

Visual Localization via Few-Shot Scene Region Classification

Siyang Dong^{1,3*} Shuzhe Wang^{2,3*} Yixin Zhuang⁴
 Juho Kannala² Marc Pollefeys^{3,5} Baoquan Chen⁶
¹Shandong University ²Aalto University ³ETH Zurich
⁴Fuzhou University ⁵Microsoft ⁶Peking University

Abstract

Visual (re)localization addresses the problem of estimating the 6-DoF (Degree of Freedom) camera pose of a query image captured in a known scene, which is a key building block of many computer vision and robotics applications. Recent advances in structure-based localization solve this problem by memorizing the mapping from image pixels to scene coordinates with neural networks to build 2D-3D correspondences for camera pose optimization. However, such memorization requires training by amounts of posed images in each scene, which is heavy and inefficient. On the contrary, few-shot images are usually sufficient to cover the main regions of a scene for a human operator to perform visual localization. In this paper, we propose a scene region classification approach to achieve fast and effective scene memorization with few-shot images. Our insight is leveraging a) pre-learned feature extractor, b) scene region classifier, and c) meta-learning strategy to accelerate training while mitigating overfitting. We evaluate our method on both indoor and outdoor benchmarks. The experiments validate the effectiveness of our method in the few-shot setting, and the training time is significantly reduced to only a few minutes.¹

1. Introduction

Visual (re)localization is a key component of many computer vision and robotics applications such as Augmented Reality (AR) and navigation. It addresses the problem of estimating the 6-DoF (Degree of Freedom) camera pose of a query image captured in a known scene. There are mainly two types of approaches: direct pose estimation by image retrieval [4, 5, 37, 44] or pose regression [24, 25, 38, 49, 50], and two-step pose estimation (also known as structure-based localization) [7–9, 15, 22, 28, 29, 33–36, 43, 51, 54] that first infers correspondences between image pixels and scene points and then solves camera pose by geometric optimization.

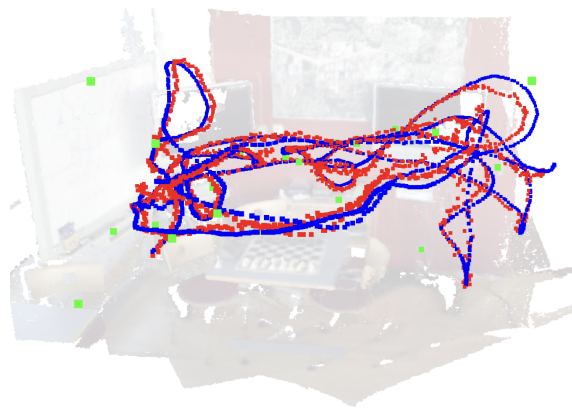


Figure 1. Visualization of our estimated (red dots) and ground truth (blue dots) camera poses in the scene CHES [39]. Our method is trained with 20 images (green dots) uniformly sampled from the original 4000 training images. We observe that the 20 images are enough to cover the main regions of the scene and construct a coarse 3D model (shown in the background) to support our method.

During the last decade, scene coordinate regression based two-step approaches [9, 22, 28, 43] achieve state-of-the-art localization accuracy on public benchmarks [25, 39, 46], which is the main focus of this paper.

The essential concept of scene coordinate regression is to memorize the mapping from image pixels to scene coordinates under a variety of viewpoints. A common way is to leverage convolutional neural networks (CNNs) and encode the aforementioned mapping as the network parameters. As a result, most of them require amounts of posed images to train their models in each specific scene, therefore can hardly be deployed in practice. On the contrary, few-shot images are usually sufficient to cover the main regions of a scene, as shown in Figure 1, and sufficient for a human operator to memorize a scene for future localization purposes. One hypothesis is that we humans have learned during daily life the prior knowledge to extract robust visual features, and only have to memorize the locations of different features in a novel scene. Incorporating the aforementioned insights into the scene coordinate regression based method, in this paper,

*Joint first authors

¹Code available at: <https://github.com/siyandong/SRC>

we study the fast memorization of novel scenes with only few-shot posed images as training data.

Introducing pre-learned visual feature extractors [13, 15, 32, 42] is not new in the visual localization community. Especially, as another popular series of the two-step pose estimation, the feature matching based methods make use of off-the-shelf feature detectors and descriptors to build scene models from posed images, to match query image features and scene coordinates. They are in nature generalized to different scenes as the visual features are scene-agnostic. However, they usually cost large memory and are not accurate as the scene-specific coordinate regression based methods. Therefore, we propose a new paradigm that combines scene-agnostic features and scene-specific coordinate estimation.

Our key idea is that the scene-agnostic feature extractor is helpful for fast scene memorization, while the scene-specific training also helps with accuracy and efficiency. Such insight motivates us to decouple the popular end-to-end scene coordinate regression pipeline into a scene-agnostic feature extractor and a scene-specific coordinate estimator. Further, we propose a scene region classification paradigm instead of direct coordinate regression for coordinate inference, so that a meta-learning strategy is applied to accelerate training.

We summarize our contributions as follows:

- We propose a novel problem setting, i.e. visual localization with only few-shot posed images as the database, along with a simple and effective method designed for the proposed few-shot setting.
- Leveraging both scene-agnostic and scene-specific information, we introduce scene region classification and meta-learning strategy for fast scene memorization.
- Experiments validate the effectiveness of our method. In the few-shot setting, we outperform the state-of-the-art scene coordinate regression based methods, with only a few minutes of training time.

2. Related Work

Direct pose estimation. This type of approach directly estimates the camera pose based on the query image. An effective direction is based on image retrieval methods [4, 5, 37, 44]. They aggregate either hand-crafted or learned local features into whole-image global descriptors to match the query image against the database. PoseNet [25] and its variants [24, 38, 49, 50] make use of neural networks to directly regress camera pose. There are also RGB-D based methods [12, 20, 45] that encode both color and depth information into global features. Although they are efficient, the localization results are not as accurate as the two-step pose estimation methods. Therefore, the approaches of direct pose estimation are not compared in this paper.

Two-step pose estimation. This type of approach first infers correspondences between image pixels and scene points and then solves camera pose by optimization. To obtain the aforementioned 2D-3D correspondences, conventional feature matching based methods [15, 33–36] leverage either hand-crafted or learned keypoint features to explicitly construct scene maps to match the query image’s keypoints. Another popular direction is scene coordinate regression based methods [7–9, 22, 28, 29, 51] that estimate the 2D-3D correspondences implicitly by memorizing the mapping from image pixels to scene coordinates. Such scene-specific memorization designs usually achieve more accurate results than scene-agnostic feature matching, while they need to be trained for each novel scene. Recent works [43, 54] propose to regress scene coordinate based on retrieved training image coordinate. There are also RGB-D based visual localization approaches [10, 11, 14, 39, 47] leveraging random forests or point cloud based backbones, with 3D-3D correspondence optimization algorithms, which is beyond our scope.

Few-shot learning. Few-shot learning, also known as low-shot learning, is the learning paradigm with only a few training data. There are roughly three types of approaches: data hallucination based methods [3, 21, 52] that leverage generators to augment training data, metric learning based methods [27, 41, 48] that learn feature representations for similarity comparisons, and meta-learning based methods [2, 17, 30] that aim to find a proper initialization for fast training on a new task. In this paper, we adopt the meta-learning strategy and obtain the initial network parameters to perform fast memorization of novel scenes.

3. Method

Problem statement. Visual localization is a scene-specific problem, with the requirement of a scene-specific database as the reference for camera pose estimation. Specifically, in each scene, a set of posed RGB-D images are given as the database to estimate the 6 DoF camera pose of single query RGB images. Different from the previous settings [9, 28], there are only a few rather than thousands of training images given in our problem. The assumption is that tens of images are sufficient to cover the major regions of a scene to support visual localization.

Method overview. Our method follows the two-step pose estimation framework and is a variant of scene coordinate regression. The framework first builds the 2D-3D correspondences between image pixels and scene coordinates and then optimizes the camera poses with a principled geometric algorithm. Figure 2 illustrates our training pipeline, which adopts a from-agnostic-to-specific structure. Given the few-shot RGB-D training images, we first build a hierarchical scene partition tree to label each image pixel a set of region IDs. Then, instead of directly regressing the scene coordinates, we employ a neural network to map the image pixels

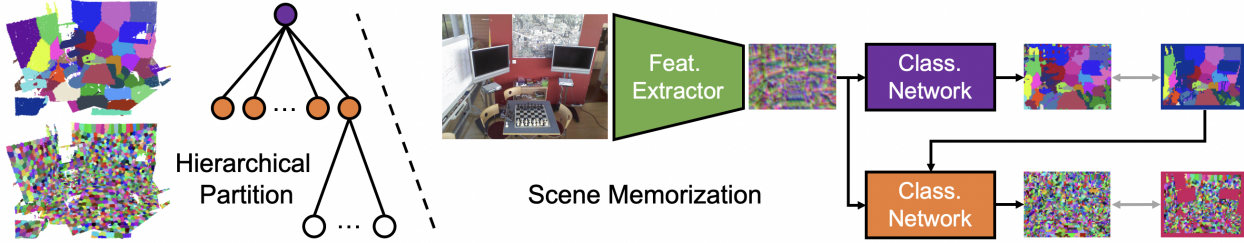


Figure 2. The training pipeline of our method. A hierarchical partition tree is built to divide the scene to regions, and then a neural network is trained to map input image pixels to region labels. The network is designed to leverage both scene-agnostic priors (i.e. the feature extractor SuperPoint [13]) and scene-specific memorization (i.e. the classifier that consists of hierarchical classification networks).

to their corresponding region IDs. Particularly, the network is composed of two components: a scene-agnostic visual feature extractor f (3.1) and a scene-specific region classifier c (3.2). Figure 3 illustrates our camera pose estimation pipeline. Through hierarchical classification, an input 2D pixel is mapped to a leaf node, i.e. a region consisting of a compact set of scene coordinates, as its 3D correspondences. Such one-to-many correspondences are fed to a Perspective-n-Point (PnP) [19, 23] algorithm inside a RANSAC [18, 38] loop for camera pose optimization (3.3).

3.1. Scene-Agnostic Feature Extractor

Recent work [38] interprets the direct pose regression networks as linear combinations of embedded features. In this work, we follow a similar hypothesis to interpret the typical scene coordinate regression network SCRNet [28]: the first-to-middle layers serve as a descriptor extractor and the rest layers serve as a coordinate regressor. The reason for requiring amounts of training data is that the visual descriptors should be learned to be robust under different viewpoints (please refer to 4.4 scene coordinate regression). Since recently learned descriptors [13, 15, 32, 42] achieve promising visual feature embedding ability under viewpoint changes and cross multiple scenes, we make use of the off-the-shelf SuperPoint [13] (without detector) as our feature extractor, so that we can obtain robust semi-dense feature maps from an input image. Such a feature extractor f is scene-agnostic and can be directly deployed in novel scenes without any training. Specifically, the feature extractor f is a VGG-Style [40] network that inputs an image $I_{H \times W}$ and outputs its feature map $F_{H_o \times W_o} = f(I_{H \times W})$, where $H_o = H/8$ and $W_o = W/8$. The output features are regarded as robust descriptors for 8×8 image patches and serve as the input for our scene-specific region classifier.

3.2. Scene-Specific Region Classifier

What makes visual localization a scene-specific problem is memorizing the scene-specific coordinate system. Direct coordinate regression takes time to converge even with only few-shot training images (please refer to 4.4 training times). Instead, in this subsection, we introduce a hierarchical scene

region classification approach for fast memorization of the mapping from image pixels to scene regions.

Scene partition tree. The goal of the scene partition is to divide scene coordinates into clusters in order to convert the task of coordinate regression to region classification. Given a set of RGB-D training images, we first fuse a scene point cloud according to their depth images and camera poses. On top of the scene point cloud, we build a hierarchical partition tree. Specifically, we run the classic K-Means algorithm to obtain m clusters for the first level of region partition. By iteratively executing region partition for each cluster, we eventually obtain a n -level m -way tree. Each node in the tree corresponds to a specific scene region, and each image pixel is automatically labeled by hierarchical node IDs.

Hierarchical classification networks. The classifier c aims to map a pixel to a leaf node. It consists of hierarchical classification networks, each of which performs a m -class classification. The whole process is denoted by $P = c(f(I))$, where $P \in \{1, 2, \dots, m\}^{H_o \times W_o}$ denotes the final classification map for the input feature map. Specifically, each level l in the tree corresponds to a classification network c_l . For the first level (the root node), the network c_1 takes the image feature map as input and outputs m -class probabilities:

$$P_1 = c_1(F), \quad (1)$$

where $P_1 \in (0, 1)_{m \times H_o \times W_o}$ denotes the classification probability map. For the rest levels, each network c_l inputs both image feature map and predicted classification probability from previous levels:

$$P_l = c_l(F, P_{1, \dots, l-1}), \quad (2)$$

where $P_l \in (0, 1)_{m \times H_o \times W_o}$. The final classification for the leaf nodes is computed by

$$P = \sum_{i=1}^n a(P_i) * m^{n-i}, \quad (3)$$

where $a(\cdot)$ denotes the argmax operator at the first channel to convert the one-hot probability to a class label.

Following recent works [14, 28], the classification networks are implemented as base and hyper networks. As

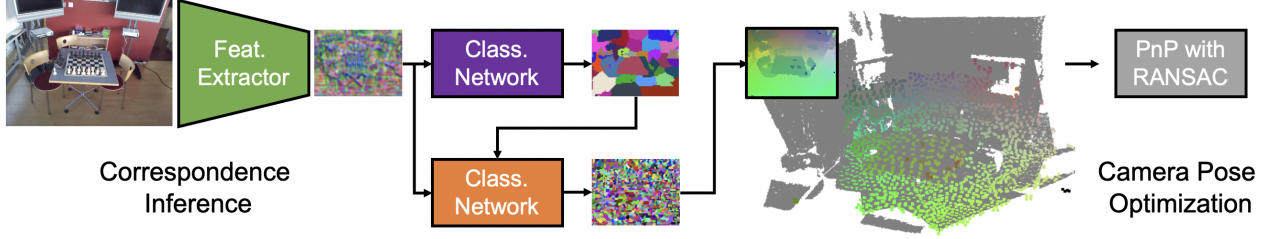


Figure 3. The camera pose estimation pipeline of our method. Given a query image, the trained network infers correspondences between image pixels and scene regions. Since each scene region corresponds to a set of scene coordinates, 2D-3D correspondences are built between image pixels and scene coordinates. Followed by a PnP algorithm with RANSAC, the camera pose is solved by optimization.

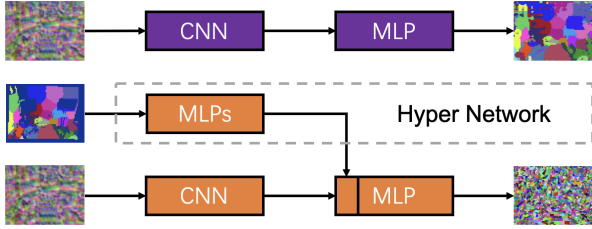


Figure 4. Illustration of the scene-specific region classifier. It consists of hierarchical classification networks. For the first level, it is only a base network. For the rest levels, each network contains both a base network and a hyper network.

shown in Figure 4, at the first level, the feature maps from the previous extraction module are fed to a CNN to extract scene-specific feature patterns, which are followed by a pixel-wise MLP to output the classification probability map. Starting from the second level, the feature patterns are modulated by a hyper network, according to the classification probability from previous levels. The intuition of the modulation is that similar feature patterns appearing in different scene regions should be classified under different labels. Since different levels handle the feature patterns on different scales, the corresponding CNNs are implemented with different receptive fields. Specifically, we make use of dilated convolutions to control the receptive fields, and the dilation parameters are computed according to the average cluster radius at each level. Please refer to 4.2 for implementation details.

Training with meta-learning. We supervise the training process at each classification output with a pixel-wise cross-entropy loss. The loss term is formulated as:

$$Loss_{i,j}(P_i^{(j)}, Y_i^{(j)}) = - \sum_{d=1}^m \log \frac{\exp(P_{i,d}^{(j)})}{\sum_{k=1}^m \exp(P_{i,k}^{(j)})} \mathbb{1}(Y_i^{(j)} = d), \quad (4)$$

where $P_i^{(j)} \in (0, 1)_m$ denotes the m -class probability prediction of pixel j at level i , $Y_i^{(j)} \in \{1, 2, \dots, m\}$ denotes

the ground truth label, and $\mathbb{1}(\cdot)$ denotes a binary function. Note that at the training stage, we use the ground truth label map as the input for each hyper network. Therefore, the multi-level classification network can be trained in parallel at each branch to reduce the training time.

Inspired by the recent meta-learning idea, we apply Reptile [30] strategy to initialize the hierarchy network with a pre-trained model. The pre-training aims to find proper network parameters so that for a novel scene the network can achieve fast convergence. This process makes use of t individual scenes as a set of hierarchical classification tasks $T = \{T_i | i = 1, 2, \dots, t\}$. The classifier parameters ϕ_{beg} are randomly initialized and iteratively updated by the Reptile gradient. For each iteration i , a task T_i is randomly sampled from T to perform k steps of stochastic gradient descent (SGD) operation, starting with ϕ_{i-1} , and resulting in $\tilde{\phi}_{i-1}$. The Reptile gradient G_i is computed as

$$G_i = \tilde{\phi}_{i-1} - \phi_{i-1}. \quad (5)$$

Then, the network parameters are actually updated as

$$\phi_i = \phi_{i-1} + \epsilon G_i. \quad (6)$$

The iteration step is repeated until convergence. Although there may be conflict among different tasks, Reptile gradients make the training process converge and we obtain ϕ_{end} . Then in each few-shot scene, we apply ϕ_{end} as our initial network parameters to perform fast memorization.

3.3. Camera Pose Optimization

Our localization pipeline leverages the trained network to build 2D-3D correspondences between image pixels and scene regions to perform the camera pose optimization. Each pixel is mapped to a leaf node in the scene partition tree, which corresponds to a specific scene region with a set of 3D coordinates. To limit the number of coordinates, we further partition each leaf node to q clusters by the K-Means algorithm and utilize the cluster centers for final pose estimation. Therefore, a 2D pixel $x_i \in \mathbb{N}^2$ is corresponded to a set of q 3D coordinates $X_i = \{X_{i,k} \in \mathbb{R}^3 | k = 1, 2, \dots, q\}$.

Let $M = \{(x_i, X_i) | i = 1, 2, \dots, N\}$ denote the set of 2D-3D correspondences. In contrast to the recent works [8, 28]

for applying the PnP solver, our camera pose optimization is a variant of PnP-RANSAC algorithm to handle one-to-many correspondences. Our algorithm consists of three stages, which are introduced in detail below.

Hypotheses Generation. The goal is to generate a set of camera pose hypothesis $H = \{H_i \in \mathbf{SE}(3) | i = 1, 2, \dots, h\}$. We first randomly sample 4 correspondences from M . For each sampled pixel x_i , we randomly select a 3D coordinate $X_{i,k}$ from X_i to form a one-to-one match. The 4 one-to-one matches formulate a minimal set for a PnP algorithm [19, 23] to solve a unique camera pose. This process is executed h times to obtain the hypotheses set H .

Ranking. We then select the hypothesis that reaches the most consensus with the input correspondences M . For each generated hypothesis H_i , we first compute the reprojection error for each correspondence indexed by j as

$$e_j(H_i, M_j) = \min_{k=1}^q \|x_j - Proj(H_i X_{j,k})\|_2. \quad (7)$$

Then we apply a kernel function to the pixel-wise reprojection errors and sum them to obtain the consensus score:

$$S_i(H_i, M) = \sum_{j=1}^N \frac{1}{1 + \exp(-0.5 * (e_j(H_i, M_j) - \tau))}, \quad (8)$$

where $\tau = 10$ is a threshold in the pixel coordinate system. We sort all the hypotheses by their scores and select the one with the lowest score as the output camera pose.

Refinement. This is the post-processing of the selected hypothesis to obtain a fine-grained camera pose as the final output. The refinement is executed iteratively. In each iteration, we recompute the reprojection error and select inlier correspondences whose error is less than the threshold τ . Then these inliers are fed into another PnP solver with Levenberg-Marquardt optimization [16] to refine the camera pose. The iteration terminates when the camera pose converges, or until the iterations reach a max limit of 20 steps.

4. Experiments

In this section, we validate the effectiveness of our method. We first introduce the few-shot version of datasets (4.1), and describe implementation details (4.2). Then we perform comparisons with state-of-the-art methods (4.3). Last but not the least, we analyze our method against our motivation and conduct several ablation studies (4.4).

4.1. Datasets

We evaluate our method on two standard benchmarks, 7-Scenes dataset [39] and Cambridge landmarks [25]. The

7-Scenes dataset consists of 7 indoor scenes with RGB-D images while the Cambridge landmarks contain 6 outdoor scenes. As the scene STREET in Cambridge failed to provide accurate 3D reconstruction, following the previous work [9, 28], we conduct experiments only on the rest 5 scenes. Besides the two datasets, we leverage the 12-Scenes dataset [46] to pre-train our classification network with Reptile [30] for model initialization.

In our few-shot setting, we uniformly sample $0.5 \sim 1.0\%$ images from the original training set. The selected training numbers are presented in Table 1 and Table 2.

4.2. Implementation Details

For all the scenes, we apply the hierarchical K-Means algorithm [31] and implement our scene partition as 2-level trees. Therefore, each scene-specific region classifier consists of two levels of classification networks. By default, we set $m = 64$ for 7-Scenes dataset, and $m = 100$ for Cambridge landmarks. Therefore, each indoor scene is divided into 4096 regions, and each outdoor scene is divided into 10000 regions. To construct the partition tree, we make use of the few-shot depth images and their camera poses to fuse the scene point cloud. Specifically, we use original depth images for 12-Scenes, calibrated depth images for 7-Scenes, and rendered depth images for Cambridge. The calibrated and rendered depth images are provided by DSAC++ [8].

The input image resolution to our feature extractor f is fixed to 480×640 (default resolution of 12-Scenes and 7-Scenes datasets). The images from Cambridge landmarks are resized to 480×852 and randomly cropped to 480×640 for training, and centered cropped for inference. The output feature map from extractor f is of shape $256 \times 60 \times 80$, where 256 is the dimension of the features. Then, the extracted map is fed to the scene-specific region classifier c and outputs the region predictions with the shape of $m \times 60 \times 80$. Each classification module consists of a 2-layer CNN and a 2-layer MLP. The dilation of each Convolution layer is set to 5 for the first level, 3 for the second level. The hyper network consists of two 2-layer MLPs to generate feature normalization parameters γ and β ($\gamma, \beta \in \mathbb{R}^{256 \times 60 \times 80}$). The feature modulation is implemented as

$$F_{out} = \gamma * F_{in} + \beta. \quad (9)$$

Except for the final output layer, each layer is followed by a layer normalization [6, 53] and a ReLU activation [1].

We set the batch size to 1 and the learning rate to $5e - 4$ in all of the training procedures. For Reptile pre-training on the 12-Scenes dataset, the number of tasks t is set to 12. We use $k = 2$ SGD steps, and $\epsilon = 5e - 4$. For the fast memorization on 7-Scenes and Cambridge, we apply the Adam optimization algorithm [26]. As for inference, the number of leaf coordinates is limited to $q = 10$ for both 7-Scenes and Cambridge. The number of hypotheses is set

Methods	Original training (median pose error in cm ^o)						Few-shot training (median pose error in cm ^o)					
	# Images	AS [36] [†]	HLoc [33,34]	SCRNet [28]	HSCNet [28]	DSAC* [9]	# Images	HLoc [33,34]	DSAC* [9]	HSCNet [28]	SP+Reg	Ours
CHESS	4000	3/0.87	2/0.85	2/0.7	2/0.7	2/1.10	20	4/1.42	3/1.16	4/1.42	4/1.28	4/1.23
FIRE	2000	2/1.01	2/0.94	2/0.9	2/0.9	2/1.24	10	4/1.72	5/1.86	5/1.67	5/1.95	4/1.53
HEADS	1000	1/0.82	1/0.75	1/0.8	1/0.9	1/1.82	10	4/1.59	4/2.71	3/1.76	3/2.05	2/1.56
OFFICE	6000	4/1.15	3/0.92	3/0.9	3/0.8	3/1.15	30	5/1.47	9/2.21	9/2.29	7/1.96	5/1.47
PUMPKIN	4000	7/1.69	5/1.30	4/1.1	4/1.0	4/1.34	20	8/1.70	7/1.68	8/1.96	7/1.77	7/1.75
REDKITCHEN	7000	5/1.72	4/1.40	5/1.4	4/1.2	4/1.68	35	7/1.89	7/2.02	10/2.63	8/2.19	6/1.93
STAIRS	2000	4/1.01	5/1.47	4/1.0	3/0.8	3/1.16	20	10/2.21	18/4.8	13/4.24	120/27.37	5/1.47

Table 1. Visual localization results on the 7-Scenes dataset. Left: original training images as database. Right: the few-shot training images as database. For each part of the results, we first list the numnbers of training images in each scene. The feature matching based methods are labeled to orange. The best and the second best results among the few-shot setting are labeled to red and blue, respectively.

Methods	Original training (median pose error in cm ^o)						Few-shot training (median pose error in cm ^o)					
	# Images	AS [36] [†]	HLoc [33,34]	SCRNet [28]	HSCNet [28]	DSAC* [9]	# Images	HLoc [33,34]	DSAC* [9]	HSCNet [28]	SP+Reg	Ours
GREATCOURT	1531	24/0.13	16/0.11	125/0.6	28/0.2	49/0.3	16	72/0.27	NA	NA	NA	81/0.47
KINGSCOLLEGE	1220	13/0.22	12/0.20	21/0.3	18/0.3	15/0.3	13	30/0.38	156/2.09	47/0.74	111/1.77	39/0.69
OLDHOSPITAL	895	20/0.36	15/0.30	21/0.3	19/0.3	21/0.4	9	28/0.42	135/2.21	34/0.41	116/2.55	38/0.54
SHOPFACADE	229	4/0.21	4/0.20	6/0.3	6/0.3	5/0.3	3	27/1.75	NA	22/1.27	NA	19/0.99
STMARYSCHURCH	1487	8/0.25	7/0.21	16/0.5	9/0.3	13/0.4	15	25/0.76	NA	292/8.89	NA	31/1.03

Table 2. Visual localization results on the Cambridge landmarks. NA indicates that the method fails (median translation error larger than 500cm) in the scene. The results of AS[†] come from the paper PixLoc [35]. DSAC* is trained with 3D models. HLoc uses SuperPoint for keypoint detection & description, and SuperGlue [34] for feature matching.

to $h = 256$ for 7-Scenes, and $h = 512$ for Cambridge. All the experiments are run on NVIDIA GeForce RTX 2080 Ti GPU and AMD Ryzen Threadripper 2950x CPU.

4.3. Comparison

We compare our method with state-of-the-art two-step pose estimation methods. We consider both feature matching based methods (AS [36] and HLoc [33, 34]) and scene coordinate regression based methods (SCRNet [28], HSCNet [28], and DSAC* [9]) as our competitors. We also set up a baseline, named SP+Reg, that equips SuperPoint [13] as a feature extractor while employing the regression network instead of classification for the coordinate estimation.

Quantitative results. The quantitative results are shown in Table 1 and Table 2. For each dataset, we list the localization results trained by the original training set in the left part, as the reference to compare with results in the few-shot setting. We first compare the results between the original and the few-shot training. All of the methods suffer an obvious performance drop as the few-shot setting is challenging. The results of scene coordinate regression based methods on Cambridge landmarks are overall worse than the feature matching based ones, as the ground truth depth images used for training are rendered from 3D reconstruction and not as accurate as 7-Scenes measured depth images.

When compared with the scene coordinate regression based methods in the few-shot setting, our method achieves

the best performance on both the 7-Scenes dataset and Cambridge landmarks. Note that for fair comparisons, all SCRNet, HSCNet, SP+Reg, and our method are trained with 9K iterations for 7-Scenes and 30K for Cambridge to ensure the aforementioned methods converge. DSAC* is trained with 100K iterations in the initialization step and 20K iterations in the end-to-end training, and the average training time for each scene is around 2 hours. Besides, the regression-only methods (DSAC* and SP+Reg) fail in several outdoor scenes, the hybrid method (HSCNet) fails fewer, while the classification-only method (Ours) achieves reasonable results across all the scenes.

We also notice that in the few-shot setting, our method overall falls behind the state-of-the-art method HLoc on the Cambridge landmarks. It is reasonable since feature matching based methods do not require the rendered depth for supervision. Nevertheless, our method consumes low memory storage (~ 40 MB) and fast inference time (~ 200 ms). In addition, the main purpose of this paper is to analyze the scene coordinate regression architecture and design a simple and effective scene memorization pipeline to increase the generalisability for novel scenes. Exploring the method combinations of feature matching and coordinate regression in the few-shot setting is worth more research.

Qualitative results. In Figure 5, we visualize the camera poses on the 7-Scenes dataset. We are glad to see that the estimated poses overall overlap their ground truth in the

challenging few-shot setting. However, for the regions which are far from training viewpoints, our method shows large jitters in pose estimation. Following recent works [9, 28], we also render the color images with the estimated poses for intuitive comparisons. The rendered and query images are well aligned, which validates the localization accuracy of our method. Please refer to the supplementary for the visual results on Cambridge landmarks.

4.4. Analysis

Scene coordinate regression. This paragraph provides an analysis of a typical scene coordinate regression network, SCRNet [28], to further explain our inspiration. Our hypothesis is that the first-to-middle layers of SCRNet serve as a feature extractor for robust visual descriptors under different viewpoints while the rest layers memorize the mapping from descriptors to scene-specific coordinates. The features from the former become less effective when the number of training data decreases. To validate this hypothesis, we conduct feature matching experiments using the intermediate feature map of SCRNet. Three examples are shown in Figure 6, where we train the model on CHESS and employ the middle layer’s output feature map as visual descriptors. The top row shows that when trained with full training images it obtains hundreds of correct matches. When the model is trained with only few-shot images, we observe a significant decrease in the correct matches, as shown in the middle row, which indicates that the visual descriptors require amounts of training data. On the bottom row, if the model is applied to extract features in a novel scene STAIRS, the number of correct matches is further decreased. These experiments demonstrate that although the first-to-middle layers of SCRNet have the ability to extract visual descriptors, they require huge training data and hardly be generalized to novel scenes. Consequently, we adopt the decoupled hypothesis of scene coordinate regression and employ an off-the-shelf feature extractor SuperPoint to handle the few-shot setting.

Off-the-shelf feature extractors. We apply SuperPoint [13] as our default feature extractor in Table 1 and Table 2. The results in Table 3 demonstrate that the extractor can also be replaced by other visual descriptors. We apply two state-of-the-art learned features D2-Net [15] and R2D2 [32] as alternatives. Similar to SuperPoint, for the two methods, we take only their semi-dense visual descriptors without detectors. Since these learned features have different dimensions, we slightly change the input channels of the followed classification network to align with different visual descriptors. The results show that the different feature extractors have similar performance in pose estimation.

Scene region partitions. To explore the impact of different scene region partitions, we train and evaluate our model with a different number of region classes. We perform experiments on two selected scenes, REDKITCHEN for indoor

Scene	D2-Net [15]		R2D2 [32]	
	t (cm)	r (°)	t (cm)	r (°)
CHESS	4	1.20	5	1.57
FIRE	4	1.46	4	1.55
HEADS	2	1.44	2	1.50
OFFICE	5	1.52	6	1.72
PUMPKIN	7	1.69	6	1.48
REDKITCHEN	6	1.93	7	1.89
STAIRS	7	1.96	7	1.95

Table 3. Impact of different feature extractors. We replace our feature extractor with state-of-the-art alternatives and evaluate these variants on the 7-Scenes dataset. The median errors of estimated camera poses are comparable with the results in Table 1.

# Classes	REDKITCHEN		GREATCOURT	
	t (cm)	r (°)	t (cm)	r (°)
$30 \times 30 = 900$	7	2.09	-	-
$50 \times 50 = 2500$	6	1.92	124	0.68
$80 \times 80 = 6400$	6	1.90	101	0.58
$90 \times 90 = 8100$	6	1.85	99	0.56
$120 \times 120 = 14400$	-	-	72	0.40

Table 4. Impact of different scene region partitions. We report the median errors of estimated camera poses in two scenes from the 7-Scenes dataset and Cambridge landmarks.

and GREATCOURT for outdoor. As shown in Table 4, we observe that more classes result in more accurate camera pose estimation. The results from REDKITCHEN indicate that the effectiveness of the finer region partition is limited since the current regions are sufficient for the pose estimation. The improvements are significant in GREATCOURT. Especially, when the regions increase to 120×120 , the translation error decreases to 72cm, which is more than 40% lower than 124cm. However, the fine-grained region partition suffers from heavy computation costs, and the processing time significantly exceeds our training time with 14400 regions.

Training times. The key feature of our method is the fast memorization ability with few-shot images. As shown in Figure 7, we compare our training time with SCRNet, HSCNet, and several baselines in REDKITCHEN. For fair comparisons, we train our classifier from random initialization without the meta-learning based pre-training. The results show that our method achieves the fastest convergence with only a few minutes of training time. The baseline named Ours w/o SP denotes a randomly initialized feature extractor instead of SuperPoint, and the comparison with it demonstrates the effectiveness of pre-learned visual descriptors. As we expect, SCRNet is the slowest to converge. Applying SuperPoint to SCR (i.e. SP+Reg) significantly accelerates the convergence and makes it approach HSCNet. HSCNet achieves a not-bad convergence speed as it also applies hi-



Figure 5. Visual results on the 7-Scenes dataset. Top: we visualize the camera poses of training images (green dots), test images (blue dots) and our estimates (red dots). Bottom: we select the image with the median pose error in each scene and visualize the accuracy by overlay the query image (left) with a rendered image (right) using the estimated pose and the ground truth 3D model.

erarchical classifications besides coordinate regression. To conclude, these experiments validate the effectiveness of our decoupled design with scene region classification.

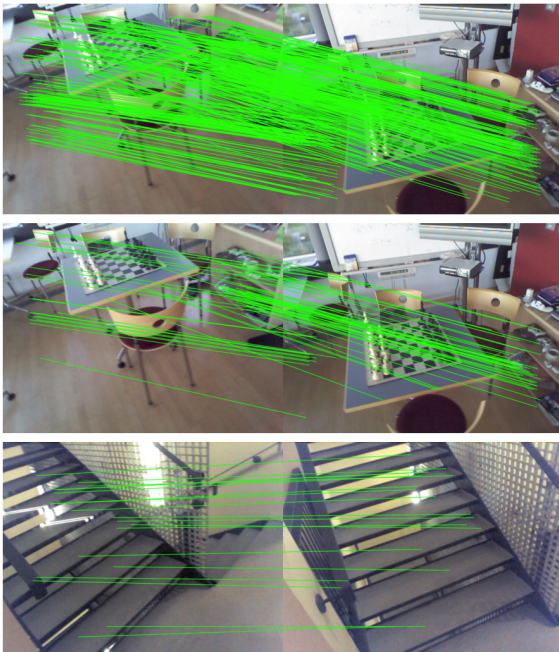


Figure 6. We visualize the correct matches. The correctness is determined by their ground truth coordinates with 5cm tolerance. Top: the model is trained with original training images and tested in the same scene. Middle: the model is trained with few-shot images and tested in the same scene. Bottom: the model is trained with original training images and tested in a novel scene.

Effectiveness of meta-learning. In Figure 8, we report the training curves of our method in REDKITCHEN. We compare three setups: training from random initialization, from pre-trained models by Reptile with $k = 2$ (approximation of MAML [17]) and $k = 5$. We observe that the pre-training accelerates the training convergence, while different types of Reptile gradients do not make much difference.

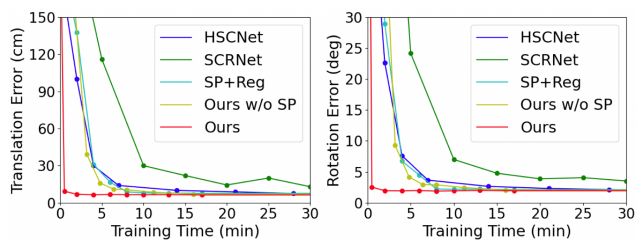


Figure 7. The camera pose median errors vs. training time.

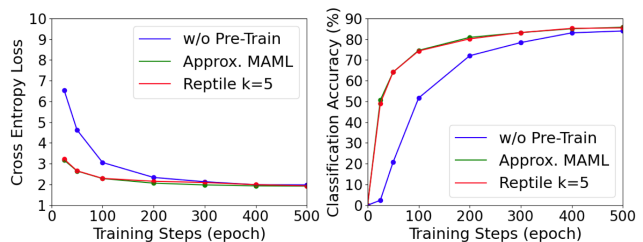


Figure 8. Our training curves with and without meta-learning.

5. Conclusion

In this paper, we propose a novel problem setting that performs visual localization with only few-shot posed images as the database, along with a simple and effective method based on hierarchical scene region classification. Experiments and analysis validate the design of our method, which achieves fast scene memorization with low localization error. Despite the high efficiency, our method suffers from a performance drop due to the few-shot setting. Improving the camera pose accuracy worth more explorations in future works.

Acknowledgements. We thank Songyin Wu, Shanghang Zhang, and Kai Xu for their early discussions and Shaohui Liu for data processing. This work was supported in part by NSFC Projects of International Cooperation and Exchanges (62161146002), NSFC (61902007), Academic of Finland (grant 327911) and Aalto Science-IT project. Siyan Dong was supported by the China Scholarship Council.

Appendix

In this appendix, we report the details of our few-shot version of datasets 7-Scenes [39] and Cambridge [25] in section A, the additional results of SCRNet [28] based feature matching in section B, the qualitative results on Cambridge in section C, the efficiency of our method in section D, and the impact of leaf granularity in section E.

A. Few-Shot Datasets

We perform few-shot experiments on the 7-Scenes dataset and the Cambridge landmarks in the main paper. To avoid bias, the few-shot training set is uniformly sampled from the original training set, as shown in Figure 9.

B. SCRNet Based Feature Matching

The main paper showcases examples that feature maps from SCRNet can match amounts of correct points in the same scene, and match fewer points in another scene. In this section, we perform a cross validation and report quantitative results. As shown in Table 5, we build image pairs by skipping every 30 images, and we count the average number of correct matches. We observe that the test results in another scene achieve consistently less correct matches than those in the same scene. Note that STAIRS is a very challenging scene with repetitive patterns and textureless regions, therefore the numbers of correct matches are overall less than those in CHESS.

Test scene	Training scene	
	CHESS	STAIRS
CHESS	746	564
STAIRS	336	486

Table 5. The number of correct matches. The model is trained separately in CHESS and STAIRS and tested in the same scene and cross scenes.

C. Qualitative Results on Cambridge

In the main paper, we show the qualitative results on the 7-Scenes dataset. The qualitative results on the Cambridge landmarks are shown in Figure 10. We observe that in overall cases, the estimated poses overlap their ground truth.

D. Time Consumption

In this section, we provide detailed time consumption. Our method takes ~ 63 ms for each training iteration (SCRNet ~ 100 ms, and HSCNet [28] ~ 125 ms). The meta-learning based pre-training takes about 5 hours and is once for all. The few-shot memorization converges with about 2 minutes. For inference, our method runs (~ 200 ms) slower

than scene coordinate regression based methods (SCRNet ~ 130 ms) and faster than feature matching based methods (HLoc [33, 34] ~ 500 ms). This is because of the more complexity of the one-to-many PnP-RANSAC mechanism.

E. Leaf Granularity

We report the running time and pose accuracy using different granularities of clustering in leaf nodes shown in Table 6. The choice of granularity is a trade-off between accuracy and efficiency. In general, the accuracy improves when using finer-grained granularities. However, an overly high cluster number, e.g. $q = 50$, may causes more randomness for RANSAC, leading to performance reduction.

# Clusters	$q = 1$	$q = 10$	$q = 30$	$q = 50$
RANSAC time (ms)	20-50	80-230	180-500	400-700
Median error (cm / $^\circ$)	8/2.00	6/1.93	6/1.89	6/1.95

Table 6. Impact of the number of leaf node clusters. Experiments are conducted in the scene REDKITCHEN.

References

- [1] Abien Fred Agarap. Deep learning using rectified linear units (relu). *arXiv preprint arXiv:1803.08375*, 2018.
- [2] Marcin Andrychowicz, Misha Denil, Sergio Gomez, Matthew W Hoffman, David Pfau, Tom Schaul, Brendan Shillingford, and Nando De Freitas. Learning to learn by gradient descent by gradient descent. *Advances in neural information processing systems*, 29, 2016.
- [3] Antreas Antoniou, Amos Storkey, and Harrison Edwards. Data augmentation generative adversarial networks. *arXiv preprint arXiv:1711.04340*, 2017.
- [4] Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef 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.
- [5] Relja Arandjelovic and Andrew Zisserman. All about vlad. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1578–1585, 2013.
- [6] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.
- [7] Eric Brachmann, Alexander Krull, Sebastian Nowozin, Jamie Shotton, Frank Michel, Stefan Gumhold, and Carsten Rother. DSAC-Differentiable RANSAC for camera localization. In *CVPR*, 2017.
- [8] Eric Brachmann and Carsten Rother. Learning less is more - 6D camera localization via 3D surface regression. In *CVPR*, 2018.
- [9] Eric Brachmann and Carsten Rother. Visual camera re-localization from RGB and RGB-D images using DSAC. *TPAMI*, 2021.
- [10] Tommaso Cavallari, Stuart Golodetz, Nicholas A Lord, Julien Valentin, Luigi Di Stefano, and Philip HS Torr. On-the-fly

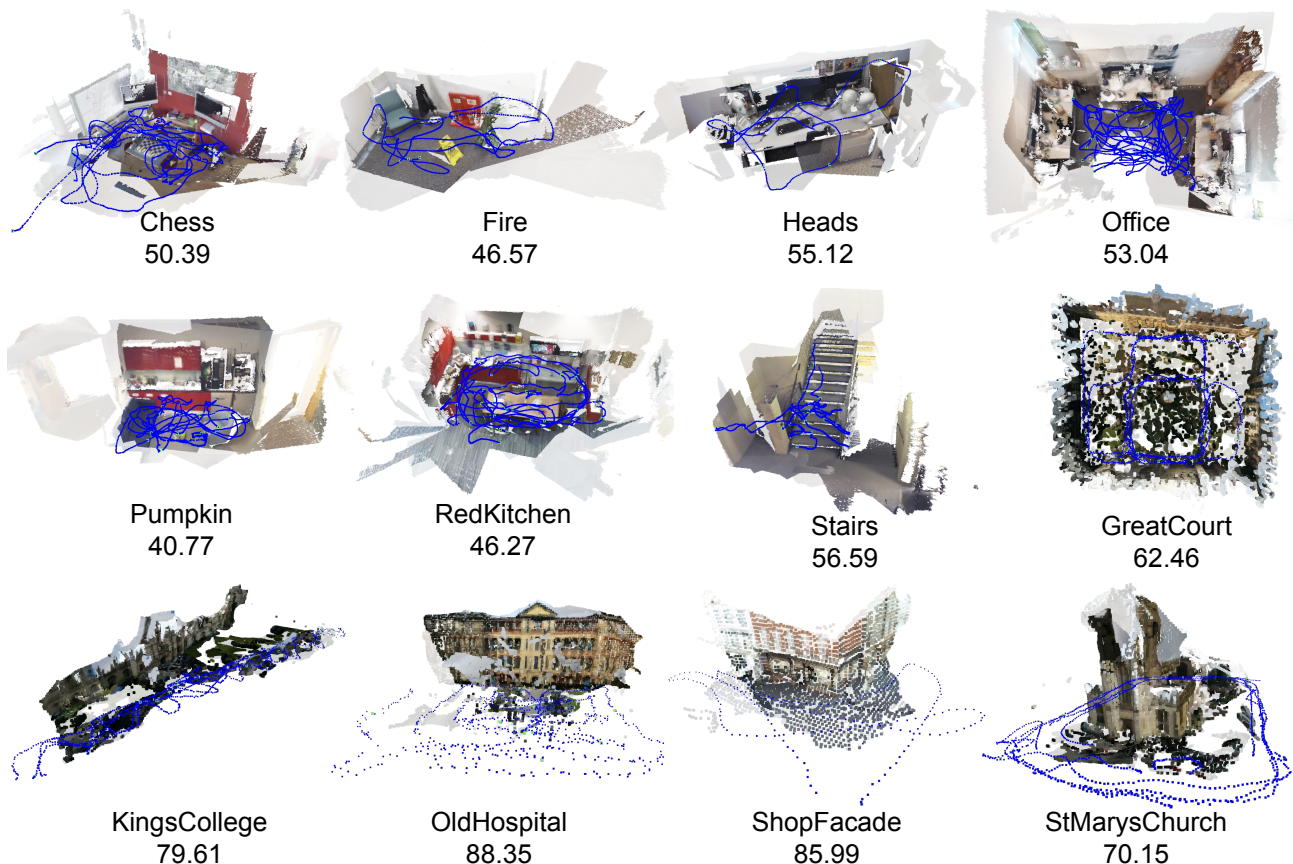


Figure 9. Visualization of the original (blue dots) and the few-shot (green dots) training sets. The reconstruction from the few-shot set is shown in colors while the full scene model is shown as the background. We report the area coverage (%) of the few-shot reconstruction for each scene. The coverage is computed with 10cm and 50cm tolerances for 7-Scenes and Cambridge, respectively.

adaptation of regression forests for online camera relocalisation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4457–4466, 2017.

- [11] Tommaso Cavallari, Stuart Golodetz, Nicholas A Lord, Julien Valentin, Victor A Prisacariu, Luigi Di Stefano, and Philip HS Torr. Real-time rgb-d camera pose estimation in novel scenes using a relocalisation cascade. *IEEE transactions on pattern analysis and machine intelligence*, 42(10):2465–2477, 2019.
- [12] Angela Dai, Matthias Nießner, Michael Zollhöfer, Shahram Izadi, and Christian Theobalt. Bundlerefusion: Real-time globally consistent 3d reconstruction using on-the-fly surface reintegration. *ACM Transactions on Graphics (ToG)*, 36(4):1, 2017.
- [13] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint: Self-supervised interest point detection and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 224–236, 2018.
- [14] Siyan Dong, Qingnan Fan, He Wang, Ji Shi, Li Yi, Thomas Funkhouser, Baoquan Chen, and Leonidas J Guibas. Robust neural routing through space partitions for camera relocalization in dynamic indoor environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8544–8554, 2021.
- [15] Mihai Dusmanu, Ignacio Rocco, Tomas Pajdla, Marc Pollefeys, Josef Sivic, Akihiko Torii, and Torsten Sattler. D2-net: A trainable cnn for joint description and detection of local features. In *Proceedings of the IEEE/cvf conference on computer vision and pattern recognition*, pages 8092–8101, 2019.
- [16] Ethan Eade. Gauss-newton/levenberg-marquardt optimization. *Tech. Rep.*, 2013.
- [17] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR, 2017.
- [18] Martin A Fischler and Robert 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.
- [19] Xiao-Shan Gao, Xiao-Rong Hou, Jianliang Tang, and Hang-Fei Cheng. Complete solution classification for the perspective-three-point problem. *IEEE transactions on pattern analysis and machine intelligence*, 25(8):930–943, 2003.
- [20] Ben Glocker, Jamie Shotton, Antonio Criminisi, and Shahram Izadi. Real-time rgb-d camera relocalization via randomized

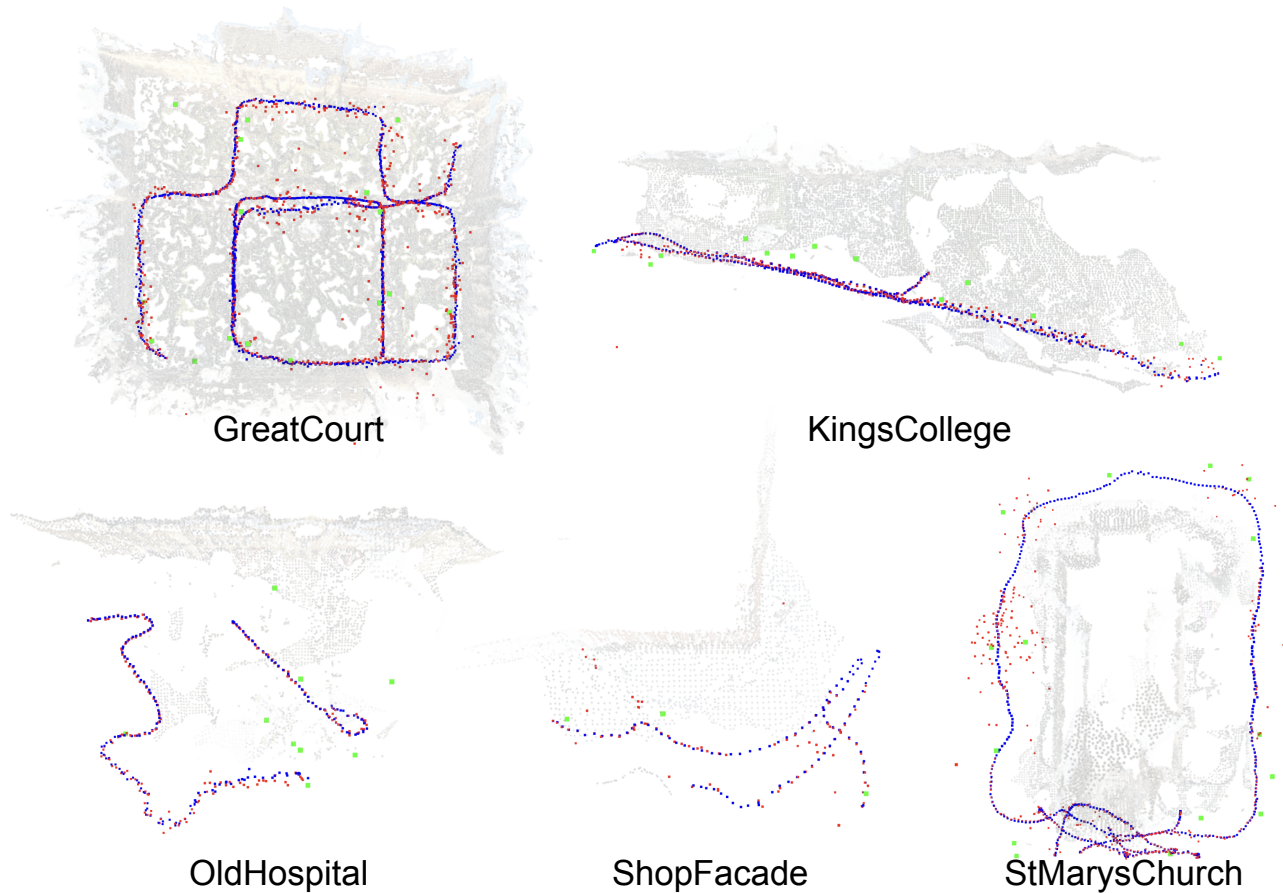


Figure 10. Visual results on the Cambridge landmarks. Training images (green dots), test images (blue dots) and our estimates (red dots).

- ferns for keyframe encoding. *IEEE transactions on visualization and computer graphics*, 21(5):571–583, 2014.
- [21] Bharath Hariharan and Ross Girshick. Low-shot visual recognition by shrinking and hallucinating features. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3018–3027, 2017.
- [22] Zhaoyang Huang, Han Zhou, Yijin Li, Bangbang Yang, Yan Xu, Xiaowei Zhou, Hujun Bao, Guofeng Zhang, and Hongsheng Li. Vs-net: Voting with segmentation for visual localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6101–6111, 2021.
- [23] Tong Ke and Stergios I Roumeliotis. An efficient algebraic solution to the perspective-three-point problem. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7225–7233, 2017.
- [24] Alex Kendall and Roberto Cipolla. Geometric loss functions for camera pose regression with deep learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.
- [25] Alex Kendall, Matthew Grimes, and Roberto Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocation. In *Proceedings of the IEEE international conference on computer vision*, pages 2938–2946, 2015.
- [26] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [27] Gregory Koch, Richard Zemel, Ruslan Salakhutdinov, et al. Siamese neural networks for one-shot image recognition. In *ICML deep learning workshop*, volume 2, page 0. Lille, 2015.
- [28] Xiaotian Li, Shuzhe Wang, Yi Zhao, Jakob Verbeek, and Juho Kannala. Hierarchical scene coordinate classification and regression for visual localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11983–11992, 2020.
- [29] Xiaotian Li, Juha Ylioinas, and Juho Kannala. Full-frame scene coordinate regression for image-based localization. *arXiv preprint arXiv:1802.03237*, 2018.
- [30] Alex Nichol and John Schulman. Reptile: a scalable meta-learning algorithm. *arXiv preprint arXiv:1803.02999*, 2(3):4, 2018.
- [31] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

- [32] Jerome Revaud, Cesar De Souza, Martin Humenberger, and Philippe Weinzaepfel. R2d2: Reliable and repeatable detector and descriptor. *Advances in neural information processing systems*, 32, 2019.
- [33] Paul-Edouard Sarlin, Cesar Cadena, Roland Siegwart, and Marcin Dymczyk. From coarse to fine: Robust hierarchical localization at large scale. In *CVPR*, 2019.
- [34] Paul-Edouard Sarlin, Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperGlue: Learning feature matching with graph neural networks. In *CVPR*, 2020.
- [35] Paul-Edouard Sarlin, Ajaykumar Unagar, Mans Larsson, Hugo Germain, Carl Toft, Viktor Larsson, Marc Pollefeys, Vincent Lepetit, Lars Hammarstrand, Fredrik Kahl, et al. Back to the feature: Learning robust camera localization from pixels to pose. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3247–3257, 2021.
- [36] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Efficient & effective prioritized matching for large-scale image-based localization. *IEEE transactions on pattern analysis and machine intelligence*, 39(9):1744–1756, 2016.
- [37] Torsten Sattler, Tobias Weyand, Bastian Leibe, and Leif Kobbelt. Image retrieval for image-based localization revisited. In *BMVC*, volume 1, page 4, 2012.
- [38] Torsten Sattler, Qunjie Zhou, Marc Pollefeys, and Laura Leal-Taixe. Understanding the limitations of cnn-based absolute camera pose regression. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3302–3312, 2019.
- [39] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2930–2937, 2013.
- [40] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
- [41] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. *Advances in neural information processing systems*, 30, 2017.
- [42] Jiaming Sun, Zehong Shen, Yuang Wang, Hujun Bao, and Xiaowei Zhou. Loftr: Detector-free local feature matching with transformers. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8922–8931, 2021.
- [43] Shitao Tang, Chengzhou Tang, Rui Huang, Siyu Zhu, and Ping Tan. Learning camera localization via dense scene matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1831–1841, 2021.
- [44] Akihiko Torii, Relja Arandjelovic, Josef Sivic, Masatoshi Okutomi, and Tomas Pajdla. 24/7 place recognition by view synthesis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1808–1817, 2015.
- [45] Mikaela Angelina Uy and Gim Hee Lee. Pointnetvlad: Deep point cloud based retrieval for large-scale place recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4470–4479, 2018.
- [46] Julien Valentin, Angela Dai, Matthias Nießner, Pushmeet Kohli, Philip Torr, Shahram Izadi, and Cem Keskin. Learning to navigate the energy landscape. In *2016 Fourth International Conference on 3D Vision (3DV)*, pages 323–332. IEEE, 2016.
- [47] Julien Valentin, Matthias Nießner, Jamie Shotton, Andrew Fitzgibbon, Shahram Izadi, and Philip HS 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.
- [48] Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. *Advances in neural information processing systems*, 29, 2016.
- [49] Florian Walch, Caner Hazirbas, Laura Leal-Taixe, Torsten Sattler, Sebastian Hilsenbeck, and Daniel Cremers. Image-based localization using lstms for structured feature correlation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 627–637, 2017.
- [50] Bing Wang, Changhao Chen, Chris Xiaoxuan Lu, Peijun Zhao, Niki Trigoni, and Andrew Markham. Atloc: Attention guided camera localization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 10393–10401, 2020.
- [51] Shuzhe Wang, Zakaria Laskar, Iaroslav Melekhov, Xiaotian Li, and Juho Kannala. Continual learning for image-based camera localization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3252–3262, 2021.
- [52] Yu-Xiong Wang, Ross Girshick, Martial Hebert, and Bharath Hariharan. Low-shot learning from imaginary data. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7278–7286, 2018.
- [53] Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018.
- [54] Luwei Yang, Ziqian Bai, Chengzhou Tang, Honghua Li, Yasutaka Furukawa, and Ping Tan. Sanet: Scene agnostic network for camera localization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 42–51, 2019.