CamNet: Coarse-to-Fine Retrieval for Camera Re-Localization

Mingyu Ding[1] Zhe Wang[2] Jiankai Sun[3] Jianping Shi[2] Ping Luo[1]
[1]The University of Hong Kong [2]SenseTime Research [3]The Chinese University of Hong Kong
{dingmyu,pluo.lhi}@gmail.com sj019@ie.cuhk.edu.hk
{wangzhe,shijianping}@sensetime.com

Abstract

Camera re-localization is an important but challenging task in applications like robotics and autonomous driving. Recently, retrieval-based methods have been considered as a promising direction as they can be easily generalized to novel scenes. Despite significant progress has been made, we observe that the performance bottleneck of previous methods actually lies in the retrieval module. These methods use the same features for both retrieval and relative pose regression tasks which have potential conflicts in learning. To this end, here we present a coarse-to-fine retrieval-based deep learning framework, which includes three steps, i.e., image-based coarse retrieval, pose-based fine retrieval and precise relative pose regression. With our carefully designed retrieval module, the relative pose regression task can be surprisingly simpler. We design novel retrieval losses with batch hard sampling criterion and two-stage retrieval to locate samples that adapt to the relative pose regression task. Extensive experiments show that our model (CamNet) outperforms the state-of-the-art methods by a large margin on both indoor and outdoor datasets.

1. Introduction

The task of camera re-localization has long been studied in various visual SLAM systems [25] or structure from motion (SfM) systems. These approaches are built on the elegant multi-view geometry theory by optimizing the geometric constraints from low-level key points. Hand-craft image descriptors (*e.g.*, SIFT, ORB [28, 4]) are widely used to find the correspondences between the local features extracted from images. The 6-DoF camera pose is then recovered from such correspondences. However, these methods have difficulty in dealing with texture-less scenes, or places with drastic changes in illumination, occlusions and repetitive structures. Also, the computational expense is high. Recently, a variety of machine learning algorithms have been applied to this problem. Some methods based on Random Forest [9, 23, 35, 40] establish 2D-3D matches to recover 6-DoF camera pose by applying RANSAC to achieve impressing results. Combining the differentiable counterpart of RANSAC and fully convolutional neural network (FCN), [5, 6] are proposed for more robust 2D-3D matches. While this approach achieves state-of-the-art results, it requires additional depth maps associated with input images at training time.

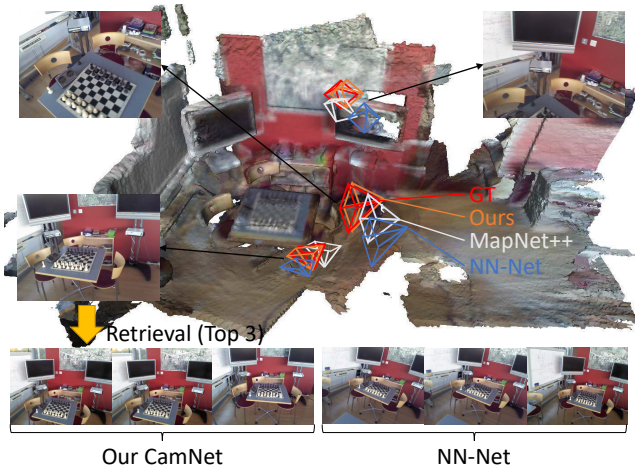


Figure 1: Our CamNet outperforms the state-of-the-art image-based method MapNet++ [7] by a large margin. As a retrieval-based model, the CamNet shows more accurate retrieval results than NN-Net [18]. The top 3 retrieval results are displayed.

Inspired by the success of deep learning models, a series of approaches [17, 41, 15, 16, 7, 38, 39] have been proposed to regress the absolute camera pose directly from RGB images. In essence, these models "remember" the scene by leveraging the expressive power of deep models. However, it means that they have to be retrained for unseen scenes, which largely limits their generalization capabilities. The other category of deep learning methods [18, 3] builds a database with extracted features of images from the target scene as well as the ground truth absolute poses. Then given a query image, it first retrieve the most similar image in the database with its absolute poses and then predict the relative

Table 1: A comparison of the four categories of methods. 'RGB' (image),'3D' (3D model/Depth) and 'Seg' (Segmentation) mean the data that the model needs. 'Generalized' means the generalization capabilities of the model, *i.e.*, used in new scenarios without the model retraining.

Category	Method	RGB	3D	Seg	Generalized	Accuracy
2D-3D matching	DSAC [5], Active Search [30], SCoRe Forest [35]	✓	✓			High
Semantic-based	SVL[34],VlocNet++ [26]	✓		✓		High
Absolute pose	PoseNet[17, 16], MapNet[7],VLocNet[38]	✓				Medium
Relative pose	NN-Net[18],RelocNet [3]	✓			Yes	Low
Relative pose	Ours	✓			Yes	High

pose based on the query image and the retrieved one. However, in practice, it is difficult to directly retrieve the optimal database entry, which then impairs the relative pose regression accuracy. Our method belongs to this category, while we tackle this problem by a novel coarse-to-fine strategy to gradually get close to the best database entry.

In this paper, we propose a coarse-to-fine retrieval-based framework for camera localization. Our framework has the merits of good generalization ability as retrieval-based models, as well as good performance as 2d-3d matching models. Previous retrieval-based methods [18, 3] use shared features for image retrieval and pose regression. However, we argue that it is unreasonable because image retrieval model should be focusing on learning scene similarities and ignore subtle camera view angle changes, while pose regression model needs to identify the very differences of view angle changes between the paired images. Therefore, these two tasks have potential conflicts in learning and may have a bad effect on each other if all the features are shared. To deal with this problem, we design a siamese architecture with three branches for image-based coarse retrieval, pose-based fine retrieval and relative pose regression, respectively. Only the encoder of the network is shared while the three tasks all have their own branches, and all the three tasks can be jointly learned in an end-to-end manner.

Our contributions are three-fold: (1) We propose a coarse-to-fine framework CamNet for camera re-localization which is highly scalable and accurate. (2) Taking advantage of relative pose regression, we propose novel retrieval losses and two-stage retrieval to further improve the accuracy of camera re-localization. (3) We demonstrate the generalization ability of our model. Extensive experiments show our CamNet yields the state-of-the-art results on three benchmark data sets.

2. Related Work

Previous works on camera re-localization can be mainly divided into three categories, including 2D-3D matching localization, metric localization, and image retrieval localization. There are also several researches working on semantic camera pose refinement. A brief comparison of these categories is shown in Figure 1.

2.1. 2D-3D Matching Localization

Based on 3D scene models that are typically reconstructed by using SfM, 2D-3D correspondences are established by using descriptor matching. The *structure-based* methods employed one or more feature descriptors such as SIFT [21] or LIFT [43]. Instead of using hand-crafted features, [44, 27] learned to find better feature correspondences. The 2D-3D correspondences are used to estimate the camera pose of the query image by applying an *n*-point-pose solver such as [31, 32] within a RANSAC loop [11]. Moreover, Schmidt *et al.* [33] advocated a new approach to learn visual descriptors by harnessing a 3D generative model to automatically label correspondences.

Rather than directly learning the matching function to obtain 2D-3D correspondences via explicit feature matching, some previous works implicitly represented the 3D scene structure by predicting 3D scene coordinate using either CNNs [19, 5, 8, 6, 9, 20] or random forests [23, 35, 40, 24, 9]. Taira *et al.* [36] predicted the 6DoF pose of a query image with respect to a large indoor 3D map and pose verification. However, these methods may fail when handling large-scale outdoor scenes [34].

However, the above approaches relied on 3D models, which are expensive and time-consuming to construct and collect. Although DSAC++ [6] demonstrated that scene coordinate regression can be learned without ground truth 3D model by a 3-step training, the necessity to carefully initialize the depth of the scene is non-scalable and degenerates the localization accuracy. In addition, Sattler *et al.* [32] experimentally demonstrated that large-scale 3D models are not strictly necessary for accurate visual camera localization. Different from the above work, the proposed method is highly scalable to handle large-scale outdoor scenes without the need for a 3D model.

2.2. Metric Localization

The metric localization methods aim to regresses the metric position and orientation of the camera. For example, PoseNet [17] trained a CNN to regress the camera pose. PoseNet has been extended in many ways such as using LSTM to extract temporal information [41], localizing over video sequences by exploiting the constraint of tem-

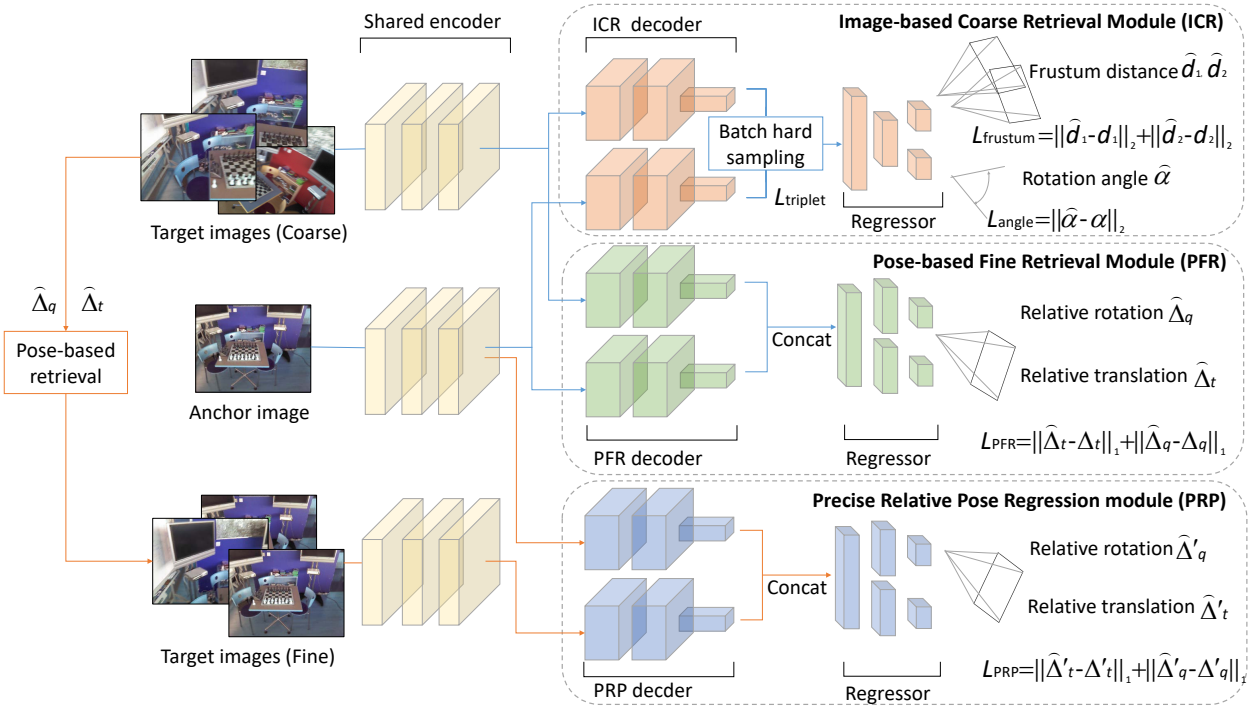


Figure 2: The overall pipeline of our framework. The blue line shows the training flow of our retrieval module, and the orange line shows the training flow of our precise relative pose regression. Best viewed in color.

poral smoothness [10], leveraging the weighted geometric loss function [16] and using a Bayesian convolutional neural network to estimate the re-localization uncertainty [15].

Furthermore, MapNet [7] brings geometric constraints and other cues such as visual odometry (vo), GPS, and IMU into learning to learn from both labeled data and unlabeled data. VLocNet [38] presented a multi-task model trained with the auxiliary Geometric Consistency Loss function to leverage relative pose information. VLocNet++ [39] simultaneously embedded geometric and semantic knowledge of the world into the pose regression network by employing a multitask learning approach.

The above approaches is effective for featureless indoor environments, where the SIFT-based structure from motion (SfM) may fail. However, these methods require learning for specific scenarios and lack of generalization capabilities, while our method is scalable and flexible to be extend to new scenarios without retraining the model.

2.3. Image Retrieval Localization

Visual camera localization is often treated as pose approximation of a query image , where the pose is estimated by the most similar images retrieved from the database. Germain *et al.* [12] introduced condition-specific subnetworks, which enables the computation of global image descriptors. Based on the Bag of Words (BoW) paradigm and the storage of image feature, Disloc [29, 2] leads to large memory consumptions for large-scale scenes. Different from the DenseVLAD [37] which relies on the handcrafted RootSIFT descriptors, the NetVLAD [1] is presented to learn the descriptors by CNN. However, these methods only predict an approximate location of the query, not an exact 6DoF pose.

Moreover, NN-Net [18] learned relative poses from pairs of RGB images to improve generalization capability without training of scene-specific deep networks. Balntas *et al.* [3] proposed to learn suitable convolutional representations for camera pose retrieval based on nearest neighbour matching and continuous metric learning-based feature descriptors. However, due to the difficulty in both retrieval and relative pose regression, the direct combination of these two components severely degrades their localization accuracy. Here, our method attempt to refine each component in a coarse-to-fine scheme and improve the final results with a considerable margin.

3. Methodology

Overview. We propose a retrieval-based coarse-to-fine framework CamNet for camera pose re-localization. The proposed model is based on a Siamese architecture, where the input to our model is a set of paired images with camera frustum overlapped. Our model consists of three modules including an image-based coarse retrieval module (ICR), a pose-based fine retrieval module (PFR), and a precise rela-

tive pose regression module (PRP). All these three modules share the same encoder network.

We design a two-stage retrieval in our framework. In general, given a query image, we first perform a nearest neighbour search by the ICR module from a database to find an image, which is most similar with the query image. Then, we perform coarse relative pose regression by our PFR module and obtain an coarse camera pose estimation. With this coarse camera pose estimation, we retrieve the closest pose and perform relative pose regression by our PRP module to obtain an accurate pose estimation.

The pipeline of our approach is illustrated in Figure 2. We construct a set of image pairs to train the Siamese network. Each image has its ground truth camera pose, which is a 4×4 matrix $\boldsymbol{M}$ in homogeneous coordinates. Specifically, $\boldsymbol{M} = \begin{pmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0} & 1 \end{pmatrix}$ where $\boldsymbol{R}$ is a 3-by-3 rotation matrix and $\boldsymbol{t}$ is a 3-by-1 translation vector. The loss function through our entire framework is as follows:

$$L = L_{\text{frustum}} + L_{\text{angle}} + L_{\text{triplet}} + L_{\text{PFR}} + L_{\text{PRP}}, \quad (1)$$

In the following, we will introduce each module and loss function of our model in detail.

3.1. Image-based Coarse Retrieval Module (ICR)

As shown at the top of Fig 2, we learn a ICR module before applying the PFR module. The image retrieval process is to find the image, which is most likely to produce an accurate relative pose estimation from an image database. The key question is what kind of data is essential to pose regression. The intuitive idea is that the larger the overlapping area between images, the better the pose regression. Thus, the overlap of camera frustum is suitable for learning pose-specific feature descriptors as [3] did.

However, different from [3] that only considered the one-way frustum overlap, we consider the bilateral camera frustum overlap in our model, which can better handle large translation. Nevertheless, we find that a pair of images with large camera frustum overlap is sometimes taken from the opposite direction. This leads to a huge relative rotation, which is difficult to predict by using a deep model. To solve this problem, we introduce a batch hard-sample mining strategy by combining a *hard triplet loss* and an *angle-based auxiliary loss*.

3.1.1 Bilateral Frustum Loss

We use the bilateral camera frustum distance d to devise our coarse retrieval loss function. In training, for RGB-D based indoor datasets, we utilize the depth and camera intrinsics $\boldsymbol{K}$ to calculate camera frustum overlap. For outdoor datasets, we propose an alternative that does not require depth information. Note that our model predicts camera pose by using only a RGB image in the inference phase, without using depth information and 3D models.

Given a pair of images with its ground truth poses denoted by $\boldsymbol{R}_1, \boldsymbol{t}_1, \boldsymbol{R}_2, \boldsymbol{t}_2$ and depth maps denoted by $\boldsymbol{D}_1, \boldsymbol{D}_2$, we can project pixels of the first image to the world coordinate system, and then project back to the pixel coordinate system of the second image. Formally, given the principal point coordinate (p_x, p_y) and the focal length f of the camera, we have

$$\boldsymbol{K} = \begin{pmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{pmatrix} \text{ and } \boldsymbol{X}_1 = \begin{pmatrix} x \\ y \\ 1 \end{pmatrix},$$

where $\boldsymbol{X}_1$ is the pixel coordinate of the first image. Then the corresponding coordinate $\boldsymbol{X}_2'$ of the second image can be obtained by the following

$$\boldsymbol{X}_2' = \boldsymbol{K}(\boldsymbol{R}_2{}^\top \boldsymbol{R}_1 \boldsymbol{K}^{-1} \boldsymbol{X}_1 \boldsymbol{D}_1(p_y, p_x) + \boldsymbol{t}_1 - \boldsymbol{t}_2), \quad (2)$$

where $\boldsymbol{D}_1(p_y, p_x)$ is the depth value of $\boldsymbol{X}_1$. We sample a uniform grid of pixels with size 10×10 from the first image, and treat the ratio of the projected coordinates inside area of the second image as the camera frustum overlap θ_1. Similarly, the reverse camera frustum overlap is denoted as θ_2.

For outdoor datasets without depth information, we use the ORB (ORiented Brief) similarity instead of the camera frustum overlap. Specifically, we extract the ORB key points S_1, S_2 of the two images, and perform a brute-force matching to get a set of matching pairs, denoted as $\text{BFMatcher}(S_1, S_2)$, which is explained in Appendix. Then, the ORB similarity denoted by θ_1' is defined as

$$\theta_1' = \frac{\text{num}\big(\text{BFMatcher}(S_1, S_2) > \xi\big)}{\text{num}\big(\text{BFMatcher}(S_1, S_2)\big)}, \quad (3)$$

where $\text{len}(\cdot)$ represents the number of corresponding key points, ξ is a threshold of the feature distance used to control the matching, and the reverse ORB similarity is denoted as θ_2'. We set ξ to 50 in all experiments for outdoor datasets.

By combining the above definitions, our camera frustum loss is defined by

$$L_{\text{frustum}} = \|\hat{d}_1 - d_1\|_2 + \|\hat{d}_2 - d_2\|_2. \quad (4)$$

For example, $d_1 = 1 - \theta_1$ denotes the camera frustum distance and $\hat{d}_1$ denotes the prediction of the frustum distance. Similarly, we predict the ORB similarity $d_1 = 1 - \theta_1'$ for outdoor datasets.

3.1.2 Angle-based Loss

Since the camera frustum overlap is insensitive to orientation between the pairs of images, we introduce another loss

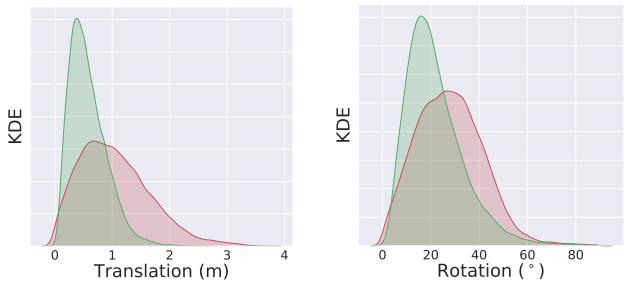


Figure 3: The training data distribution on 7-Scenes dataset of our two modules. The PFR module and the PRP module are displayed in red and green.

function, the angle-based auxiliary loss, for more accurate image retrieval. The camera rotation angle denoted as α corresponding to two rotation matrices can be defined by

$$\alpha = \arccos(\frac{\text{trace}(\boldsymbol{R}_1^\top \boldsymbol{R}_2 - 1)}{2})/\pi. \tag{5}$$

Then we define our angle-based auxiliary loss by

$$L_{\text{angle}} = \|\hat{\alpha} - \alpha\|_2, \tag{6}$$

where $\hat{\alpha}$ is the prediction of the camera rotation angle.

3.1.3 Hard Triplet Loss

By learning the rotation angle, the weakness of the camera frustum overlap can be alleviated. To further improve the accuracy of image retrieval, we introduce a batch hard sampling with a *hard-triplet loss* by following [42].

In particular, for each anchor image in the training set, we randomly combine it with some other images to form image pairs. Then, we divide all image pairs into three categories including easy, moderate, and hard, according to their camera frustum overlap as well as their relative rotations and translations. For each anchor image of a training mini-batch with batch size N, we randomly sample one image from each of the above three categories to construct a hard sampling pool.

Our goal is to make an image pair from the easy category has smaller distance than that from the moderate category in the feature space. The same goal applies between the moderate and the hard category. Formally, let $f_a^i, f_e^i, f_m^i, f_h^i$ denote the feature of the i_{th} anchor image, easy image, moderate image, and hard image respectively, our hard-triplet loss is defined by

$$\begin{aligned} L_{\text{triplet}} = &\big(\max_i(\|f_a^i, f_e^i\|_2) - \min_j(\|f_a^j, f_m^j\|_2) + \beta\big)_+ \\ &+ \big(\max_k(\|f_a^k, f_m^k\|_2) - \min_l(\|f_a^l, f_h^l\|_2) + \beta\big)_+, \end{aligned} \tag{7}$$

where $(z)_+ = \max(z, 0)$ is a maximum function and β is the value of the margin set to be 0.1, allowing the network to distinguish the positive samples from the negative ones. The maximum function of each line shortens the distances of positive pairs, while the minimum function enlarges the distances of negative pairs.

Algorithm 1 The pose refinement algorithm

Input: Anchor image $\mathbf{I}_a$
Target images $\mathbf{I}_e, \mathbf{I}_m, \mathbf{I}_h$
Parameter $\lambda = 0.5, \alpha = 0.04, \epsilon = 0.02$
for each batch **do**
1. Obtain the relative pose $\boldsymbol{\Delta}_M^e, \boldsymbol{\Delta}_M^m, \boldsymbol{\Delta}_M^h$ by the PFR module;
2. Calculate estimated camera pose of the anchor image as $\hat{\boldsymbol{M}}_e, \hat{\boldsymbol{M}}_m, \hat{\boldsymbol{M}}_h$, where $\hat{\boldsymbol{M}} = \{\hat{\boldsymbol{t}}, \hat{\boldsymbol{q}}\}$.
3. $\zeta(\hat{\boldsymbol{M}}, \boldsymbol{M}) = norm(\|\hat{\boldsymbol{t}}, \boldsymbol{t}\|_2) + 2\arccos(abs(\hat{\boldsymbol{q}} \cdot \boldsymbol{q}))/\pi$.
for each $\hat{\boldsymbol{M}}$ **do**
4. Find the closest image $\mathbf{I}'$ with $\arg\min_{I'} \zeta(\hat{\boldsymbol{M}}, \boldsymbol{M}_{\boldsymbol{I'}})$;
5. Construct pairs $(\mathbf{I}_a, \mathbf{I}')$;
end for
6. Train the PRP module with $\{(\mathbf{I}_a, \mathbf{I}'_e), (\mathbf{I}_a, \mathbf{I}'_m), (\mathbf{I}_a, \mathbf{I}'_h)\}$;
end for

3.2. Pose-based Fine Retrieval Module (PFR)

Now we introduce the PFR module as shown in the middle of Fig.2. It is known that it is difficult to accurately estimate the camera position and orientation by using RGB images [17, 7, 18]. However, we find that with a precise retrieval system, the relative pose regression task can be surprisingly simpler.

Formally, the ground-truth relative orientation denoted as $\boldsymbol{\Delta}_R = \boldsymbol{R}_1^\top \boldsymbol{R}_2$ and translation denoted as $\boldsymbol{t}_2 - \boldsymbol{t}_1$ are represented by the quaternion $\boldsymbol{\Delta}_q$ and $\boldsymbol{\Delta}_t$ following [7, 17, 18]. For each training batch, we obtain the relative pose $\hat{\boldsymbol{\Delta}}_{\boldsymbol{t}}$ and $\hat{\boldsymbol{\Delta}}_{\boldsymbol{R}}$ estimated by the regression part of our PFR module. Then, the coarse camera position and orientation of the anchor image can be estimated by computing $\boldsymbol{R}_2\hat{\boldsymbol{\Delta}}_R^{-1}$ and $\boldsymbol{t}_2 - \hat{\boldsymbol{\Delta}}_t$. With this coarse camera pose estimation, we retrieve the image that is closest to the estimated anchor pose from the database, and combine the retrieved image with the anchor image into a pair. By using this image pairs as input data, our precise PRP module learns learns in a easier way to make the pose estimation better. The pose-based retrieval algorithm is given in Algorithm 1.

We train our PFR module by using Euclidean loss with the following objective loss function,

$$L_{\text{PFR}} = \|\hat{\boldsymbol{\Delta}}_t - \boldsymbol{\Delta}_t\|_1 + \|\frac{\hat{\boldsymbol{\Delta}}_q}{\|\hat{\boldsymbol{\Delta}}_q\|} - \boldsymbol{\Delta}_q\|_1, \tag{8}$$

where $\hat{\boldsymbol{\Delta}}_t$ and $\hat{\boldsymbol{\Delta}}_q$ denote the predicted relative pose.

Note that [17, 18] used the $l2$ Euclidean norm. However, we find that $l1$ Euclidean norm is much better to estimate position and orientation. Here we simply set the weight of

Table 2: Ablation study on the 7-Scenes dataset. Median translation error (°) and rotation error (m) is reported.

Scene	Br	Br + Fl	Br + Al	Br + Bl	Nr	Nr + Pr	NR + Pr + Rp	Full
Chess	0.30m, 12.86°	0.25m, 12.93°	0.29m, 10.49°	0.23m, 11.22°	0.21m, 9.12°	0.08m, 4.40°	0.05m, 2.02°	**0.04**m, **1.73**°
Fire	0.35m, 15.09°	0.32m, 15.19°	0.33m, 11.03°	0.37m, 14.88°	0.24m, 10.37°	0.07m, 4.08°	0.04m, 2.02°	**0.03**m, **1.74**°
Heads	0.31m, 15.52°	0.19m, 17.94°	0.20m, 12.25°	0.21m, 14.46°	0.22m, 8.62°	0.09m, 4.97°	0.06m, 2.25°	**0.05**m, **1.98**°
Office	0.42m, 18.38°	0.27m, 13.24°	0.27m, 9.56°	0.29m, 11.46°	0.24m, 9.78°	0.08m, 4.20°	0.05m, 1.86°	**0.04**m, **1.62**°
Pumpkin	0.31m, 13.03°	0.29m, 13.22°	0.31m, 10.40°	0.30m, 11.25°	0.27m, 11.29°	0.07m, 3.85°	0.05m, 1.88°	**0.04**m, **1.64**°
RedKitchen	0.30m, 10.44°	0.26m, 13.59°	0.41m, 10.86°	0.28m, 11.70°	0.28m, 10.31°	0.08m, 3.86°	0.05m, 1.91°	**0.04**m, **1.63**°
Stairs	0.33m, 18.16°	0.22m, 16.41°	0.31m, 12.59°	0.32m, 10.63°	0.22m, 9.18°	0.08m, 3.91°	0.05m, 1.65°	**0.04**m, **1.51**°
Average	0.33m, 14.78°	0.26m, 14.65°	0.31m, 11.03°	0.28m, 12.23°	0.25m, 9.96°	0.08m, 4.18°	0.05m, 1.94°	**0.04**m, **1.69**°

the position and orientation to 1:1, resulting in nearly optimal results.

3.3. Precise Relative Pose Regression Module (PRP)

Compared to the model that estimates the relative pose directly such as [18, 3], our model improves the accuracy of camera re-localization by the pose-based retrieval. Through this module, the range of relative pose can be further reduced. Thus the relative pose regression task is surprisingly simpler.

We introduce the precise PRP module for relative pose regression. Although the PFR module and the PRP module have the same structure, the data distributions they learned are different, which make them used for two different tasks: the fine image retrieval and the fine pose regression. With the efficient data distribution, the relative pose regression can be learned more easily by our PRP module. The kernel density estimation distributions of the training data on 7-Scenes dataset for these two modules are shown in Figure 3. We train our PRP module with the same $l1$ norm as the above PFR module,

$$L_{\text{PRP}} = \|\hat{\mathbf{\Delta}}'_t - \mathbf{\Delta}'_t\|_1 + \|\frac{\hat{\mathbf{\Delta}}'_q}{\|\hat{\mathbf{\Delta}}'_q\|} - \mathbf{\Delta}'_q\|_1, \tag{9}$$

where $\mathbf{\Delta}'_t$ and $\mathbf{\Delta}'_q$ denote the relative pose.

3.4. Inference Stage

In this section, we discuss our inference framework. We first create a database using the key-frame of the training data. For each ground truth absolute camera pose $\boldsymbol{M}$, we store its corresponding retrieval descriptor f_r through the ICR module and two types of pose representations f_c, f_f through our PFR and PRP module, respectively. Our inference process is as follows:

- Given a query image, with the feature descriptor f_r^q obtained by our ICR module, we perform top K Nearest Neighbour search from the database, where K is 3 in all experiments.
- Coarse representation of query image f_c^q obtained by our PFR decoder and its top K ranked reference representations $\{f_c^1, ..., f_c^k\}$ are concatenated and fed to our PFR regressor to predict coarse estimated poses.
- For each estimated pose, we retrieve the closest pose from the database as the new reference pose. Fine representation of query image f_f^q obtained by our PRP decoder and its new reference representations $\{f_f^1, ..., f_f^k\}$ are concatenated and fed to our PRP regressor to predict accurate poses $\{\hat{\boldsymbol{M}}_f^1, ..., \hat{\boldsymbol{M}}_f^k\}$.
- For the inference of sequence data, we leverage two frames before the query frame and estimate the poses of query frame $\hat{\boldsymbol{M}}^{seq1}, \hat{\boldsymbol{M}}^{seq2}$ by our PRP regressor.
- Finally, the poses $\{\hat{\boldsymbol{M}}_f^1, ..., \hat{\boldsymbol{M}}_f^k, \hat{\boldsymbol{M}}^{seq1}, \hat{\boldsymbol{M}}^{seq2}\}$ are filtered by a RANSAC-based algorithm, as in [18]. The final result is averaged from all inliers.

Note that our three modules share a same encoder thus the encoder only need to be forwarded once. Compared to the encoder, the time spent in our three modules is little.

4. Experiments

4.1. Datasets

We evaluate our framework for camera re-localization on three benchmark datasets: the 7-Scenes [35], the RobotCar [22] and the ApolloScape datasets [14].

7-Scenes is a collection of tracked RGB-D camera frames of seven indoor environments. All scenes were recorded from a handheld Kinect RGB-D camera at 640×480 resolution. Multiple sequences were captured for each environment, and each sequence is 500 or 1000 frames.

Oxford RobotCar is a large-scale countryside-view dataset which contains over 100 repetitions of a consistent route (about 10km) through central Oxford captured twice a week over a period of over a year. Thus the dataset captures different combinations of weather, traffic, pedestrians, construction and roadworks. Following [7, 10], we used the LOOP subset of this dataset, with a total length of 1120m.

ApolloScape is a large-scale city-view dataset that consists of RGB videos, corresponding GPS and dense 3D point

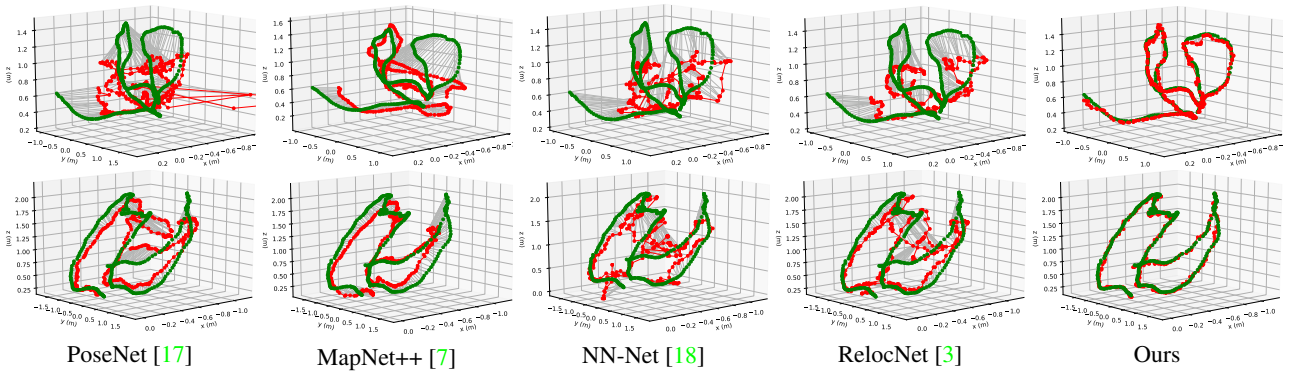


Figure 4: Camera localization results on Fire-seq-04 and Redkitchen-seq-12 sequences of the 7-Sevenscenes dataset. The ground truth camera trajectory is the green line and the red lines show the camera pose predictions.

Table 3: Median localization error on the 7-Scenes dataset. Note that [5, 30] are 3D-based methods.

Scene	PoseNet2 [16]	MapNet [7]	NN-Net [18]	RelocNet [3]	VLocNet [38]	DSAC [5]	Active Search [30]	Ours
Chess	0.13m, 4.48°	0.08m, 3.25°	0.13m, 6.46°	0.12m, 4.14°	0.04m, 1.71°	**0.02**m, **1.2**°	0.04m, 1.96°	0.04m, 1.73°
Fire	0.27m, 11.3°	0.27m, 11.69°	0.26m, 12.72°	0.26m, 10.40°	0.04m, 5.34°	0.04m, **1.5**°	**0.03**m, 1.53°	**0.03**m, 1.74°
Heads	0.17m, 13.0°	0.18m, 13.25°	0.14m, 12.34°	0.14m, 10.50°	0.05m, 6.64°	0.03m, 2.7°	**0.02**m, **1.45**°	0.05m, 1.98°
Office	0.19m, 5.55°	0.17m, 5.15°	0.21m, 7.35°	0.18m, 5.32°	**0.04**m, 1.95°	**0.04**m, **1.6**°	0.09m, 3.61°	**0.04**m, 1.62°
Pumpkin	0.26m, 4.75°	0.22m, 4.02°	0.24m, 6.35°	0.26m, 4.17°	**0.04**m, 2.28°	0.05m, 2.0°	0.08m, 3.10°	**0.04**m, **1.64**°
RedKitchen	0.23m, 5.35°	0.23m, 4.93°	0.24m, 8.03°	0.23m, 5.08°	**0.04**m, 2.20°	0.05m, 2.0°	0.07m, 3.37°	**0.04**m, **1.63**°
Stairs	0.35m, 12.4°	0.30m, 12.08°	0.27m, 11.82°	0.28m, 7.53°	0.10m, 6.48°	1.17m, 33.1°	**0.03**m, 2.22°	0.04m, **1.51**°
Average	0.23m, 8.12°	0.21m, 7.77°	0.21m, 9.30°	0.21m, 6.73°	0.05m, 3.80°	0.20m, 6.3°	0.05m, 2.46°	**0.04**m, **1.69**°

clouds. The dataset is recorded from six roads under different lighting conditions with stereo pair of images. Each road of the dataset has multiple records.

4.2. Implementation Details

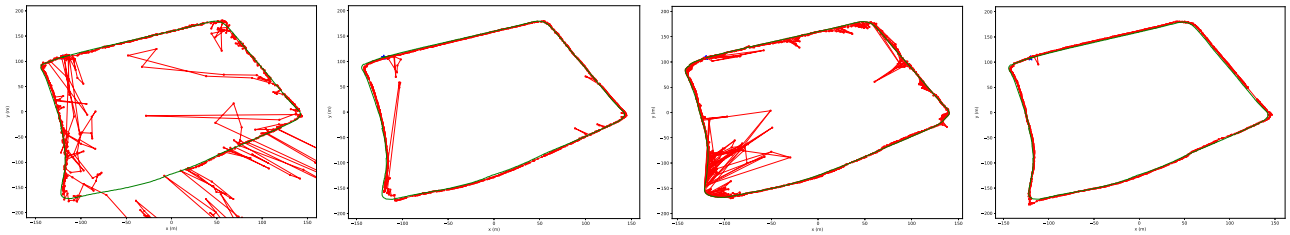
Specifically, the shared encoder contains layer 1-3 of ResNet34 [13] and layer 4 is duplicated for three modules. The weights are initialized and fine-tuned from the pre-trained model on ImageNet classification task. In each branch, the 512D features of two images are concatenated and then passed through the regressor which consists two fully connected layers (FC), where the dimension of the first FC layer is 512 and the second FC layer is task-specific. The FC layers are initialized randomly.

The training images are resized to 224×224 pixels. We perform flip, random gaussian blur and color jittering with a threshold 0.4 for data augmentation. The network is optimized by SGD, where momentum and weight decay are set to 0.9 and 0.0001 respectively. We take a mini-batch size N of 128 on 8 TITAN Xp GPUs with synchronous Batch Normalization. We use the 'poly' learning rate policy and set base learning rate to 0.01 and power to 0.9. The maximum number of epochs for training process is set to 300.

4.3. Ablation Study

Our coarse-to-fine framework has three stages: image-based retrieval with novel losses, pose-based fine retrieval, final pose estimation. To show the contribution of each part of our full CamNet, we make comparison to its seven simplified versions: (1) Br – the basic retrieval results used in [18]. (2) Br + Fl – the retrieval results with bilateral frustum loss. (3) Br + Al – the retrieval results with angle-based loss. (4) Br + Bl – the retrieval results with batch hard sampling loss. (5) Nr – our novel retrieval results with all losses. (6) Nr + Pr – the estimated pose of our PFR module. (7) NR + Pr + Rp – the estimated pose of our PRP module. (8) Full – the final pose of our framework averaged from all inliers.

The ablation study results are presented in Table 2. It can be seen that: (1) The performance continuously increases when more components are used for camera re-localization, showing the contribution of each part. (2) The original retrieval is the bottleneck of localization accuracy. The retrieval performance has been greatly improved by our three losses. (3) Separating retrieval and relative pose regression into a multi-branch architecture with shared weights benefits both tasks. (4) Our bilateral frustum loss focuses on improving the retrieval performance of translation, while our angle-based loss focuses on improving the retrieval performance of orientation. The batch hard sampling greatly improves the retrieval performance of the confusing scene, such as Stairs. (5) The estimated pose of our PFR module can be further improved by our PRP module, which shows the effectiveness of our coarse-to-fine framework.



PoseNet [17] (25.29m, 17.45°) MapNet++ [7] (6.73m, 2.23°) NN-Net [18] (11.37m, 4.57°) Ours (2.56m, 1.54°)

Figure 5: Camera localization results on the LOOP scene of the Oxford RobotCar dataset. The ground truth camera trajectory is the green line and the red lines show the camera pose predictions. The caption shows the mean translation error (m) and mean rotation error (°).

4.4. Comparative results

4.4.1 Experiments on the Indoor Dataset

We compare our coarse-to-fine model to the state-of-the-art alternatives [17, 16, 7, 18, 38, 5, 30, 35, 3] on the 7-Scenes dataset. Since 7-Scenes is a indoor dataset with 3D models, both image-based methods and 3D model-based methods are included. Following same convention of prior work [17, 7, 18], we use ResNet34 [13] as our base network and compute the median error for camera translation and rotation. Table 3 shows the quantitative comparisons. Since the calculation of our frustum overlap requires depth information, we provide a version that uses ORB similarity rather than frustum overlap. The median localization error of our RGB trained model for 7-Scenes is $\mathbf{0.05}$m, $\mathbf{1.83}^\circ$, which is superior to the state-of-the-art image-based methods.

We observe that: our model not only outperforms the state-of-the-art image-based methods by a large margin, but also yields comparable results with 3d-based models. Figure 4 shows the camera trajectories for testing sequences in the 3D plot. It can be seen that our model fits the camera trajectory best and significantly outperforms other image-based models.

4.4.2 Experiments on the Outdoor Dataset

Since 3D-based models, such as [6, 5, 35], fail to handle large-scale outdoor scenes, we compare our coarse-to-fine model to the image-based deep models [17, 7, 18, 38, 3] on the RobotCar and ApolloSpace datasets. We train and test our model using the same settings as on 7-Scenes dataset.

Table 4 shows quantitative comparisons on ApolloSpace dataset. In experiments 'road11' and 'road12', we use different records of the same road for training and testing. We also conduct generalized setting which is trained on road11 but tested on road12 to show the generalization ability of our model. Note that this setting can only be used on retrieval-based methods. It can be seen that our model outperforms the state-of-the-art image-based methods and shows excellent generalization ability.

Figure 5 shows the camera trajectories for testing sequences of RobotCar dataset. We observe that: (1) Although the retrieval-based approach [18] performs well on the ApolloSpace dataset, it fails to process the RobotCar set. This is because ApolloSpace is a city-view dataset that is easier to retrieve, while RobotCar is countryside-view, that all roads are similar and the retrieval is difficult. Compared with [18], we show huge improvements through our carefully designed two-stage retrieval module. (2) Our model outperforms the methods of learning absolute pose. MapNet++ [7] uses pose-graph optimization (PGO) to refine the poses agree with the input visual odometries and obtain smoother results. However, our model uses a combination of relative poses between sequences and those of query image and retrieved images and to get better results.

Table 4: Quantitative comparisons on AppoloSpace dataset. 'generalized' means the setting that the model is trained on road11 but tested on road12.

Method	road11	road12	generalized
PoseNet [17]	13.85m, 3.49°	11.24m, 3.55°	–
MapNet [7]	8.30m, 2.77°	6.83m, 2.72°	–
NN-Net [18]	6.90m, 3.28°	6.34m, 3.33°	16.60m, 3.49°
Ours	**5.24**m, **2.57**°	**5.19**m, **2.70**°	**8.63**m, **2.97**°

5. Conclusion

In summary, we present a coarse-to-fine retrieval-based deep learning framework (CamNet) with our three modules: the image-based coarse retrieval module, the pose-based fine retrieval module and the precise relative pose regression module. We show that the relate pose regression task can be surprisingly simpler with our carefully designed modules. We also design novel retrieval losses and refinement algorithm for our coarse-to-fine network. We show that different loss functions benefit the retrieval in different ways and data distribution is critical to retrieval accuracy. Extensive experiments show that our model outperforms the state-of-the-art methods by a large margin on both indoor and outdoor datasets.

Acknowledgement. Ping Luo is partially supported by the HKU Seed Funding for Basic Research and SenseTime's Donation for Basic Research.

References

[1] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5297–5307, 2016. 3

[2] R. Arandjelović and A. Zisserman. Dislocation: Scalable descriptor distinctiveness for location recognition. In *Asian Conference on Computer Vision*, pages 188–204. Springer, 2014. 3

[3] V. Balntas, S. Li, and V. Prisacariu. Relocnet: Continuous metric learning relocalisation using neural nets. In *The European Conference on Computer Vision (ECCV)*, September 2018. 1, 2, 3, 4, 6, 7, 8

[4] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool. Speeded-up robust features (surf). *Computer Vision & Image Understanding*, 110(3):346–359, 2008. 1

[5] E. Brachmann, A. Krull, S. Nowozin, J. Shotton, F. Michel, S. Gumhold, and C. Rother. Dsac-differentiable ransac for camera localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6684–6692, 2017. 1, 2, 7, 8

[6] E. Brachmann and C. Rother. Learning less is more-6d camera localization via 3d surface regression. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4654–4662, 2018. 1, 2, 8

[7] S. Brahmbhatt, J. Gu, K. Kim, J. Hays, and J. Kautz. Geometry-aware learning of maps for camera localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2616–2625, 2018. 1, 2, 3, 5, 6, 7, 8

[8] M. Bui, S. Albarqouni, S. Ilic, and N. Navab. Scene coordinate and correspondence learning for imabe-based localization. *arXiv preprint arXiv:1805.08443*, 2018. 2

[9] T. Cavallari, S. Golodetz, N. A. Lord, J. Valentin, L. Di Stefano, and P. H. Torr. On-the-fly adaptation of regression forests for online camera relocalisation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4457–4466, 2017. 1, 2

[10] R. Clark, S. Wang, A. Markham, N. Trigoni, and H. Wen. Vidloc: A deep spatio-temporal model for 6-dof video-clip relocalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6856–6864, 2017. 3, 6

[11] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981. 2

[12] H. Germain, G. Bourmaud, and V. Lepetit. Efficient condition-based representations for long-term visual localization. *arXiv preprint arXiv:1812.03707*, 2018. 3

[13] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 7, 8

[14] X. Huang, X. Cheng, Q. Geng, B. Cao, D. Zhou, P. Wang, Y. Lin, and R. Yang. The apolloscape dataset for autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 954–960, 2018. 6

[15] A. Kendall and R. Cipolla. Modelling uncertainty in deep learning for camera relocalization. In *2016 IEEE international conference on Robotics and Automation (ICRA)*, pages 4762–4769. IEEE, 2016. 1, 3

[16] A. Kendall and R. Cipolla. Geometric loss functions for camera pose regression with deep learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5974–5983, 2017. 1, 2, 3, 7, 8

[17] A. Kendall, M. Grimes, and R. Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *Proceedings of the IEEE international conference on computer vision*, pages 2938–2946, 2015. 1, 2, 5, 7, 8

[18] Z. Laskar, I. Melekhov, S. Kalia, and J. Kannala. Camera relocalization by computing pairwise relative poses using convolutional neural network. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 929–938, 2017. 1, 2, 3, 5, 6, 7, 8

[19] X. Li, J. Ylioinas, and J. Kannala. Full-frame scene coordinate regression for image-based localization. *arXiv preprint arXiv:1802.03237*, 2018. 2

[20] X. Li, J. Ylioinas, J. Verbeek, and J. Kannala. Scene coordinate regression with angle-based reprojection loss for camera relocalization. In *European Conference on Computer Vision*, pages 229–245. Springer, 2018. 2

[21] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004. 2

[22] W. Maddern, G. Pascoe, C. Linegar, and P. Newman. 1 year, 1000 km: The oxford robotcar dataset. *The International Journal of Robotics Research*, 36(1):3–15, 2017. 6

[23] D. Massiceti, A. Krull, E. Brachmann, C. Rother, and P. H. Torr. Random forests versus neural networkswhat's best for camera localization? In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5118–5125. IEEE, 2017. 1, 2

[24] L. Meng, F. Tung, J. J. Little, J. Valentin, and C. W. de Silva. Exploiting points and lines in regression forests for rgb-d camera relocalization. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6827–6834. IEEE, 2018. 2

[25] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE transactions on robotics*, 31(5):1147–1163, 2015. 1

[26] N. Radwan, A. Valada, and W. Burgard. Vlocnet++: Deep multitask learning for semantic visual localization and odometry. *IEEE Robotics and Automation Letters*, 3(4):4407–4414, 2018. 2

[27] I. Rocco, M. Cimpoi, R. Arandjelović, A. Torii, T. Pajdla, and J. Sivic. Neighbourhood consensus networks. In *Proceedings of the 32nd Conference on Neural Information Processing Systems*, 2018. 2

[28] E. Rublee, V. Rabaud, K. Konolige, and G. R. Bradski. Orb: an efficient alternative to sift or surf. In *International Conference on Computer Vision*, 2012. 1

[29] T. Sattler, M. Havlena, K. Schindler, and M. Pollefeys. Large-scale location recognition and the geometric burstiness problem. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1582–1590, 2016. 3

[30] T. Sattler, B. Leibe, and L. Kobbelt. Efficient & effective prioritized matching for large-scale image-based localization. *IEEE transactions on pattern analysis and machine intelligence*, 39(9):1744–1756, 2017. 2, 7, 8

[31] T. Sattler, W. Maddern, C. Toft, A. Torii, L. Hammarstrand, E. Stenborg, D. Safari, M. Okutomi, M. Pollefeys, J. Sivic, et al. Benchmarking 6dof outdoor visual localization in changing conditions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8601–8610, 2018. 2

[32] T. Sattler, A. Torii, J. Sivic, M. Pollefeys, H. Taira, M. Okutomi, and T. Pajdla. Are large-scale 3d models really necessary for accurate visual localization? In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1637–1646, 2017. 2

[33] T. Schmidt, R. Newcombe, and D. Fox. Self-supervised visual descriptor learning for dense correspondence. *IEEE Robotics and Automation Letters*, 2(2):420–427, 2017. 2

[34] J. L. Schönberger, M. Pollefeys, A. Geiger, and T. Sattler. Semantic visual localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6896–6906, 2018. 2

[35] J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi, and A. Fitzgibbon. Scene coordinate regression forests for camera relocalization in rgb-d images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2930–2937, 2013. 1, 2, 6, 8

[36] H. Taira, M. Okutomi, T. Sattler, M. Cimpoi, M. Pollefeys, J. Sivic, T. Pajdla, and A. Torii. Inloc: Indoor visual localization with dense matching and view synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7199–7209, 2018. 2

[37] A. Torii, R. Arandjelovic, J. Sivic, M. Okutomi, and T. Pajdla. 24/7 place recognition by view synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1808–1817, 2015. 3

[38] A. Valada, N. Radwan, and W. Burgard. Deep auxiliary learning for visual localization and odometry. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6939–6946. IEEE, 2018. 1, 2, 3, 7, 8

[39] A. Valada, N. Radwan, and W. Burgard. Incorporating semantic and geometric priors in deep pose regression. In *Workshop on Learning and Inference in Robotics: Integrating Structure, Priors and Models at Robotics: Science and Systems (RSS)*, 2018. 1, 3

[40] J. Valentin, M. Nießner, J. Shotton, A. Fitzgibbon, S. Izadi, and P. H. Torr. Exploiting uncertainty in regression forests for accurate camera relocalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4400–4408, 2015. 1, 2

[41] F. Walch, C. Hazirbas, L. Leal-Taixe, T. Sattler, S. Hilsenbeck, and D. Cremers. Image-based localization using lstms for structured feature correlation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 627–637, 2017. 1, 2

[42] Q. Xiao, H. Luo, and C. Zhang. Margin sample mining loss: A deep learning based method for person re-identification. *arXiv preprint arXiv:1710.00478*, 2017. 5

[43] K. M. Yi, E. Trulls, V. Lepetit, and P. Fua. Lift: Learned invariant feature transform. In *European Conference on Computer Vision*, pages 467–483. Springer, 2016. 2

[44] K. M. Yi, E. Trulls Fortuny, Y. Ono, V. Lepetit, M. Salzmann, and P. Fua. Learning to find good correspondences. In *Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, number CONF, 2018. 2