb{R}^{H_M \times W_M \times C_M}$ and $\tilde{M} \in \mathbb{R}^{H_M \times W_M \times C_M}$ for image I and \hat{I} respectively, where H_M and W_M are the spatial dimensions and C_M is the channel dimension of the feature maps. To measure the difference between two feature maps, we compute a cosine similarity between feature $m_i \in \mathbb{R}^{C_M}$ and $\tilde{m}_i \in \mathbb{R}^{C_M}$ for each feature location i :

$$\cos(m_i, \tilde{m}_i) = \frac{m_i \cdot \tilde{m}_i}{\|m_i\|_2 \cdot \|\tilde{m}_i\|_2}. \quad (1)$$

By minimizing the feature-metric direct matching loss $\mathcal{L}_{dm} = \sum_i (1 - \cos(m_i, \tilde{m}_i))$, the pose estimator \mathcal{F} can be trained in a semi-supervised manner (note no ground truth label required for the input image I).

Our direct feature matching may optionally follow the procedure of semi-supervised training proposed by MapNet+ [4] to improve pose estimation with unlabeled sequences captured in the same scene. Unlike [4], which requires sequential frames to enforce a relative geometric constraint using a VO algorithm, our feature-matching can be trained by images from arbitrary viewpoints without ground truth pose annotation. Our method can be used at train time with a batch of unlabeled images, or as a pose refiner for a single test image. In the latter case, our direct matching can also be regarded as a post-processing module. During the training stage, only the weights of the pose estimator will be updated, whereas the feature extractor part remains frozen to back-propagation.

Closing the Domain Gap We notice that synthetic images from NeRF are imperfect due to rendering artifacts or lack of adaption of the dynamic content of the scene, which leads to a domain gap between render and real images. This domain gap poses difficulties to our feature extractor (Fig. 3), which we expect to produce features far away if two views are from different poses and to produce similar features between a rendered view and a real image from the same pose.

Intuitively, we could simply enforce the feature extractor to produce similar features for a rendered image \hat{I} and a real image I via a distance function $d(\cdot)$ during training. However, this approach leads to model collapse [6], which motivates us to explore the original triplet loss:

$$\mathcal{L}_{triplet}^{ori} = \max \left\{ d(M_{real}^P, M_{syn}^P) - d(M_{real}^P, M_{syn}^{\bar{P}}) + \text{margin}, 0 \right\}, \quad (2)$$

where M_{real}^P and M_{syn}^P , the feature maps of a real image and a synthetic image at pose P , compose a positive pair, and $M_{syn}^{\bar{P}}$ is a feature map of a synthetic image rendered at an arbitrary pose \bar{P} other than the pose P .

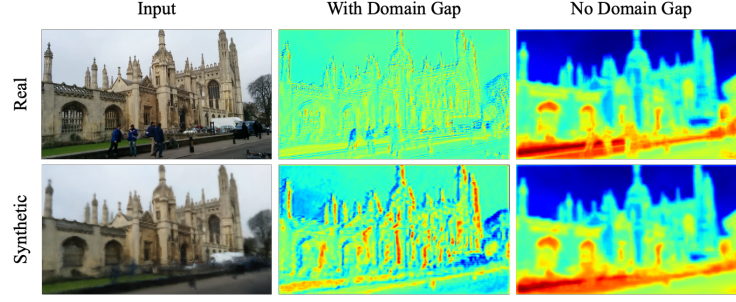


Fig. 3. A visual comparison of features before and after closing the domain gap. Ideally, a robust feature extractor shall produce indistinguishable features between real and rendered images from the same pose. Column 2/Column 3 are features trained without/with using our proposed $\mathcal{L}_{triplet}$ loss, where our method can effectively produce similar features across two domains.

With a closer look at the task of feature-metric direct matching, we implement a customized in-triplet mining which explores the minimum distances among negative pairs:

$$\mathcal{L}_{triplet} = \max \{d(M_{real}^P, M_{syn}^P) - q_{\ominus} + \text{margin}, 0\}, \quad (3)$$

where the positive pair is as same as Eq. (2) and q_{\ominus} is the minimum distance between four negative pairs:

$$q_{\ominus} = \min \left\{ d(M_{real}^P, M_{real}^{\bar{P}}), d(M_{real}^P, M_{syn}^{\bar{P}}), d(M_{syn}^P, M_{real}^{\bar{P}}), d(M_{syn}^P, M_{syn}^{\bar{P}}) \right\}, \quad (4)$$

which essentially takes the hardest negative pair among all matching pairs between synthetic images and real images that are in different camera poses. The margin value is set to 1.0 in our implementation. Since finding the minimum of negative pairs is non-differentiable, we implement the in-triplet mining as a prior step before $\mathcal{L}_{triplet}$ is computed.

Overall, to train the pose estimator and to obtain domain invariant and transformation sensitive property, we adapt a siamese-style training scheme as illustrated in Fig. 2a. Given an input image I and its ground truth pose P , a synthetic image \hat{I} can be rendered via the NVS module \mathcal{H} (assumed pre-trained) using the ground truth pose P . We then present both the real image I and the synthetic image \hat{I} to the pose estimator and the feature extractor, resulting in pose estimations \hat{P}_{real} and \hat{P}_{syn} and feature maps M_{real} and M_{syn} for the real image I and synthetic image \hat{I} , respectively. The training then is supervised via a combined loss function

$$\mathcal{L}_{DFNet} = \mathcal{L}_{triplet} + \mathcal{L}_{RVS} + \frac{1}{2}(\|P - \hat{P}_{real}\|_2 + \|P - \hat{P}_{syn}\|_2), \quad (5)$$

where $\|\cdot\|$ denotes a L_2 loss and \mathcal{L}_{RVS} is a supervision signal from our RVS training strategy, which we explain in Section 3.3.

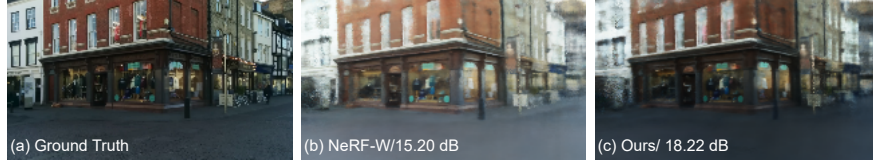


Fig. 4. Typically NeRF only renders views that reflect the appearance of its training sequences, as shown by NeRF-W’s synthetic view (b). However, in relocalization tasks, the query set may have different appearances or exposures to the train set. The proposed histogram-assisted NeRF (c) can render a more accurate appearance to the unseen query set (a) in both quantitative (PSNR) and visual comparisons. We refer to the supplementary for more examples.

3.2 Histogram-assisted NeRF

The DFNet pipeline relies on an NVS module that renders a synthetic image from which we extract a feature map and compare it with a real image. Theoretically, while the NVS module in our pipeline can be in any form as long as it provides high-quality novel view renderings, in practice, we found that due to the presence of auto exposure during image capturing, it is necessary to have a renderer that can render images in a compensated exposure condition. Although employing direct matching in feature space could mediate the exposure issue to some extent, we find decoupling the exposure issue from the domain adaption issue leads to better pose estimation results.

One off-the-shelf option is a recent work NeRF-W [18], which offers the ability to control rendered appearance via an appearance embedding that is based on frame indices. However, in the context of direct matching, since we aim to compare a real image with its synthetic version, we desire a more fine-grained exposure control to render an image that matches the exposure condition of the real image, as illustrated in Fig. 4.

To this end, we propose a novel view renderer histogram-assisted NeRF (Fig. 2b) which renders an image $\hat{I} = \mathcal{H}(P, \mathbf{y}_I)$ that matches the exposure level of a query real image I via a histogram embedding \mathbf{y}_I of the query image I at an arbitrary camera pose P . Specifically, our NeRF contains 3 components:

1. A base network \mathcal{H}_b that provides a density estimation σ_b and a hidden state \mathbf{z} for a coarse estimation: $[\sigma_b, \mathbf{z}] = \mathcal{H}_b(\gamma(\mathbf{x}))$.
2. A static network \mathcal{H}_s to model density σ_s and radiance \mathbf{c}_s for static structure and appearance: $[\sigma_s, \mathbf{c}_s] = \mathcal{H}_s(\mathbf{z}, \gamma(\mathbf{d}), \mathbf{y}_I)$.
3. A transient network \mathcal{H}_t to model density σ_t , radiance \mathbf{c}_t and an uncertainty estimation β for dynamic objects: $[\sigma_t, \mathbf{c}_t, \beta] = \mathcal{H}_t(\mathbf{z}, \mathbf{y}_I)$.

As for the input, \mathbf{x} is a 3D point and \mathbf{d} is a view angle that observes the 3D point, with both of them encoded by a positional encoding [10,35,20] operator $\gamma(\cdot)$ before injecting to each network.

During training, the coarse density estimation from the base network \mathcal{H}_b provides a distribution where the other two networks could sample more 3D points

near non-empty space accordingly. Both the static and the transient network are conditioned on a histogram-based embedding $\mathbf{y}_I \in \mathbb{R}^{C_y}$, which is mapped from a N_b bins histogram. The histogram is computed on the luma channel Y of a target image in YUV space. We found this approach works well in a direct matching context, not only in feature-metric space but also in photometric space.

We adopt a similar network structure and volumetric rendering method as in NeRF-W[18], to which we refer readers for more details.

3.3 Random View Synthesis

During the training of DFNet, we can generate training data by synthesis more views from randomly perturbed training poses. We refer this process as Random View Synthesis (RVS), and we use this data generation strategy to help the DFNet to better generalize to unseen views.

Specifically, given a training pose P , a perturbed pose P' can be generated around the training pose with a random translation noise of ψ meters and random rotation noise of ϕ degrees. A synthetic image $I' = \mathcal{H}(P', \mathbf{y}_{I_{nn}})$ is then rendered via histogram-assisted NeRF \mathcal{H} , with $\mathbf{y}_{I_{nn}}$ being the histogram embedding of the training image with the nearest training pose. The synthetic pose-image pair (P', I') is used as a training sample for the pose estimator to provide an additional supervision signal $\mathcal{L}_{RVS} = \|P' - \hat{P}'\|_2$, where $\hat{P}' = \mathcal{F}(I')$ is the pose estimation of the rendered image.

A key advantage of our method is efficiency in comparison with prior training sample generation methods. For example, LENS [22] generates high-resolution synthetic data with a maximum of 40s/image and requires complicated parameter settings in finding candidate poses within scene volumes. In contrast, our RVS is a lightweight strategy that seamlessly fits our DFNet training at a much cheaper cost (12.2 fps) and with fewer constraints in pose generation while being able to reach similar performance. We refer to Section 4.5 for more discussion.

4 Experiments

4.1 Implementation

We introduce the implementation details for histogram-assisted NeRF, DFNet, and direct feature matching. We also provide more details in the supplementary.

NeRF Our histogram-assisted NeRF model is trained with a re-aligned and re-centered pose in $SE(3)$, similar to Mildenhall *et al.* [20]. The image histogram bin size is set to $N_b = 10$ and embedded with a vector dimension of 50 for the static model and 20 for the transient model. We train the model with a learning rate of 5×10^{-4} and an exponential decay of 5×10^{-4} for 600 epochs.

DFNet Our DFNet adapts an ImageNet pre-trained VGG-16 [31] as the backbone, and an Adam optimizer with a learning rate of 1×10^{-4} is applied during training. For feature extraction, we extract $L = 3$ feature maps from the end of the encoder’s first, third, and fifth blocks before pooling layers. All final

Table 1. Pose regression results on 7-Scenes dataset. We compare DFNet and DFNet_{dm} (DFNet with feature-metric direct matching) with prior single-frame APR methods and unlabeled training methods, in median translation error (m) and rotation error (°). Note that MapNet+ and MapNet+PGO are sequential methods with unlabeled training. Numbers in **bold** represent the best performance.

	Methods	Chess	Fire	Heads	Office	Pumpkin	Kitchen	Stairs	Average
1-frame APR	PoseNet(PN)[14]	0.32/8.12	0.47/14.4	0.29/12.0	0.48/7.68	0.47/8.42	0.59/8.64	0.47/13.8	0.44/10.4
	PN Learn σ^2 [13]	0.14/4.50	0.27/11.8	0.18/12.1	0.20/5.77	0.25/4.82	0.24/5.52	0.37/10.6	0.24/7.87
	geo. PN[13]	0.13/4.48	0.27/11.3	0.17/13.0	0.19/5.55	0.26/4.75	0.23/5.35	0.35/12.4	0.23/8.12
	LSTM PN[36]	0.24/5.77	0.34/11.9	0.21/13.7	0.30/8.08	0.33/7.00	0.37/8.83	0.40/13.7	0.31/9.85
	Hourglass PN[19]	0.15/6.17	0.27/10.8	0.19/11.6	0.21/8.48	0.25/7.0	0.27/10.2	0.29/12.5	0.23/9.53
	BranchNet[38]	0.18/5.17	0.34/8.99	0.20/14.2	0.30/7.05	0.27/5.10	0.33/7.40	0.38/10.3	0.29/8.30
	MapNet[4]	0.08/3.25	0.27/11.7	0.18/13.3	0.17/5.15	0.22/4.02	0.23/ 4.93	0.30/12.1	0.21/7.77
	Direct-PN[5]	0.10/3.52	0.27/8.66	0.17/13.1	0.16/5.96	0.19/3.85	0.22/5.13	0.32/10.6	0.20/7.26
	TransPoseNet[29]	0.08/5.68	0.24/10.6	0.13/12.7	0.17/6.34	0.17/5.6	0.19/6.75	0.30/7.02	0.18/7.78
	MS-Transformer[28]	0.11/4.66	0.24/9.60	0.14/12.2	0.17/5.66	0.18/4.44	0.17/5.94	0.17/5.94	0.18/7.28
	DFNet (ours)	0.05/1.88	0.17/6.45	0.06/3.63	0.08/2.48	0.10/2.78	0.22/5.45	0.16/3.29	0.12/3.71
Unlabel Data	MapNet+ _(seq.) [4]	0.10/3.17	0.20/9.04	0.13/11.1	0.18/5.38	0.19/3.92	0.20/5.01	0.30/13.4	0.19/7.29
	MapNet+ _{PGO(seq.)} [4]	0.09/3.24	0.20/9.29	0.12/8.45	0.19/5.42	0.19/3.96	0.20/4.94	0.27/10.6	0.18/6.55
	Direct-PN+U[5]	0.09/2.77	0.16/4.87	0.10/6.64	0.17/5.04	0.19/3.59	0.19/4.79	0.24/8.52	0.16/5.17
	DFNet _{dm} (ours)	0.04/1.48	0.04/2.16	0.03/1.82	0.07/2.01	0.09/2.26	0.09/2.42	0.14/3.31	0.07/2.21

feature outputs are upsampled to the same size as the input image $H \times W$ with bilinear upsampling. For pose regression, we regresses the SE(3) camera pose with a fully connected layer. A singular value decomposition (SVD) is applied to ensure the rotation component of \hat{P} is normalized [5].

Direct Feature Matching To validate our feature-metric direct matching formulation, we follow the same procedure from MapNet+ [4] and Direct-PN+U [5], which use a portion of validation images without the ground truth poses for finetuning. When finetuning DFNet, we optimize the pose regression module \mathcal{F} solely based on the direct feature matching loss \mathcal{L}_{dm} . We set the batch size to 1 and the learning rate to 1×10^{-5} . For naming simplicity, we named our model trained with direct feature matching as DFNet_{dm}.

4.2 Evaluation on the 7-Scenes Dataset

We evaluate our method on an indoor camera localization dataset 7-Scenes [11,30]. The dataset consists of seven indoor scenes scaled from $1m^3$ to $18m^3$. Each scene contains 1000 to 7000 training sets and 1000 to 5000 validation sets. Both histogram-assisted NeRF and DFNet use subsampled training data with a spacing window $d = 5$ for scenes containing ≤ 2000 frames and $d = 10$ otherwise. RVS poses are sampled on the training pose, and the DFNet parameters are $t_\psi = 0.2m$, $r_\phi = 10^\circ$, and $d_{max} = 0.2m$. For fair comparison to other unlabeled training methods such as MapNet+ and Direct-PN, we finetune our DFNet_{dm} using the same amount of unlabeled samples, which is 1/5 or 1/10 of the sequences based on the spacing window above to ensure our method is not overfitting to the entire test sequences.

We compared our method quantitatively with prior single-frame APR methods and unlabeled training APR methods in Table 1. The results show that both our DFNet and DFNet_{dm} obtain superior accuracy, and DFNet_{dm} achieves

Table 2. Single-frame APR results on Cambridge dataset. We report the median position and orientation errors in $m/^\circ$ and the respective rankings over scene average as in [29,28]. The best results is highlighted in **bold**. For fair comparisons, we omit prior APR methods which did not publish results in Cambridge.

Methods	Kings	Hospital	Shop	Church	Average	Ranks	Final Rank
PoseNet(PN)[14]	1.66/4.86	2.62/4.90	1.41/7.18	2.45/7.96	2.04/6.23	9/9	9
PN Learn σ^2 [13]	0.99/1.06	2.17/2.94	1.05/3.97	1.49/3.43	1.43/2.85	6/3	5
geo. PN[13]	0.88/1.04	3.20/3.29	0.88/3.78	1.57/3.32	1.63/2.86	7/4	6
LSTM PN[36]	0.99/3.65	1.51/4.29	1.18/7.44	1.52/6.68	1.30/5.51	5/8	7
MapNet[4]	1.07/1.89	1.94/3.91	1.49/4.22	2.00/4.53	1.63/3.64	7/7	8
TransPoseNet[29]	0.60/2.43	1.45/3.08	0.55/3.49	1.09/4.94	0.91/3.50	2/6	3
MS-Transformer[28]	0.83/1.47	1.81/2.39	0.86/3.07	1.62/3.99	1.28/2.73	4/2	2
DFNet (ours)	0.73/2.37	2.00/2.98	0.67/2.21	1.37/4.03	1.19/2.90	3/5	3
DFNet _{dm} (ours)	0.43/0.87	0.46/0.87	0.16/0.59	0.50/1.49	0.39/0.96	1/1	1

Table 3. Comparison between our method and sequential-based APR methods and 3D structure-based methods.

	3D	Seq. APR				1-frame
Methods	AS[27]	MapNet +PGO[4]	CoordiNet[21]	CoordiNet +Lens[22]	VLocNet[34]	DFNet _{dm}
Chess	0.04/2.0	0.09/3.24	0.14/6.7	0.03/1.3	0.04/1.71	0.04/1.48
Fire	0.03/1.5	0.20/9.29	0.27/11.6	0.10/3.7	0.04/5.34	0.04/2.16
Heads	0.02/1.5	0.12/8.45	0.13/13.6	0.07/5.8	0.05/6.65	0.03/1.82
Office	0.09/3.6	0.19/5.42	0.21/8.6	0.07/1.9	0.04/1.95	0.07/2.01
Pumpkin	0.08/3.1	0.19/3.96	0.25/7.2	0.08/2.2	0.04/2.28	0.09/2.26
Kitchen	0.07/3.4	0.20/4.94	0.26/7.5	0.09/2.2	0.04/2.21	0.09/2.42
Stairs	0.03/2.2	0.27/10.6	0.28/12.9	0.14/3.6	0.10/6.48	0.14/3.31
Average	0.05/2.5	0.18/6.55	0.22/9.7	0.08/3.0	0.05/3.80	0.07/ 2.21
Kings	0.42/0.6	-	0.70/2.92	0.33/0.5	0.84/1.42	0.43/0.87
Hospital	0.44/1.0	-	0.97/2.08	0.44/0.9	1.08/2.41	0.46/0.87
Shop	0.12/0.4	-	0.73/4.69	0.27/1.6	0.59/3.53	0.16/0.59
Church	0.19/0.5	-	1.32/3.56	0.53/1.6	0.63/3.91	0.50/1.49
Average	0.29/0.63	-	0.92/2.58	0.39/1.15	0.78/2.82	0.39/0.96

56% and 57% improvement over averaged median translation and rotation errors compared to prior SOTA performance.

4.3 Evaluation on Cambridge Dataset

We further compare our approach on four outdoor scenes from the Cambridge Landmarks [14] dataset, scaling from $875m^2$ to $5600m^2$. Each scene contains from 200+ to 1500 training samples. Our models are trained with 50% of training data, and DFNet’s RVS are $t_\psi = 3m$, $r_\phi = 7.5^\circ$, and $d_{max} = 1m$. For finetuning DFNet_{dm} with unlabeled data, we use 50% of the unlabeled validation sequence since fewer validation sets are available than 7-Scenes. Table 2 shows a comparison between our approach and prior single-frame APR methods, which omits prior APR methods that did not report results in Cambridge. We observe that our DFNet_{dm} outperforms other methods significantly (60%+ in scene average), which further proves the effectiveness of our approach.

Table 4. (a) The effect of various level of features on DFNet_{dm} result. Letter F, M, and C denote features extracted from fine, middle, and coarse levels in DFNet. **(b)** Ablation on DFNet (upper part) and histogram-assisted NeRF in photometric direct matching (lower part). DFM denotes Direct Feature Matching.

(a) Feature level vs. pose error		(b) Ablation	
Feature Level	DFNet _{dm} (ShopFacade)	Method	Shop Facade
F	0.15m, 0.64°	DFNet w/ $\mathcal{L}_{triplet}^{ori}$	1.49m/5.80°
F+M	0.19m, 0.77°	+RVS	0.86m/4.05°
F+M+C	0.20m, 0.77°	+ $\mathcal{L}_{triplet}$	0.72m/2.58°
		+DFM (NeRF-W)	0.43m/1.62°
		+DFM (NeRF-Hist)	0.15m/0.65°
		Direct-PN	1.10m/4.25°
		Direct-PN+U	1.41m/6.97°
		+ NeRF-Hist	0.72m/3.39°

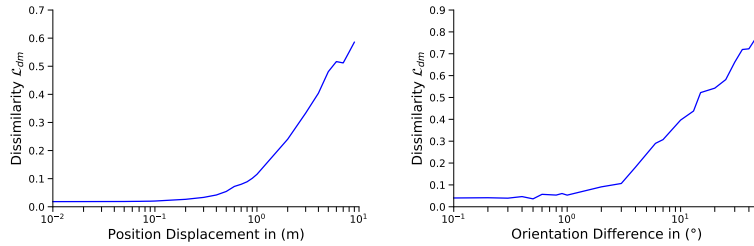


Fig. 5. Pose difference vs. feature dissimilarity. X-axis: camera position (**left**) and orientation difference (**right**) between a real image and a rendered image. Y-axis: feature dissimilarity \mathcal{L}_{dm} . Our direct feature matching loss \mathcal{L}_{dm} is closely related to pose error, leading to effective training of the APR method.

4.4 Comparison to Sequential APR and 3D Approaches

Table 3 compares our method to other types of relocalization approaches, such as several state-of-the-art sequential-based APR approaches and 3D structure-based method Active Search [27]. We notice that our DFNet_{dm} outperforms most sequential-based APR methods except the translation error of VLocNet [34] on 7-Scenes in terms of the scene average performance. However, we still achieve superior accuracy than VLocNet in 7 out of 11 scenes. For the first time, the performance of single-image APR is comparable to 3D-structure methods. Our DFNet_{dm} is slightly more accurate than Active Search [27] in average rotation error of 7-scenes. However, our method is still slightly behind in terms of translation error and Cambridge errors although by smaller margins.

4.5 Ablation Study

Effectiveness of Direct Feature Matching We run a toy example of direct feature matching on Shop Facade using finest features and combinations of multi-level features, as in Table 4(a). We discover that finer-level features are more

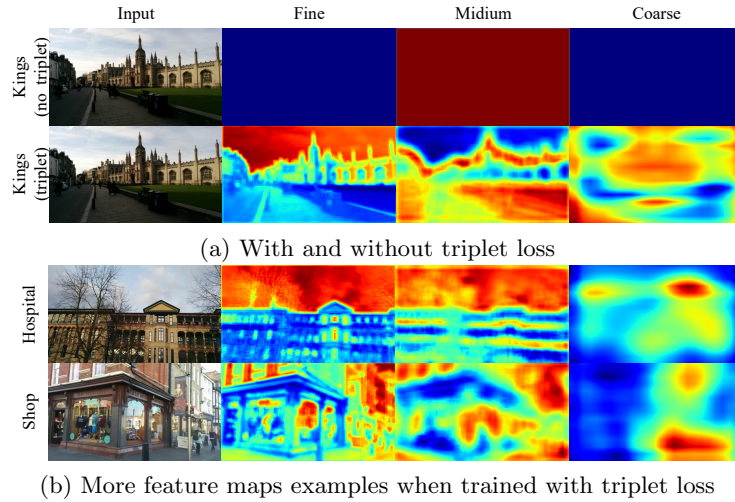


Fig. 6. (a) Top row: feature collapsing when training DFNet on Kings without using triplet loss. Bottom row: training DFNet with triplet loss can avoid the feature collapsing issue. (b) Feature maps of other scenes in Cambridge when training with triplet loss. We show that more refined level features consistently contain more meaningful details and, therefore more beneficial to use for direct feature matching.

Table 5. Data generation strategy comparison: RVS vs. LENS [22] on 7-Scenes. An EfficientNet backbone (as in LENS) is used in DFNet for a fair comparison. Our RVS strategy obtains a comparable results to LENS while using much less training data and rendering in much lower resolution, enabling online training.

Model	Backbone Top-1	Pose Error Acc.	Real Data Quantity/Epoch	Synthetic Data Quantity/Epoch	Synthetic Resolution	Rendering Cost	Generation Mode
DFNet(VGG16)	71.59%	0.12/3.71 (m/degree)	10-20%	10-20%	Low	Cheap	Online
DFNet(EB0)	76.3%	0.08/3.47	10-20%	10-20%	Low	Cheap	Online
LENS(EB3)	81.1%	0.08/3.00	71%-100%	710%-1000%	High	Expensive	Offline

helpful for direct matching. We believe this to be due to their capability to preserve high frequency details and sharper contents, as shown in (Fig. 6(b)). This explains why we only use the finest feature in the feature-metric direct matching implementation. Furthermore, Fig. 5 shows how the direct matching loss \mathcal{L}_{dm} successfully correlates the pose differences to the feature similarity between real images and rendered images.

Features Collapse We demonstrate the difference when training DFNet’s feature extractor with and without triplet loss in Fig. 6(a). We replace our triplet loss with a mean square error (MSE) loss for the without triplet loss case. Intuitively, losses that only minimize positive sample distances such as MSE, L_2 , or L_1 losses may lead to feature collapsing [6] since the feature extraction blocks in DFNet are likely to learn to cheat. On the other hand, using triplet loss super-

vised with additional negative samples works well for extracting dense domain invariant features.

Summary of Ablation We break down our design decisions to show how each component contributes to the pose regression accuracy in Table 4(b). We start with training an DFNet model using with standard triplet loss without mining. The performance improves noticeably when we add the RVS. We also see around 16%/36% gain in translation and rotation errors when adding the customized triplet loss $\mathcal{L}_{triplet}$. We then validate our DFNet_{dm}’s direct feature matching (DFM), which further reduces error significantly. The DFM approach with histogram-assisted NeRF outperforms the NeRF-W one, which validates the effectiveness of our histogram embedding design. Finally, we attempt to train a Direct-PN+U model with our histogram-assisted NeRF modification. Our results show that the photometric direct matching-based method that can benefit from our new NVS method, though the pose estimation accuracy is worse than our feature-metric direct matching method.

Effectiveness of RVS Table 5 shows a comparison between our online RVS strategy with another peer work LENS [22] that uses NeRF data generation for APR training. Although both data generation methods effectively improve APR performance, our RVS strategy is a much cheaper alternative requiring lower rendering resolution (80x60 vs. 320x240 [22]) and fewer data. We are able to reach similar performance with LENS when we replace our VGG16 backbone with an EfficientNet-B0 [33], which proves that a simpler data generation strategy could also effectively improves APR methods.

5 Conclusion

In summary, we introduce an Absolute Pose Regression (APR) pipeline for camera re-localization. Specifically: 1) we propose a histogram-assisted NeRF to compensate dramatic exposure variance in large scale scene with challenging exposure conditions. The histogram-assisted NeRF, serving as a novel view renderer, enables a direct matching training scheme; 2) we explore a direct matching scheme in feature space, leading to a more robust performance than the photometric approach, and address a domain gap issue that arises when matching real images with synthetic images via a contrastive learning scheme; 3) we devise an efficient data generation strategy, which proposes pseudo training poses around existing training trajectories, leading to better generalization capability to unseen data. As a result, our method achieves a state-of-the-art accuracy by outperforming existing single-image APR methods by as much as 56%, comparable to 3D structure-based methods.

Acknowledgments The authors thank Michael Hobley, Theo Costain, Lixiong Chen, and Kejie Li for their thoughtful comments. Shuai Chen was supported by gift funding from Huawei.

References

1. Brachmann, E., Krull, A., Nowozin, S., Shotton, J., Michel, F., Gumhold, S., Rother, C.: DSAC - Differentiable RANSAC for Camera Localization. In: CVPR (2017)
2. Brachmann, E., Rother, C.: Learning Less is More - 6D Camera Localization via 3D Surface Regression. In: CVPR (2018)
3. Brachmann, E., Rother, C.: Visual camera re-localization from RGB and RGB-D images using DSAC. arXiv (2020)
4. Brahmbhatt, S., Gu, J., Kim, K., Hays, J., Kautz, J.: Geometry-Aware Learning of Maps for Camera Localization. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2018)
5. Chen, S., Wang, Z., Prisacariu, V.: Direct-PoseNet: Absolute pose regression with photometric consistency. In: 3DV (2021)
6. Chen, X., He, K.: Exploring simple siamese representation learning. In: CVPR (2021)
7. DeTone, D., Malisiewicz, T., Rabinovich, A.: Superpoint: Self-supervised interest point detection and description. In: CVPRW (2018)
8. Engel, J., Koltun, V., Cremers, D.: DSO: Direct sparse odometry. In: IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI) (2017)
9. Engel, J., Schops, T., Cremers, D.: Semi-dense visual odometry for a monocular camera. In: In Proceedings of IEEE International Conference on Computer Vision (ICCV) (2013)
10. Gehring, J., Auli, M., Grangier, D., Yarats, D., Dauphin, Y.N.: Convolutional sequence to sequence learning. In: ICML (2017)
11. Glocker, B., Izadi, S., Shotton, J., Criminisi, A.: Real-time rgb-d camera relocation. In: International Symposium on Mixed and Augmented Reality (ISMAR) (2013)
12. Kendall, A., Cipolla, R.: Modelling uncertainty in deep learning for camera relocation. In: ICRA (2016)
13. Kendall, A., Cipolla, R.: Geometric loss functions for camera pose regression with deep learning. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017)
14. Kendall, A., Grimes, M., Cipolla, R.: Posenet: A convolutional network for real-time 6-dof camera relocation. In: International Conference on Computer Vision (2015)
15. Kendall, A., Cipolla, R.: Modelling uncertainty in deep learning for camera relocation. In: ICRA (2016)
16. Li, X., Han, K., Li, S., Prisacariu, V.: Dual-resolution correspondence networks. In: NeurIPS (2020)
17. Lindenberger, P., Sarlin, P.E., Larsson, V., Pollefeys, M.: Pixel-Perfect Structure-from-Motion with Featuremetric Refinement. ICCV (2021)
18. Martin-Brualla, R., Radwan, N., Sajjadi, M.S.M., Barron, J.T., Dosovitskiy, A., Duckworth, D.: NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections. In: CVPR (2021)
19. Melekhov, I., Ylioinas, J., Kannala, J., Rahtu, E.: Image-based localization using hourglass networks. In: ICCV Workshops (2017)
20. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: NeRF Representing scenes as neural radiance fields for view synthesis. In: ECCV (2020)

21. Moreau, A., Piasco, N., Tsishkou, D., Stanciulescu, B., de La Fortelle, A.: Co-ordinet: uncertainty-aware pose regressor for reliable vehicle localization. In: arxiv preprint, arxiv:2103.10796 (2021)
22. Moreau, A., Piasco, N., Tsishkou, D., Stanciulescu, B., de La Fortelle, A.: LENS: Localization enhanced by nerf synthesis. In: CoRL (2021)
23. Ng, T., Lopez-Rodriguez, A., Balntas, V., Mikolajczyk, K.: Reassessing the limitations of cnn methods for camera pose regression. In: arXiv preprint arXiv:2108.07260 (2021)
24. Purkait, P., Zhao, C., Zach, C.: Synthetic view generation for absolute pose regression and image synthesis. In: BMVC (2018)
25. Sarlin, P.E., DeTone, D., Malisiewicz, T., Rabinovich, A.: Superglue: Learning feature matching with graph neural networks. In: CVPR (2020)
26. Sarlin, P.E., Unagar, A., Larsson, M., Germain, H., Toft, C., Larsson, V., Pollefeys, M., Lepetit, V., Hammarstrand, L., Kahl, F., Sattler, T.: Back to the Feature: Learning robust camera localization from pixels to pose. In: CVPR (2021)
27. Sattler, T., Leibe, B., Kobbelt, L.: Improving image-based localization by active correspondence search. In: European conference on computer vision (2012)
28. Shavit, Y., Ferens, R., Keller, Y.: Learning multi-scene absolute pose regression with transformers. In: ICCV (2021)
29. Shavit, Y., Ferens, R., Keller, Y.: Paying attention to activation maps in camera pose regression. In: arXiv preprint arXiv:2103.11477 (2021)
30. Shotton, J., Glocker, B., Zach, C., Izadi, S., Criminisi, A., Fitzgibbon, A.: Scene coordinate regression forests for camera relocalization in rgb-d images. In: CVPR (2013)
31. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. In: ICLR (2015)
32. Taira, H., Okutomi, M., Sattler, T., Cimpoi, M., Pollefeys, M., Sivic, J., Pajdla, T., , Torii, A.: InLoc: Indoor Visual Localization with Dense Matching and View Synthesis. CVPR (2018)
33. Tan, M., EfficientNet, Q.L.: Efficientnet: Rethinking model scaling for convolutional neural networks. In: PMLR (2019)
34. Valada, A., Radwan, N., Burgard, W.: Deep auxiliary learning for visual localization and odometry. In: ICRA (2018)
35. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. NeurIPS (2017)
36. Walch, F., Hazirbas, C., Leal-Taixe, L., Sattler, T., Hilsenbeck, S., Cremers, D.: Image-based localization using lstms for structured feature correlation. In: International Conference on Computer Vision (2017)
37. Wang, Z., Wu, S., Xie, W., Chen, M., Prisacariu, V.A.: NeRF—: Neural radiance fields without known camera parameters. arXiv preprint arXiv:2102.07064 (2021)
38. Wu, J., Ma, L., Hu, X.: Delving Deeper into Convolutional Neural Networks for Camera Relocalization. In: ICRA (2017)
39. Yen-Chen, L., Florence, P., Barron, J.T., Rodriguez, A., Isola, P., Lin, T.Y.: inerf: Inverting neural radiance fields for pose estimation. In: arxiv arXiv:2012.05877 (2020)

6 Supplementary

6.1 Implementation

Histogram-assisted NeRF Here, we provide more implementation details of our methods. As we mentioned in Section 3.3 of the main paper, our histogram-assisted NeRF model renders at a speed of 12.2 fps (benchmarked by a 3080Ti GPU) to achieve online RVS training. In order to achieve a balanced trade-off between speed and quality, we choose to render small images with a shorter side of 60 pixels. In addition, we set the NeRF model architecture to 64 coarse and 64 fine sampling with an MLP width of 128.

DFNet Our DFNet takes an image input with a shorter side of 240 pixels. For feature extractor module \mathcal{G} of DFNet, features are fed through a Conv-Relu-Conv-Batch Norm architecture with 64 kernels and 128 output channels. The DFNet is trained with a batch size of 4 or 8, depending on the GPU’s memory. We implement an early stopping strategy with a patient value of 200 and schedule the learning rate decay of 0.95 when validation loss plateaus for every 50 epochs. For every $N = 20$ epochs, we will randomly generate the same amount of views as the training sample size using RVS.

Direct Feature Matching To train the DFNet_{dm} model with feature-metric direct matching using unlabeled data, we set the batch size to 1 and the learning rate to 1×10^{-5} with the same early stopping strategy mentioned above. We discover that only low-level features (i.e., features from the first blocks of VGG) are needed to achieve the best performance, which we discussed earlier in Section 4.5 of the main paper.

6.2 Visualization

Qualitative Comparison on 7-Scenes We show a selection of qualitative comparisons on the 7-Scenes dataset with several baseline APR methods [13,4,5] in Fig. 7. We also encourage our readers to check out our supplementary video, in which we rendered views of the predicted pose using NeRF synthesis.

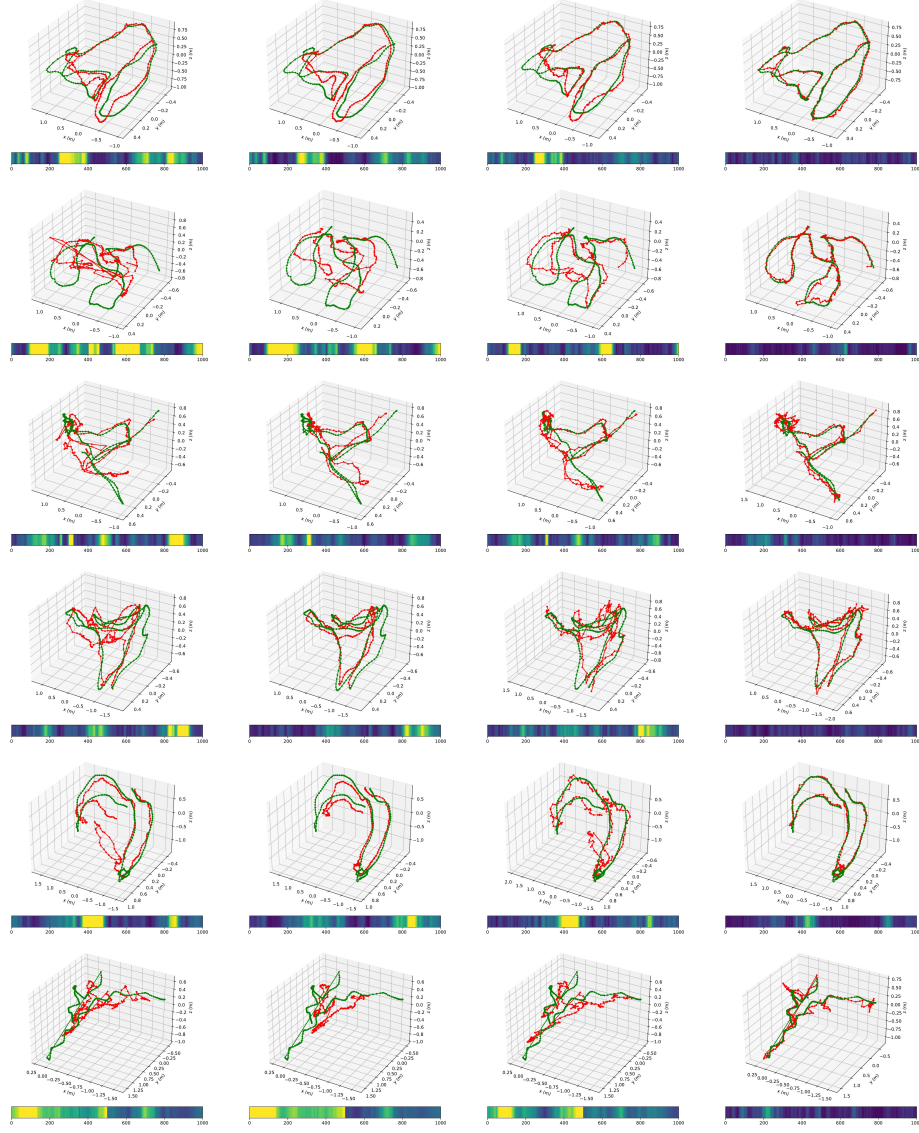
NeRF-W vs. Histogram-assisted NeRF In real-life camera localization applications, since training and testing data are likely to be taken from different sequences, camera exposures, or time of the day, our histogram-assisted NeRF would be more helpful to render accurate appearance Fig. 8. We experimentally found our histogram-assisted NeRF is helpful in both photometric matching and feature-metric matching approaches.

6.3 Additional Discussion

Photometric Distortion As discussed in section 3.2 of the main paper, photometric matching relies on RGB-wise differences between the query and rendered images. However, if those images appear in different lighting/exposure conditions, the photometric loss will fail due to large RGB-wise differences even under the same camera pose. Previous photometric matching approaches, such as Direct-PN+U, perform worse in pose estimation when using unlabeled data with large appearance variations from the training sequences (refer to the lower part of main paper Table 4b). We observed that such degradation is consistent in other outdoor scenes.

Two Properties of Domain Invariant Features We had two clear goals for designing our robust feature extractor: (1) We want the extracted features to be sensitive to pose changes. (2) We want the features to be indistinguishable between real and rendered image features from the same pose (Close the Domain Gap). Our first goal is achieved by the L_2 pose loss supervision, which ensures the features are closely related to the pose regression task. We specifically design the Feature Extractor to share the backbone with the Pose Module (see main paper Fig 2a). Although the deeper layer features may lose semantic meaning, we observe that those features can respond to pose changes.

The triplet loss is primarily designed to achieve the second goal without feature collapse in the training process. We previously tried to force real and rendered image features to be the same by using MSE/ L_2 losses, leading to feature collapse (main paper Fig 6a). This is because the pink layers in main paper Fig 2a, despite being shallow, are likely to learn to cheat since those layers are not supervised by other meaningful losses. Thus, we introduce the triplet loss to prevent features collapsing. We experimentally find that the proposed in-triplet mining adds extra robustness to both feature extraction and pose regression and leads to better APR performance overall. Such observation could hint that removing the domain gap benefits APR training when using extra randomly generated synthetic training data.



(a) PoseNet[14,12,13] (b) MapNet+PGO[4] (c) Direct-PN+U[5] (d) DFNet_{dm}

Fig. 7. Qualitative comparison on the 7-Scenes dataset. The 3D plots show the camera positions, **green** for ground truth and **red** for predictions. The bottom color bar represents rotational errors for each subplot, where yellow means large error and blue means small error for each test sequence. Sequence names from top to bottom are: Chess-seq-03, Fire-seq-04, Office-seq-07, Kitchen-seq-06, Kitchen-seq-12, Stairs-all.

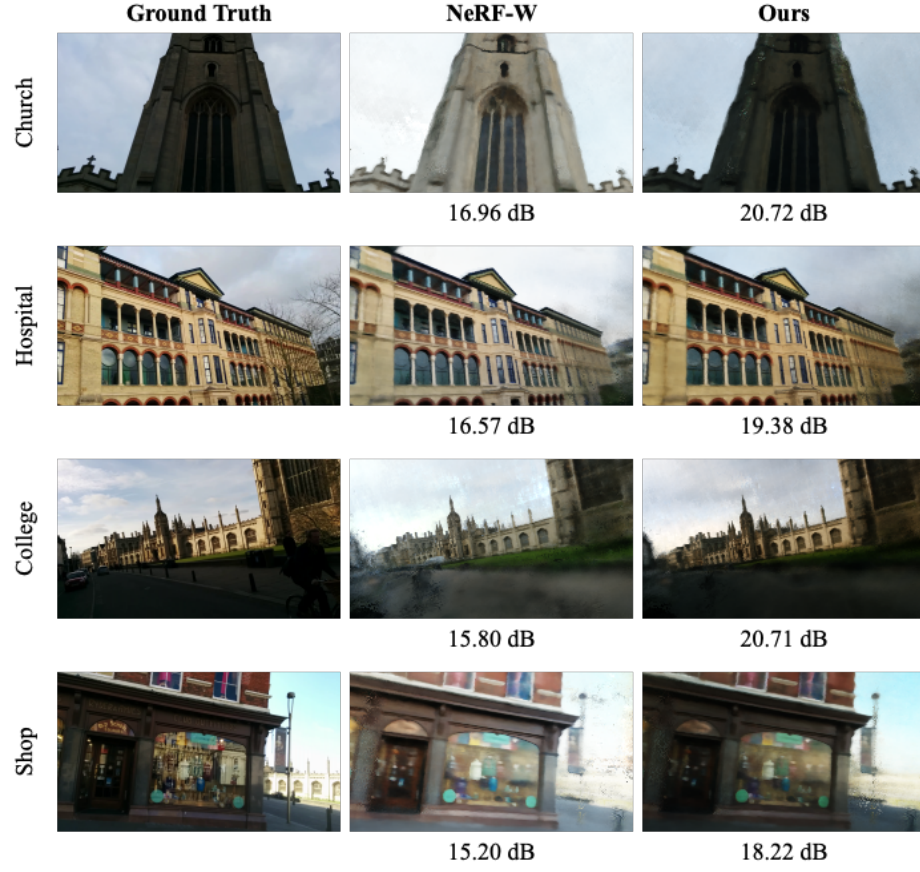


Fig. 8. A visual comparison between NeRF-W and our histogram-assisted NeRF on the testing sequences of Cambridge Landmarks dataset. The corresponding scene’s test PSNR is displayed at the bottom of each sub-figure.