Direct-PoseNet: Absolute Pose Regression with Photometric Consistency

Shuai Chen Zirui Wang Victor Prisacariu
Active Vision Lab, University of Oxford
{shuaic, ryan, victor}@robots.ox.ac.uk

2021 International Conference on 3D Vision (3DV) | 978-1-6654-2688-6/21/$31.00 ©2021 IEEE | DOI: 10.1109/3DV53792.2021.00125

Abstract

We present a relocalization pipeline, which combines an absolute pose regression (APR) network with a novel view synthesis based direct matching module, offering superior accuracy while maintaining low inference time. Our contribution is twofold: i) we design a direct matching module that supplies a photometric supervision signal to refine the pose regression network via differentiable rendering; ii) we show that our method can easily cope with additional unlabeled data without the need for external supervision such as traditional visual odometry or pose graph optimization. As a result, our method achieves state-of-the-art performance among all other single-image APR methods on the 7-Scenes benchmark and the LLFF dataset.

1. Introduction

Camera localization is a classical problem in computer vision and robotics. It is a core component for many applications such as virtual and augmented reality, indoor navigation systems, and autonomous driving. A typical visual-based localization algorithm is designed to determine the camera's 6-DOF positions and orientations from taking as input an RGB or RGB-D image.

The classical approach to solve this problem is built upon finding 2D-3D correspondences [2, 3, 5, 47, 48, 51, 55] between 2D image position and 3D points in space. Then an n-point pose (PnP) solver is applied to the 2D-3D matches inside a RANSAC [4, 8, 12, 44] loop. Traditionally, 2D-3D matches can be found using local feature descriptor matching, and many approaches require depth or structure-from-motion (SfM) reconstruction to build robust 3D geometric correlations [46, 48]. Recent methods use machine learning to regress 3D scene coordinates from image patches [2, 3, 5, 51] directly. Overall, 3D structure-based methods still achieve state-of-the-art (SOTA) accuracy, as discussed by Sattler *et al*. [49]. However, the presence of highly accurate depth images or SfM models is not universally available in real-life applications, especially for many consumer-grade devices such as smartphones or tablets. Most deep 3D structure-based methods are computation resources intensive and cannot easily achieve real-time inference with SOTA accuracy constraints.

Another line of approaches is deep learning-based pose regression [6, 20, 21, 22, 30, 35, 43, 59, 58, 61], also known as absolute pose regression (APR). These approaches propose to train a scene-specific deep neural network to predict 6-DoF camera pose relative to a scene directly from images. Despite obtaining inferior performances in localization benchmarks, it has gained popularity due to its high efficiency and simplicity by learning the full localization pipeline in a Convolutional Neural Network (CNN). The end-to-end approach has several appealing features compared to 3D structure-based methods: (1) most APR algorithms display great portability for commercial deployment at applications where fast and reliable performance is crucial. For example, the groundwork PoseNet [22] runs its entire process in less than $6ms$. (2) the CNN only requires RGB images input and does not rely on depth maps or SfM reconstructions, which is less hardware constrained. (3) it keeps a low memory footprint in megabytes regardless of scene sizes.

Despite these benefits, drawbacks of the APR method are also apparent. It is known to be prone to overfit the training set and significantly less accurate than structure-based methods, as shown in Sattler *et al*. [49] and Shavit *et al*. [52] . Both studies suggest that scene geometry is key for obtaining accurate pose estimation. Prior efforts have tried to add geometric constraints by finding relative pose [1, 6, 43, 58] or using reprojection error [21]. Nevertheless, it is clear that existing single image APR solutions are not yet able to compete with structure-based methods.

We address the problem of single-image APR by introducing direct matching supervision inspired by direct Visual Odometry (VO) approaches [10, 11]. The key intuition is that the predicted pose error is inversely proportional to the visual similarity between the query image and a rendering of the 3D scene of the relocalized pose. The proposed APR framework improves pose regression using direct matching supervised by the photometric similarity between the input query image and the rendered image of the scene using the

978-1-6654-2688-6/21/$31.00 ©2021 IEEE
DOI 10.1109/3DV53792.2021.00125

predicted pose. In the testing stage, our method runs like a standard APR method without extra computational cost. To our knowledge, this paper is the first camera pose regression method to use direct matching/photometric supervision. We summarize our contributions as follows:

- We introduce a novel camera relocalization pipeline consisting of a pose regression network and a direct matching module such that network learning is supervised by not only the traditional pose regression loss, but also a photometric loss.
- We show how unlabeled images can be leveraged using photometric loss in the direct matching module to further improve the pose regression performance without extra supervision, such as relative pose constraints.

With contributions above, our method achieves state-of-the-art performance in single-image APR on 7-Scenes benchmark and LLFF dataset.

This paper is organized as follows: we introduce existing APR methods and other related work in Section 2. Our relocalization pipeline is detailed in Section 3, with experimental results and analysis discussed in Section 4. Section 5 concludes our work.

2. Related Work

Absolute Pose Regression Absolute pose regression methods typically require a CNN classifier that has been pre-trained from the image classification dataset. It then uses transfer learning to fine-tune the feature extractors to regress the camera pose from one or more given image sequences. To get a thorough review in this area, the interested reads is refered to Sattler *et al*. [49].

The common practice in this area is introduced by PoseNet [22]. A simple pose regressor can take an arbitrary RGB image as the input and learn to regress the correspondent camera position and orientation. Successors of PoseNet focus to improve the framework in several aspects. [30, 59, 61] seek to enhance network architectures. LSTM PoseNet [59] combines LSTM with CNN to reduce feature dimensions for pose regression. Hourglass PoseNet [30] adapts an encoder-decoder style backbone. BranchNet [61] uses a multi-task CNN where low-level common features are extracted before splitting the network into two separate branches to predict the camera position and orientation. Kendall and Cipolla [21] proposed learnable weights to sidestep hyperparameter tunning that balances the translation and rotation loss in PoseNet. Besides learning the optimal weight between losses, they attempt to leverage scene geometry for pose regression by formulating a reprojection error between the ground-truth pose and predicted pose. While we share the same insight that geometry would help pose regression, we differ from theirs in that 1) the scene geometry is implicitly represented in a novel view synthesis-enabled direct matching module; 2) we introduce a dense pixel-level photometric supervision.

Unlabeled training in APR Rather than only training on images with ground-truth pose annotation, MapNet [6] is able to train on unlabeled video frames by adding pairwise geometric constraints between video frames using additional VO algorithms [10, 11]. During inference, it can utilize pose graph optimization (PGO) for post-processing to further boost the performance. We also recognize the importance of using additional unlabeled data in this work. However, instead of acquiring constraints from an additional VO algorithm or PGO, which assumes video frames are available, our method can train on images from arbitrary viewpoints by minimizing the photometric loss between the rendered images and the input images.

Direct Matching for Motion Estimation Direct matching methods or direct methods refer to the commonly used methods to recover camera motions by directly measuring image intensities [18] in VO and simultaneous localization and mapping (SLAM) systems. In contrast to the feature-based methods [9, 19, 24, 25] that minimize reprojection errors based on corresponding features among frames, direct methods exploit all information in the image and recover camera motions by minimizing the photometric error. Compared to feature-based methods, direct methods are often more reliable in sparse textured environments and do not have feature extraction operations to add in computation costs. DTAM [36], REMODE [42], and LSD-SLAM [11] employ dense reconstruction using direct methods. SVO [13, 14] proposes a hybrid approach to implement direct VO motion estimation at the SLAM frontend and uses feature-based methods in the backend mapping thread. DSO [10] further proposes a sparse and direct method. This paper is partly inspired by direct methods and adopts photometric supervision in training the pose regression network. However, our method is different from direct VO methods in several aspects. First, our method does not compute photometric errors based on image intensities but RGB differences. Second, our method provides absolute pose estimation using a single image, but the methods above are designed to take a pair of neighboring frames for computing relative motion.

Novel View Synthesis Novel view synthesis (NVS) is a long-standing problem in computer graphics. It aims to generate novel camera perspectives based on image samples of the scenes [56]. Early works in this field can be traced back to nearly 30 years ago when some required using densely captured views of the scene [17, 26], and others [7] interpolate novel views using image warping. Recently, novel view

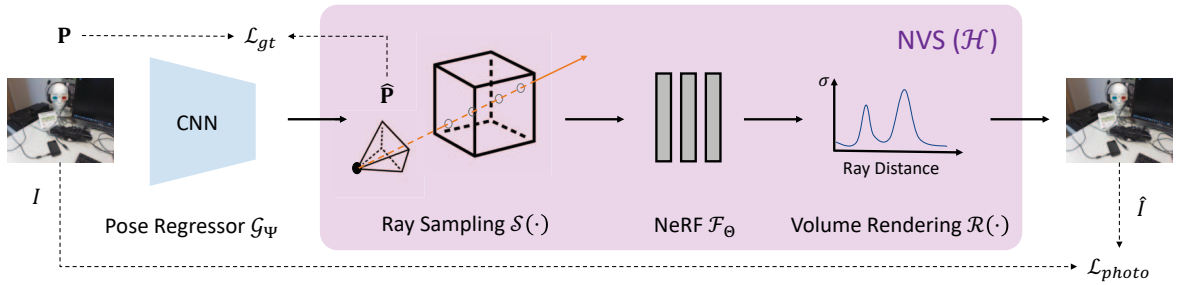


Figure 1: Overview of our proposed training pipeline. Given an input image I, the pose regressor $\mathcal{G}_\Psi$ predicts a pose estimation $\hat{\mathbf{P}}$, from which the NVS system $\mathcal{H}$ renders a synthetic image $\hat{I}$, supplying our direct matching supervision signal $\mathcal{L}_{photo}$ and refining $\mathcal{G}_\Psi$ along with the ground truth supervision $\mathcal{L}_{gt}$.

synthesis has made rapid progress in achieving photorealistic view synthesis [27, 28, 29, 32, 34, 39, 41, 50, 53, 57] with sparser view samples, thanks to recent development in neural 3D shape representation [31, 37, 38, 54]. We select a recent popular approach from Mildenhall *et al*. [34] to build our camera re-localization training pipeline in this work. Specifically, we incorporate NeRF architecture to provide photometric supervision to the pose regression model. We consider two other NeRF-based works that are able to estimate camera pose related to our paper: iNeRF [62] estimates pose iteratively by inverting a pre-trained NeRF model on the test images. Wang *et al*. [60] show that 3D scene representation and camera poses can be jointly optimized within a NeRF framework. Nonetheless, the fundamental difference of the proposed method to theirs is that we only use differentiable rendering to compute photometric loss in training of our pose regressor. After training, we are able to predict camera pose in a single network forward pass, whereas their methods require iterative optimization in test time.

3. Method

Fig. 1 illustrates our proposed relocalization pipeline, which consists of an NVS-enabled direct matching module and a pose regression network. We aim to predict a camera pose $\hat{v}$ for an input image I via a pose regression network $\mathcal{G}_\Psi$, which is supervised by a novel direct matching signal along with ground truth poses during training. At test time, only the pose regression network is required, ensuring rapid inference while offering superior relocalization accuracy.

This section is organized as follows: Section 3.1 details our direct matching module. A full system setup is explained in Section 3.2. To explore the possibility of utilizing more data, a further unlabeled training scheme is explained in Section 3.3.

3.1. Direct Matching

Direct matching is a common approach in traditional SLAM and VO systems [10, 11, 13, 14, 36, 42]. It refers to the process that optimizes camera poses via minimizing a photometric loss. In this work, we adapt the direct matching concept and apply it to assist the training of our pose regression network. Concretely, assuming a pre-trained pose regression network $\mathcal{G}_\Psi$ and an NVS system $\mathcal{H}$ are available, given an image I captured at viewpoint v and its pose estimation $\hat{v} = \mathcal{G}_\Psi(I)$, our direct matching module constrains Ψ via minimizing the photometric difference $\mathcal{L}_{photo}(\hat{I}, I)$ between a synthetic image $\hat{I} = \mathcal{H}(\hat{v})$ rendered by the NVS model $\mathcal{H}$ at viewpoint $\hat{v}$, and its true observation I:

$$\mathcal{L}_{photo}(\hat{I}, I) = \|\hat{I} - I\|_2. \tag{1}$$

Any NVS system with a differentiable renderer could be selected as the NVS system $\mathcal{H}$. For simplicity, we choose the NeRF [34] in this work, due to the high quality reconstructions it produces.

From a high-level point of view, NeRF-based NVS requires three main components: 1) a neural radiance field function $\mathcal{F}_\Theta$ that models a 3D volume, 2) a differentiable volume renderer $\mathcal{R}(\cdot)$ that enables back-propagation, and 3) a viewpoint-dependent volume sampling function $\mathcal{S}(\cdot)$ that provides 3D sample locations and view directions for $\mathcal{F}_\Theta$ and $\mathcal{R}(\cdot)$ given a camera pose. Consequently, an image rendered from an NVS system $\mathcal{H}$ using NeRF can be formulated by:

$$\hat{I}_v = \mathcal{H}(v) \triangleq \mathcal{R}(\mathcal{F}_\Theta, \mathcal{S}(v)), \tag{2}$$

where $\hat{I}_v$ denotes a synthetic image rendered at a pose v, and all operations above are differentiable. As a result, our direct matching system updates Ψ by minimizing photometric loss in Eq. (1).

3.2. System Setup

Training Pipeline As mentioned above, our system consists of two main components, a pose regression network

$\mathcal{G}_\Psi$ and an NVS-enabled direct matching module. With each component pre-trained on target data, we join them together to further refine the pose regression network: given an input image I, the pose regression network predicts a pose $\hat{v}$, from which an NVS system $\mathcal{H}$ renders a synthetic image $\hat{I}_{\hat{v}} = \mathcal{H}(\hat{v})$, enabling our direct matching supervision $\mathcal{L}_{photo}$. Meanwhile, the ground truth supervision $\mathcal{L}_{gt}$ is applied as well. As a result, the pose regression network $\mathcal{G}_\Psi$ is refined through a back-propagation of a weighted sum of $\mathcal{L}_{photo}$ and $\mathcal{L}_{gt}$ (Eq. (3)). Mathematically, our training pipeline can be framed as:

$$\Psi^* = \arg\min_{\Psi}(\lambda_1 \mathcal{L}_{photo} + \lambda_2 \mathcal{L}_{gt}), \tag{3}$$

where Ψ^* denotes optimized network parameters, and λ_1, λ_2 denote weights for each loss terms, respectively.

Pose Regression Network Our pose regression network $\mathcal{G}_\Psi$ follows the line of PoseNet [22] work, including a pre-trained feature extractor backbone and a fully connected layer that outputs a camera pose matrix $\hat{\mathbf{P}}$. Prior works [6, 20, 21, 22, 30, 43, 58, 59] typically use a quaternion or an axis-angle representation during pose estimation, requiring balancing between rotation and translation terms. Instead, our network regress a camera pose v using the representation of $\mathbf{P} = [\mathbf{R}|\mathbf{t}]$ to overcome this issue, where $\mathbf{R} \in \mathrm{SO}(3)$ is a rotation matrix denotes a camera orientation and $\mathbf{t} \in \mathbb{R}^3$ denotes camera position. For clarity, we refer v as a general pose concept and $\mathbf{P}$ as a specific pose representation.

Our ground truth supervision loss is defined as the L2 distance between a ground truth pose $\mathbf{P}$ and an estimated pose $\hat{\mathbf{P}}$:

$$\mathcal{L}_{gt} = \|\mathbf{P} - \hat{\mathbf{P}}\|_2, \tag{4}$$

removing the need for balancing rotation and translation terms while offering competitive performance.

NeRF Two challenges arise while adapting NeRF as our NVS system $\mathcal{H}$ in relocalization context. First, the training cost is expensive in relocalization tasks, where a training video could easily yield thousands of images even after frame subsampling. We resolve this issue by reducing the NeRF model size and removing the hierarchical training scheme.

Second, strong artifacts occur in synthetic images if the NeRF is applied in our task without modifications. Two reasons account for that: a) NeRF is not designed for outward-looking scenes [63] and b) photometric consistency is violated when auto-focus/exposure fluctuation and rolling shutter effect appear. We mitigate this issue by adapting a coarse-to-fine positional encoding scheme $\gamma_\alpha(\cdot)$ from Nerfies [39]. Specifically, an input signal p is encoded by $\gamma_\alpha(p) = [p, \ldots, w_k(\alpha_t)\sin(2^k\pi p), w_k(\alpha_t)\cos(2^k\pi p), \ldots]$, where $0 \le k \le m-1, m \in \mathbb{N}$ and $w_k(\alpha_t)$ activates each band over epoch t, controlled by $\alpha_t = mt/N$. N is a user-defined maximum epoch number in training, where k reaches the maximum frequency band $m-1$. We refer interested readers to the full mathematical expression for $w_k(\alpha_t)$ in our supplementary material.

With the progressive positional encoding function, we are able to adapt NeRF as our NVS system $\mathcal{H}$, reducing artifacts and preserving high frequency details. Please refer to Section 4.4 for more discussion on the effectiveness in this approach.

3.3. Unlabeled Training

Inspired by MapNet+ [6], we propose to improve pose estimation in a semi-supervised manner, with unlabeled sequences captured in the same training scene. Unlike [6], which enforces a relative geometric constraint between two nearby frames and requires an additional VO algorithm, we rely on our bootstrapped pipeline to further refine the pose regression network $\mathcal{G}_\Psi$. Given an input image without ground truth pose annotation but not too far from labeled training videos, the training of $\mathcal{G}_\Psi$ can be supervised by the photometric loss between the synthetic image rendered by the direct matching module using the predicted pose. This semi-supervised training scheme can be effectively set up by setting $\lambda_1 = 1.0$ and $\lambda_2 = 0.0$. We find our unlabeled training works well, evidenced by the performance in Table 2.

4. Experiments

In the following, we discuss the implementation details of our solution in Section 4.1. We perform a thorough evaluation of the proposed method in Section 4.2 on the 7-Scenes dataset. We further evaluate our method on the LLFF dataset to demonstrate that the proposed method benefits from both the traditional pose regression method and the direct matching method (Section 4.3). Finally, to gain more insights to our modification on positional encoding and the effectiveness of direct matching for camera localization, we apply more experiments in the ablation study (Section 4.4).

4.1. Implementation Details

Pose Regression We build our pose regression model upon prior DNN-based methods [6, 21, 22] using PyTorch [40]. We choose to use the MobileNetV2 [45] backbone in this work. We freeze the batch normalization layers [16] from the pre-trained ImageNet backbone to train our baseline pose regression model. Since a direct rotation matrix regression may not belong to SO(3), a singular value decomposition (SVD) is applied to normalize the rotation component of $\hat{\mathbf{P}}$ during inference time. However, we also

	without unlabeled data									with unlabeled data		
Scene	PN [22]	PN learned weights [21]	geo. PN [21]	LSTM PN [59]	Hourglass PN [30]	BranchNet [61]	DSO [10]	MapNet [6]	Direct-PN	MapNet+ [6]	MapNet+ PGO [6]	Direct-PN+U
Chess	0.32/8.12	0.14/4.50	0.13/4.48	0.24/5.77	0.15/6.17	0.18/5.17	0.17/8.13	0.08/3.25	0.10/3.52	0.10/3.17	0.09/3.24	0.09/2.77
Fire	0.47/14.4	0.27/11.8	0.27/11.3	0.34/11.9	0.27/10.8	0.34/8.99	0.19/65.0	0.27/11.7	0.27/8.66	0.20/9.04	0.20/9.29	0.16/4.87
Heads	0.29/12.0	0.18/12.1	0.17/13.0	0.21/13.7	0.19/11.6	0.20/14.2	0.61/68.2	0.18/13.3	0.17/13.1	0.13/11.1	0.12/8.45	0.10/6.64
Office	0.48/7.68	0.20/5.77	0.19/5.55	0.30/8.08	0.21/8.48	0.30/7.05	1.51/16.8	0.17/5.15	0.16/5.96	0.18/5.38	0.19/5.42	0.17/5.04
Pumpkin	0.47/8.42	0.25/4.82	0.26/4.75	0.33/7.00	0.25/7.0	0.27/5.10	0.61/15.8	0.22/4.02	0.19/3.85	0.19/3.92	0.19/3.96	0.19/3.59
Kitchen	0.59/8.64	0.24/5.52	0.23/5.35	0.37/8.83	0.27/10.2	0.33/7.40	0.23/10.9	0.23/4.93	0.22/5.13	0.20/5.01	0.20/4.94	0.19/4.79
Stairs	0.47/13.8	0.37/10.6	0.35/12.4	0.40/13.7	0.29/12.5	0.38/10.3	0.26/21.3	0.30/12.1	0.32/10.61	0.30/13.4	0.27/10.6	0.24/8.52
Average	0.44/10.44	0.24/7.87	0.23/8.12	0.31/9.85	0.23/9.53	0.29/8.30	0.26/29.4	0.21/7.77	0.20/7.26	0.19/7.29	0.18/6.55	0.16/5.17

Table 1: Pose regression results on 7 Scenes datasets. We compare our method with both direct matching and absolute pose regression methods, in median translation error (m) and rotation error (°). Bottom row is the average of median errors of all scenes. PN denotes PoseNet. Numbers in red represent the best performance with or without unlabeled data.

find that the pose regression network learns to predict orthogonal rotation matrices even without using the SVD. All models are optimized with the Adam optimizer [23]. The base model is trained with a batch size of 4 and a learning rate of 1×10^{-4}. We implement an early stopping strategy with a patience value of 200 and schedule the learning rate decay for every 50 epochs on validation loss plateau with a factor of 0.95.

NeRF Our NeRF model is trained with input poses in SE(3). To ensure a consistent coordinate system in pose regression and NVS, we further align and recenter the camera poses with zero-means similar in Mildenhall *et al*. [34]. The NeRF architecture mainly follows the original implementation [34], except we apply a coarse-to-fine positional encoding $\gamma_\alpha(p)$ for both positions and directions. We set the maximum frequency band $m = 8$, and time to reach the maximum frequency band $N = 1200$ epochs.

Training For our proposed methods in Section 3.2 and Section 3.3, we train the pose regression model with direct matching (Direct-PoseNet) with $\lambda_1 = 0.3$ and $\lambda_2 = 0.7$, and further fine-tune the model (Direct-PoseNet+U) with $\lambda_1 = 1.0$ and $\lambda_2 = 0.0$ to simulate the unlabeled data circumstances. We set the batch size to 1 for training both models. The learning rate is set to 1×10^{-5} with the same early stopping strategy as above. All models are trained within 24 hours with a single Nvidia 1080Ti graphic card. In our experience, the NeRF training time can reach approximately the same as training PoseNet models. More details on network architecture and training procedure are provided in the supplementary material.

4.2. Evaluation on 7-Scenes

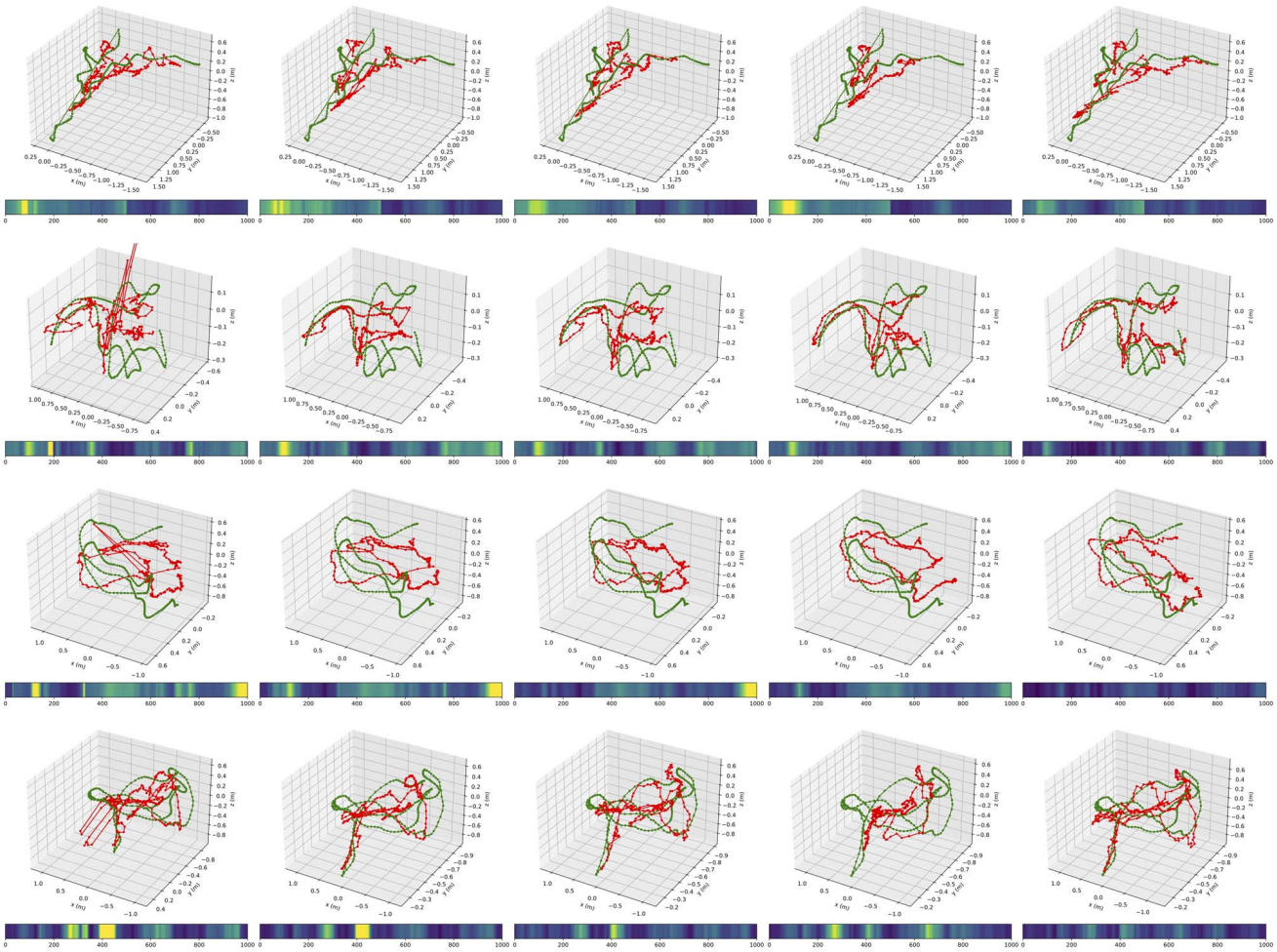
We evaluate our method on a well-known camera localization dataset 7-Scenes [15, 51]. It consists of seven indoor scenarios, each scale from $2m^3$ to $6m^3$. The sequences were shot by Kinect RGB-D camera at 640×480 resolution, and the ground truth poses were obtained by a dense 3D model.

The pose regression network takes an input image in 320×240 and the pre-trained NeRF model is trained with resized images in 160×120, but inference in 320×240 for Direct-PoseNet and Direct-PoseNet+U training. For each scene, we train our NeRF model with a learning rate of 5×10^{-4} for 4000 epochs with Adam optimizer and decays exponentially to 8×10^{-5} throughout the course of optimization. We set the near and far bounds $[b_n, b_f]$ to $[0.5, 4]$ except for the Heads scene, we set them to $[0.5, 2.5]$. However, unlike the original NeRF [34], which uses a coarse-to-fine sampling approach in its architecture, we only use a single MLP model with a width of 128. We sample one image with a batch of 1024 rays for each iteration, and each ray uniformly samples $N = 128$ bins. The above modifications achieve approximately $3\times$ speed up compare to the original NeRF paper. To further improve training efficiency, our NeRF model, Direct-PoseNet model, and Direct-PoseNet+U model only use a spacing window $d = 5$ of the training set for scenes contain ≤ 2000 frames, and $d = 10$ of the training set otherwise.

We summarize complete quantitative comparisons of our proposed method with prior absolute pose regression works and DSO in Table 1. For the experiment of Direct-PoseNet+U, we follow MapNet+ to use the unlabeled test sequences for fine-tuning. We do not use the entire test sequences for fine-tuning, but only 1/5 or 1/10 of the sequences described above to ensure our method is not overfitting to the entire test sequences. We also demonstrate a selection of the visual comparisons in Fig. 2.

4.3. Evaluation on LLFF

We further evaluate our method on another real-world complex scene dataset, the LLFF dataset [33]. The dataset consisting of 8 forward-facing scenes captured with a handheld cellphone and holds out $1/8$ of the data as the test set.



(a) PoseNet [6, 21, 22] (b) MapNet [6] (c) Direct-PoseNet (d) MapNet+PGO [6] (e) Direct-PoseNet+U

Figure 2: Visualization of camera relocalization results on 7-Scenes dataset [15, 51]. For each 3D plot, we show the ground truth camera trajectory in green and the predicted trajectory in red. The bottom color bar represents rotation errors for each subplot. Yellow represents high rotation error, and blue represents low rotation error for each test sequence. Sequence names from top to bottom are: Stairs-all, Heads-all, Fire-seq-03, Office-seq-09.

It is ideal for an experiment because a high-quality NeRF can be trained on LLFF to examine the combined effects of pose regression and direct matching supervision.

We compare our method with both the pose regression method and an inverting NVS method iNeRF by Lin *et al*. [62]. The iNeRF uses an iterative optimization approach on each test image to recover the camera pose by inverting a trained NeRF model. On the other hand, our method does not rely on iterative optimization and produces a more generalized and efficient model. To ensure a fair comparison, we trained our NeRF model to follow the same setting with Lin *et al*., which uses a standard NeRF model with a ray batch size of 2048. We fine-tune the NeRF model on four scenes (*Fern*, *Fortress*, *Horn*, *Room*) with the baseline pose regression model and compute the percentage of predicted pose whose error is less than 5cm and the percentage of predicted pose whose error is less than 5°. We report the experiment results in Table 2. We observe that our proposed pose regression method gains benefit both from the

error rate in %	iNeRF	PoseNet+SE(3)	Direct-PoseNet+U
<5cm, <5°	73%, 71%	57%, 100%	78%, 100%

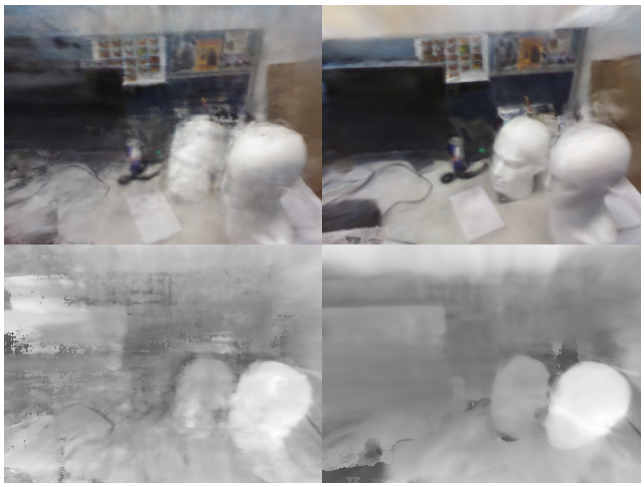
Table 2: We report the percentage of correctly re-localized frames below an error threshold of 5cm and percentage of re-localized frames below an error threshold 5° on the *Fern*, *Fortress*, *Horns*, *Room* scenes of LLFF dataset [33]

pose regression approach and the direct matching approach, resulting in the top performance.

4.4. Ablation Study

Effectiveness of Modified Positional Encoding In real-life datasets such as 7-Scenes, there are multiple sources to keep NeRF from rendering a high-quality, photorealistic view of the scene. Artifacts may be produced by letting the NeRF model learn from images with severe deformation (*e.g.*, from camera rolling shutter or deforming object) or motion blur from long exposure among frames. Moreover, training and testing in very different camera trajectories is a situation for NeRF likely to fail because it tries to generate the scene from unfamiliar volumetric rendering locations.

We build a toy example to demonstrate the phenomenon in the 7-Scenes dataset, and the effectiveness of our modification in NeRF's positional encoding. We randomly select a frame in Heads and sample a portion of training and validation data that lies within its frustum overlap threshold using an approach similar to Balntas *et al*. [1]. For this experiment, we set the frustum overlap threshold to be 0.85. We report the peak signal-to-noise ratio (PSNR) on this toy dataset for fixed full encoding ($m = 10$) in the original NeRF paper, fixed half encoding ($m = 5$), and the coarse-to-fine encoding schemes in Table 3. We show a qualitative comparison between the original NeRF embedding scheme and our modified coarse-to-fine scheme in Fig. 3. Overall, we find that using a coarse-to-fine positional encoding approach generally obtains a higher quality NeRF model throughout the 7-Scenes dataset in our experiments.



(a) Fixed P.E. [34] (b) Coarse-to-fine P.E.

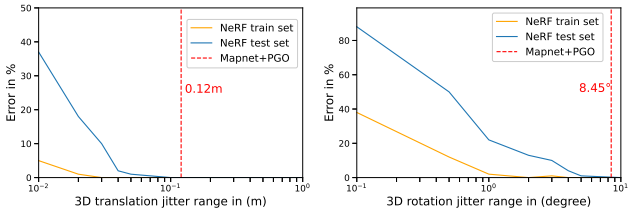
Figure 3: A visual comparison between (a) fixed positional encoding (P.E.) and (b) the coarse-to-fine P.E in Heads scene. Top: testset renderings from two NeRF models with different P.E. schemes. Bottom: disparity maps (inverse depth). Notice that NeRF with fixed P.E. produces stronger artifacts in this outward looking scene. Even though our encoding do not completely remove all artifacts, it recovers more structure and details than the original NeRF scheme. We provide more detailed discussion on why NeRF suffers from severe artifacts in Section 4.4.

Model	Full encoding	Half encoding	Coarse-to-fine
PSNR	16.64	17.16	**17.50**

Table 3: Comparison of different NeRF positional encoding scheme in our toy dataset (validation split).

Effectiveness of Direct Matching We investigate the effectiveness of direct matching for supervising the pose regression training. We first train a NeRF model using the Heads data. We randomly perturb the ground truth pose in different ranges and compute the photometric loss $\mathcal{L}_{ph\_perturb}$ with the ground truth image. We then count the percentage that $\mathcal{L}_{ph\_perturb} < \mathcal{L}_{ph\_GT}$, where $\mathcal{L}_{ph\_GT}$ denotes the photometric loss using the ground truth pose. We define such a percentage as the *error rate*. Ideally, views generated from perturbed poses should have higher photometric loss than the loss we get from using the ground truth poses. Thus the *error rate* indicates the chances that non-ideal cases would happen, which potentially damages the optimization of our pose regression.

Intuitively, the greater range we perturb the ground truth pose with, the more displacement in the appearance of rendered images will have and shall lead to a lower *error rate*. In this experiment, we jitter poses in the range between $\pm[0.01m, 1m]$ for 3D translation and the range between



(a) 3D translation jitter only (b) 3D rotation jitter only

Figure 4: We feed randomly perturbed poses to the NeRF model trained in Heads, and validate the robustness of our photometric loss $\mathcal{L}_{photo}$. By cumulatively counting the chances of $\mathcal{L}_{ph\_perturb} < \mathcal{L}_{ph\_GT}$, the *error rate* remains nearly 0 when the perturbation is smaller to the Mapnet+PGO reference threshold, for both translation and rotation. This indicates that the photometric loss from direct matching is effective to supervise the training of a pose regression network.

Scene	Geo. PoseNet [21]	PoseNet (ResNet34) [6]	PoseNet+logq (ResNet34) [6]	PoseNet+SE(3) (ResNet34)	PoseNet+SE(3) (MobileNetV2)
Backbone Error (Top-1/Top-5)	31.3%/11.1%	**26.7%/8.58%**	**26.7%/8.58%**	**26.7%/8.58%**	28.12%/9.71%
Average	0.23m, 8.12°	0.23m, 8.49°	0.22m, 8.07°	**0.21m**, 8.71°	**0.21m**, **7.84**°

Table 4: A comparison between our SE(3) PoseNet baseline and other quaternion-based baselines. Our direct SE(3) supervision offers competitive results with both backbones while removing the need for balancing rotation and translation terms.

$\pm[0.1°, 10°]$ for 3D rotation movement. For each of the selected scales, we randomly jitter 500 poses to estimate the expected *error rate*. As the results are shown in Fig. 4, both 3D translation and 3D rotation *error rates* drop close to 0 below the reference threshold of MapNet+PGO, which may explain why training with direct matching can obtain better performance overall.

Effectiveness of Regressing SO(3) We also compares our baseline pose regression model with prior baselines from PoseNet and MapNet in Table 4. We achieve on-par results by directly replacing the rotation representation from quaternion to SO(3) rotation representation. Our MobileNetV2 performs overall the best in terms of average results. We use identical training hyperparameters to train our baseline model for each scene, and our MobileNetV2 feature extractor is not the best regarding the ImageNet benchmark compared to prior baselines. Our baseline models' superior performance indicates that our rotation representation is just as effective as quaternion representation. A full table on scene specific performance is provided in the supplementary material.

Summary of Ablation We justify our design decisions to show how each component variation contributes to the relocalization performance in Table 5. First, replacing the SE(3) representation with separate quaternion rotation and translation position terms leads to lower accuracy due to the balancing requirement of the two terms during training. The most significant performance drop is when removing the coarse-to-fine training strategy on NeRF. It indicates that the NVS reconstruction quality does affect the overall relocalization accuracy. In addition, we observe that using full NeRF architecture to train our model can obtain slight accuracy improvement. However, this is at the cost of a much longer training time (i.e. 22hrs vs. 78hrs on Kitchen). We observe that the same phenomena hold valid when training with unlabeled data as well.

Method	7 Scenes
Direct-PN	0.20m, 7.26°
- SE(3)	0.21m, 7.58°
- Coarse-to-fine	0.22m, 7.91°
- Direct Matching	0.22m, 8.07°
Direct-PN + Full NeRF	**0.20m, 7.16**°

Table 5: A performance breakdown for each component in our method. The performance drops for when our modifications on SE(3), coarse-to-fine encoding, and the direct-matching module are removed. The performance improves slightly if a full-size NeRF [34] (with the hierarchical architecture, and a MLP with deeper and wider layers), but at the cost of a much longer training time.

5. Conclusion and Discussion

In this work, we show that one can use a differentiable renderer to improve pose regression performance. We present a relocalization pipeline that outperforms previous single-image APR methods on the 7-Scenes benchmark and achieves state-of-the-art performance on the LLFF dataset, with two main contributions. First, we joint a direct matching module with a pose regression network, offering superior performance while maintaining low inference cost. Second, we further boost our method's performance by applying a simple but effective semi-supervised training scheme to unlabeled data. To adapt with outward-looking relocalization datasets, which violates assumptions in NeRF, we employ a coarse-to-fine positional encoding strategy to improve rendering qualities.

One of the limitations of this work is that the effectiveness of our direct matching highly depends on the robustness of the NVS methods, which might fail for various reasons. For example, NeRF does not perform well in large-scale scenes, dynamic environments, or outdoor scenarios where auto-exposure fluctuates. The next challenge for us is to circumvent the assumptions made in NeRF so that our method is extensible to more challenging scenarios.

Acknowledgements We thank Kejie Li for his advice on experimental design and generous help to polish our paper. We also appreciate Henry Howard-Jenkins and Theo W. Costain for some great comments and discussions.

References

[1] V. Balntas, S. Li, and V. Prisacariu. Relocnet: Continuous metric learning relocalisation using neural nets. In *ECCV*, 2018. 1, 7

[2] Eric Brachmann, Alexander Krull, Sebastian Nowozin, Jamie Shotton, Frank Michel, Stefan Gumhold, and Carsten Rother. DSAC - Differentiable RANSAC for Camera Localization. In *CVPR*, 2017. 1

[3] Eric Brachmann and Carsten Rother. Learning Less is More - 6D Camera Localization via 3D Surface Regression. In *CVPR*, 2018. 1

[4] Eric Brachmann and Carsten Rother. Neural- Guided RANSAC: Learning where to sample model hypotheses. In *ICCV*, 2019. 1

[5] Eric Brachmann and Carsten Rother. Visual camera relocalization from RGB and RGB-D images using DSAC. *arXiv*, 2020. 1

[6] S. Brahmbhatt, J. Gu, K. Kim, J. Hays, and J. Kautz. Geometry-Aware Learning of Maps for Camera Localization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1, 2, 4, 5, 6, 8

[7] S.E. Chen and L. Williams. View interpolation for image synthesis. In *Computer Graphics Proceedings, Annual Conference Series*, 1993. 2

[8] Ondrej Chum and Jiri Matas. Optimal Randomized RANSAC. In *PAMI*, 2008. 1

[9] A. Davison, I. Reid, N. Molton, and O. Stasse. MonoSLAM: Realtime single camera SLAM. In *IEEE Trans. Pattern Anal. Mach. Intell.*, 2007. 2

[10] J. Engel, V. Koltun, and D. Cremers. DSO: Direct sparse odometry. In *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 2017. 1, 2, 3, 5

[11] J. Engel, T. Schops, and D. Cremers. Semi-dense visual odometry for a monocular camera. In *In Proceedings of IEEE International Conference on Computer Vision (ICCV)*, 2013. 1, 2, 3

[12] Martin A. Fischler and Robert C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. In *CACM*, 1981. 1

[13] C. Forster, M. Pizzoli, and D. Scaramuzza. SVO: Fast semi-direct monocular visual odometry. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2014. 2, 3

[14] C. Forster, Z. Zhang, M. Gassner, M. Werlberger, and D. Scaramuzza. SVO: Semi-direct visual odometry for monocular and multi-camera systems. In *IEEE Transactions on Robotics*, 2017. 2, 3

[15] Ben Glocker, Shahram Izadi, Jamie Shotton, and Antonio Criminisi. Real-time rgb-d camera relocalization. In *International Symposium on Mixed and Augmented Reality (ISMAR)*, 2013. 5, 6

[16] Ben Glocker, Shahram Izadi, Jamie Shotton, and Antonio Criminisi. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *ICCV*, 2015. 4

[17] Steven J Gortler, Radek Grzeszczuk, Richard Szeliski, and Michael F Cohen. The lumigraph. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, 1996. 2

[18] M. Irani and P. Anandan. All about direct methods. In *Workshop Vis. Algorithms: Theory Pract.*, 1999. 2

[19] H. Jin, P. Favaro, and S. Soatto. Real-time 3-d motion and structure of point features: Front-end system for vision-based control and interaction. In *IEEE Conf. Comput. Vis. Pattern Recognit.*, 2000. 2

[20] A. Kendall and R. Cipolla. Modelling uncertainty in deep learning for camera relocalization. In *ICRA*, 2016. 1, 4

[21] A. Kendall and R. Cipolla. Geometric loss functions for camera pose regression with deep learning. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1, 2, 4, 5, 6, 8

[22] A. Kendall, M. Grimes, and R. Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *International Conference on Computer Vision*, 2015. 1, 2, 4, 5, 6

[23] D. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 5

[24] G. Klein and D. Murray. Parallel tracking and mapping for small ar workspaces. In *IEEE ACM Int. Symp. Mixed Augmented Reality*, 2007. 2

[25] G. Klein and D. Murray. Parallel tracking and mapping for small ar workspaces. In *IEEE Trans. Robot.*, 2015. 2

[26] Marc Levoy and Pat Hanrahan. Light field rendering. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, 1996. 2

[27] Lingjie Liu, Jiatao Gu, Kyaw Zaw Lin, Tat-Seng Chua, and Christian Theobalt. Neural sparse voxel fields. *NeurIPS*, 2020. 3

[28] Stephen Lombardi, Tomas Simon, Jason Saragih, Gabriel Schwartz, Andreas Lehrmann, and Yaser Sheikh. Neural volumes: Learning dynamic renderable volumes from images. In *ACM Trans. Graph.*, 2019. 3

[29] Ricardo Martin-Brualla, Noha Radwan, Mehdi S. M. Sajjadi, Jonathan T. Barron, Alexey Dosovitskiy, and Daniel Duckworth. NeRF in the Wild: Neural Radiance Fields for Unconstrained Photo Collections. In *CVPR*, 2021. 3

[30] I. Melekhov, J. Ylioinas, J. Kannala, and E. Rahtu. Image-based localization using hourglass networks. In *ICCV Workshops*, 2017. 1, 2, 4, 5

[31] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 3

[32] Moustafa Meshry, Dan B Goldman, Sameh Khamis, Hugues Hoppe, Rohit Pandey, Noah Snavely, and Ricardo Martin-Brualla. Neural rerendering in the wild. In *In IEEE Conf. Comput. Vis. Pattern Recog.*, 2019. 3

[33] Ben Mildenhall, Pratul P Srinivasan, Rodrigo Ortiz-Cayon, Nima Khademi Kalantari, Ravi Ramamoorthi, Ren Ng, and Abhishek Kar. Local light field fusion: Practical view synthesis with prescriptive sampling guidelines. In *ACM TOG*, 2019. 5, 6

[34] 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 *ECCV*, 2020. 3, 5, 7, 8

[35] T. Naseer and W. Burgard. Deep regression for monocular camera-based 6-dof global localization in outdoor environments. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017. 1

[36] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison. Dtam: Dense tracking and mapping in real-time. In *Int. Conf. on Computer Vision (ICCV)*, 2011. 2, 3

[37] Michael Niemeyer, Lars Mescheder, Michael Oechsle, and Andreas Geiger. Differentiable volumetric rendering: Learning implicit 3d representations without 3d supervision. In *CVPR*, 2020. 3

[38] Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove. Deepsdf: Learning continuous signed distance functions for shape representation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 3

[39] Keunhong Park, Utkarsh Sinha, Jonathan T. Barron, Sofien Bouaziz, Dan B Goldman, Steven M. Seitz, and Ricardo Martin-Brualla. Deformable Neural Radiance Fields. *arXiv preprint arXiv:2011.12948*, 2020. 3, 4

[40] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems*, pages 8024–8035, 2019. 4

[41] Dai Peng, Zhang Yinda, Li Zhuwen, Liu Shuaicheng, and Zeng Bing. Neural point cloud rendering via multi-plane projection. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020. 3

[42] M. Pizzoli, C. Forster, and D. Scaramuzza. Remode: Probabilistic, monocular dense reconstruction in real time. In *IEEE Int. Conf. on Robotics and Automation (ICRA)*, 2014. 2, 3

[43] N. Radwan, A. Valada, and W. Burgard. Vlocnet++: Deep multitask learning for semantic visual localization and odometry. In *IEEE Robotics and Automation Letters*, 2018. 1, 4

[44] Rahul Raguram, Ondrej Chum, Marc Pollefeys, Jiri Matas, and Jan-Michael Frahm. USAC: A Universal Framework for Random Sample Consensus. In *PAMI*, 2013. 1

[45] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and LC. Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *CVPR*, 2018. 4

[46] T. Sattler, B. Leibe, and L. Kobbelt. Fast image-based localization using direct 2d-to-3d matching. In *ICCV*, 2011. 1

[47] T. Sattler, B. Leibe, and L. Kobbelt. Improving image-based localization by active correspondence search. In *European conference on computer vision*, 2012. 1

[48] T. Sattler, B. Leibe, and L. Kobbelt. Efficient & Effective Prioritized Matching for Large-Scale Image-Based Localization. In *PAMI*, 2017. 1

[49] T. Sattler, Q. Zhou, M. Pollefeys, and L. Leal-Taixe. Understanding the limitations of cnn-based absolute camera pose regression. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1, 2

[50] Katja Schwarz, Yiyi Liao, Michael Niemeyer, and Andreas Geiger. Graf: Generative radiance fields for 3d-aware image synthesis. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2020. 3

[51] Jamie Shotton, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew Fitzgibbon. Scene coordinate regression forests for camera relocalization in rgb-d images. In *CVPR*, 2013. 1, 5, 6

[52] Y. Shvit and R. Ferens. Introduction to camera pose estimation with deep learning. *arXiv preprint arXiv: 1907.05272*, 2019. 1

[53] Vincent Sitzmann, Justus Thies, Felix Heide, Matthias Nießner, Gordon Wetzstein, and Michael Zollhöfer. DeepVoxels: Learning persistent 3d feature embeddings. In *Proc. Computer Vision and Pattern Recognition (CVPR)*, 2019. 3

[54] Vincent Sitzmann, Michael Zollhöfer, and Gordon Wetzstein. Scene representation networks: Continuous 3D-structureaware neural scene representations. In *NeurIPS*, 2019. 3

[55] Hajime Taira, Masatoshi Okutomi, Torsten Sattler, Mircea Cimpoi, Marc Pollefeys, Josef Sivic, Tomas Pajdla, , and Akihiko Torii. InLoc: Indoor Visual Localization with Dense Matching and View Synthesis. *CVPR*, 2018. 1

[56] Ayush Tewari, Christian Theobalt, Dan B Goldman, Eli Shechtman, Gordon Wetzstein, Jason Saragih, Jun-Yan Zhu, Justus Thies, Kalyan Sunkavalli, Maneesh Agrawala, Matthias Nießner, Michael Zollhöfer, Ohad Fried, Ricardo Martin Brualla, Rohit Kumar Pandey, Sean Fanello, Stephen Lombardi, Tomas Simon, and Vincent Sitzmann. State of the art on neural rendering. In *Computer Graphics Forum*, 2020. 2

[57] Justus Thies, Michael Zollhöfer, and Matthias Nießner. Deferred neural rendering: Image synthesis using neural textures. In *ACM Transactions on Graphics*, 2019. 3

[58] A. Valada, N. Radwan, and W. Burgard. Deep auxiliary learning for visual localization and odometry. In *ICRA*, 2018. 1, 4

[59] F. Walch, C. Hazirbas, L. Leal-Taixe, T. Sattler, S. Hilsenbeck, and D. Cremers. Image-based localization using lstms for structured feature correlation. In *International Conference on Computer Vision*, 2017. 1, 2, 4, 5

[60] Zirui Wang, Shangzhe Wu, Weidi Xie, Min Chen, and Victor Adrian Prisacariu. NeRF−−: Neural radiance fields without known camera parameters. *arXiv preprint arXiv:2102.07064*, 2021. 3

[61] J. Wu, L. Ma, and X. Hu. Delving Deeper into Convolutional Neural Networks for Camera Relocalization. In *ICRA*, 2017. 1, 2, 5

[62] Lin Yen-Chen, Pete Florence, Jonathan T. Barron, Alberto Rodriguez, Phillip Isola, and Tsung-Yi Lin. inerf: Inverting neural radiance fields for pose estimation. In *arxiv arXiv:2012.05877*, 2020. 3, 6

[63] Kai Zhang, Gernot Riegler, Noah Snavely, and Vladlen Koltun. Nerf++: Analyzing and improving neural radiance fields. *arXiv:2010.07492*, 2020. 4