Decoupling Features and Coordinates for Few-shot RGB Relocalization

Siyan Dong
Shandong University

Songyin Wu
Peking University

Yixin Zhuang
Peking University

Shanghang Zhang
Carnegie Mellon University

Kai Xu
National University of Defense Technology

Baoquan Chen
Peking University

Abstract

Cross-scene model adaption is a crucial feature for camera relocalization applied in real scenarios. It is preferable that a pre-learned model can be quickly deployed in a novel scene with as little training as possible. The existing state-of-the-art approaches, however, can hardly support few-shot scene adaption due to the entangling of image feature extraction and 3D coordinate regression, which requires a large-scale of training data. To address this issue, inspired by how humans relocalize, we approach camera relocalization with a decoupled solution where feature extraction, coordinate regression and pose estimation are performed separately. Our key insight is that robust and discriminative image features used for coordinate regression should be learned by removing the distracting factor of camera views, because coordinates in the world reference frame are obviously independent of local views. In particular, we employ a deep neural network to learn view-factorized pixel-wise features using several training scenes. Given a new scene, we train a view-dependent per-pixel 3D coordinate regressor while keeping the feature extractor fixed. Such a decoupled design allows us to adapt the entire model to novel scene and achieve accurate camera pose estimation with only few-shot training samples and two orders of magnitude less training time than the state-of-the-arts.

1. Introduction

Image-based relocalization addresses the problem of estimating the 6D camera pose of an image captured in a known environment. It is the crux of many applications such as robot navigation, simultaneous localization and mapping (SLAM), and augmented reality. There are two basic types of approaches: Feature matching based and coordinate regression based. While the former matches entire images/landmarks to the 3D scene model, the latter directly maps image pixels to 3D coordinates. Camera pose is computed based on the resulting 3D coordinates. Without the reliance on a 3D scene model, coordinate regression is practically more versatile in real scenarios.

Random forest is perhaps the most popular model for coordinate regression [27]. In such models, pixel-wise feature extraction and coordinate regression are performed in an integrated fashion. Recently, deep neural networks have been proposed to directly regress camera pose from an input image [17]. Since direct pose prediction is an extremely hard task, such end-to-end approaches still cannot outperform random forest models. Common to these integrated or end-to-end models, however, is that they all require large
amount of training data while suffering from poor model generality across different scenes.

Cross-scene model adaption is an important requirement for camera relocalization in nowadays [5, 4]. It is preferable that a pre-learned model can be quickly deployed in a novel scene with as little training as possible. The existing approaches, however, can hardly support few-shot scene adaption due to the entangling of image feature extraction and 3D coordinate regression, which requires a lot of training data. This greatly limits their practical utility as it is expensive to gather a plenty of training images with pose labels for a novel scene.

To adapt to few-shot scenes, inspired by how humans perform relocalization, we propose to approach camera relocalization in a decoupled way. In particular, we advocate that feature extraction and coordinate regression ought to be performed separately, to best support few-shot learning of powerful model of image-based relocalization. Our key insight behind is that robust and discriminative images features used for coordinate regression should be learned by removing the distracting factor of camera views, because coordinates in the world reference frame are apparently independent on local views. This motivates us to learn a model for view-factorized feature extraction for each image pixel, and use the extracted features with another model trained only for coordinate regression.

In our method, since the view-independent features are extracted with a deep neural network pre-trained using images captured in several known scenes, given a new scene, only the view-dependent part, i.e. the coordinate regressor, needs to be trained with very few images captured in the scene. The decoupled design enables few-shot training of the whole model while achieving high pose accuracy. Through delegating feature extraction to a separate model, the coordinate regressor can be realized with a light-weight decision tree, replacing the commonly used random forests. This further reduces the required amount of training. Such mechanism releases the model’s demand of large training data when adapted to new scenes, and is also inline with humans relocalization process, which first learns visual concepts from rich experience and then estimate pose in new scenes with only a few landmarks. To accomplish relocalization with only RGB images, we propose a Perspective-n-Point (PnP) [14] based Preemptive RANSAC, named P3SAC. Based on the pixel-wise 3D coordinates predicted by the decision tree, we iteratively solve the PnP inside a preemptive RANSAC loop. We conducted extensive quantitative evaluation and demonstrate we achieve highest accuracy with the same amount of training data. In summary, our work makes the following contributions.

- We propose the first few-shot image-based relocalization through decoupling feature extraction, coordinate regression and pose optimization.
- We train a dedicated convolutional neural network with a self-supervised triplet loss for learning view-factorized pixel feature embedding.
- To achieve RGB-based relocalization, we design a PnP-based preemptive RANSAC for 6D camera pose estimation from the 3D coordinate regression.

2. Related Work

Feature matching based approaches. Early works on image-based relocalization usually perform full image matching. Based on holistic image descriptions such as bag-of-word features and randomized ferns, relocalization is solved by key frame retrieval and frame-to-frame registration [7, 22, 9, 10]. Another line of research pursues 2D-3D feature point (landmark) matching between the input image and a 3D scene model of the scene, based on image pixel or patch level descriptors [32, 34, 26, 19]. Based on these matching, camera poses can then solved with the help of PnP and/or RANSAC.

Coordinate regression based approaches. There are an excellent series of works on regressing 3D coordinates of image pixels using learned random forests [27, 29, 5, 4], reaching the state-of-the-art performance. Brachmann et al. [3] propose DSAC, a differentiable RANSAC for camera relocalization. They train a deep neural network that takes image patches as input and regresses the 3D coordinates of their center pixels. Li et al. [20] propose a dedicated deep network to perform pixel-wise coordinate regression.

End-to-end approaches. Empowered by the feature learning capacity of deep networks, many works have been attempting end-to-end camera pose prediction from an image. PoseNet [17] is a typical work of this type, which attains promising results on challenging cases such as motion blur and illumination change. Several improvement have been proposed through modeling uncertainty [15], exploiting structured feature correlation [31] and designing geometric loss functions [16]. However, the localization accuracy of these methods is still far below traditional approaches.

Scene adaptation. It is shown in [5] and [4] that pre-trained forests in one scene can be adapted to another without changing the structure of the decision trees. Such adaption can obtain close to state-of-the-art performance. [4] further shows that forests with randomly generated parameters (no pre-training) can also be adapted to novel scenes with comparable performance. Due to the entangling between feature learning and coordinate regression, such models can be cannot achieve few-shot generalization: The adapted model still needs heavy fine-tuning with a large mount of training data from the new scene.

Few-shot learning. Few-shot learning (FSL) is the recognition paradigm with only a few training samples. The data-hallucination based methods devise generators to
transfer the data distributions [13] or visual styles [2] to augment the novel examples. These methods induce domain shift between the generated data and the data of the few-shot classes. The metric learning based methods learn the feature representations and compare them so that samples of the same class show higher similarity than those of different classes, where the similarity can be evaluated with cosine similarity [30], Euclidean distance [28], and graph neural networks [12, 8]. The meta-learning based methods aim to learn the feature representations and compare them so that samples of the same class show higher similarity than those of different classes, where the similarity can be evaluated with cosine similarity [30], Euclidean distance [28], and graph neural networks [12, 8]. The meta-learning based methods aim to learn a good model initialization [6] or an optimizer [23]. However, both of these approaches suffer from the need to fine-tune on the target problem. To the best of our knowledge, there is no existing work on few-shot RGB relocalization.

### 3. Method

In this section, we describle our few-shot visual relocalization framework, which contains three major components, a view-invariant feature descriptor, a scene-dependent Decision Tree, and a RANSAC-based pose predictor. By decoupling the 2D-3D regression task, we are able to perform faithful coordinate regression with only few images using a pre-learned scene prior. Figure 2 shows the outline of our framework. In next few sections, we will introduce our scene independent feature descriptor, and by converting pixels to more general representation, we show how to build local scene coordinate regressor efficiently. Finally, we introduce PnP-based Preemptive RANSAC for online pose estimation.

#### 3.1. Scene Prior Learning

Our scene prior is a view-invariant feature descriptor, which is independent to any specific scene coordinate system. In general, our feature descriptor is formulated as a neural network, and trained on images from scenes with different scales and viewpoints. By the mean of view-invariant, the descriptor is expected to know whether pixels from different views are the same or not according to the same scene position, judged by global 3D coordinate. We learn the view priors with a bunch of positive and negative pairs between images across several scenes, and train the network in a siamese fashion.

Specifically, we follows a similar design as the work of [20], by making use of U-Net framework. The conventional U-Net consists of a contractive part and an expanding part, performs pixel-wise prediction [24]. Our network generates feature tensor in the last layer, with each pixel represented by a n-dimensional feature vector respectively. Our siamese network is conducted with one way pushing positive pair of pixels closer and the other way moving negative pair distant in the embeded space.

In order to achieve such metric learning, we use triplet loss for training. Let \( F(x) \) denote the feature at location \( x = (x, y) \) of image \( I \). The loss function is defined on the feature vectors from images \( I \) and \( \hat{I} \), at indices \( x \) and \( \hat{x} \).
respectively. If \( x \) and \( \hat{x} \) correspond to the same coordinate, we use them as a positive pair, otherwise as a negative pair. The full loss is formulated as

\[
L_{\text{triplet}} = \sum_{i=1}^{k} \max(\|F_I(x^a_i) - F_I(\hat{x}^p_i)\|_2 - \|F_I(x^a_i) - F_I(\hat{x}^n_i)\|_2 + \text{margin}, 0),
\]

where \( x^a_i \) is the anchor and \( \hat{x}^p_i, \hat{x}^n_i \) denotes the positive and negative pair respectively.

We use five hundred correspondences from each valid image pair. Due to the noisy and sparsity of the scene model, we relax the requirement for two pixels with corresponding coordinates close within a threshold as positive sample. We use the publicly available '7-Scenes' dataset from Microsoft as training data, and optimize the parameters of \( F \) with Adam optimizer.

We train our network in a self-supervised fashion similar to [33]. To train our network, we first reconstruct each scene from training sequences. The training triplets are extracted online from the scene reconstructions. Each pair of training images is from the same scene. For each pair of the images, we first re-project pixels into scene reconstruction to detect correspondences. If there are more enough correspondences, the pair of images is used as training data. If not, there are not enough overlap, we discard the pair. Then the remain pairs are used as anchor and positive, we randomly select another pixel as negative, to form the triplet. Our method of generating training triplets is automatic without human supervision or manual labor.

Figure 3 shows the t-SNE visualization of embedding space of visual features. Each cluster of points visualized in the same color corresponds to the same coordinate. The distance among visualized points represents the similarity of features. Since the feature is insensitive to the change of view, coordinate regression by given feature as input would only requires few images.

### 3.2. Few-shot Scene Adaptation

Inference scene coordinate from few-shot observation is very challenge, due to the lackness of ‘depth’ and dense view. However, our human can easily understand the structure of scene with just a glance. This is because one has seen a lot of scenes, and stores the knowledge of scene from many aspects in one’s brain. The View factor seems the most important one to understand the structure of environment and the position of the observer. Hence, our view-invariant feature descriptor brings huge possibility on few-shot scene relocalization.

Given few images of a new scene, we construct a feature-based scene coordinate regressor with the pre-learned scene prior. Our local scene representation is a Decision Tree with each intermediate nodes as a binary classifier, and the leaf node as a cluster containing multiple scene coordinates. Since our feature is robust to the disturbance of view change, our few-shot Decision Tree is able to retrieve accurate coordinates from very different view angles. Beside factoring out view influence, the key to find the right coordinate is that our Decision Tree allows soft decision by mapping one feature to many possible coordinates. Unlike one-to-one mapping, Decision Tree survives in much difficult situations, such as motion blur, reflecting surface and repeating structures. Meanwhile, different features lying outside the distribution of training set may retrieve empty leaf node, and are discarded.

The input of Decision Tree are \( n \)-dimensional vectors. The basic idea is to partition the input space to regions. It consists of multi-layers with intermediate nodes as classifier that subdivides the coming data into two sets by a learned threshold, using mean shift clustering. As a result, Decision Tree maps input features to multiple classes. We use 5-10 images to train the classifiers. Within each leaf node, coordinates contains the 3D position, color and normal informations. The geometry, color and texture give strong indica-
tion of the confidence on coordinate retrieving. Our scene regressor provides rich information from various aspects of scene. The simplicity of our few-shot scene adaptation is very practical in terms of feasibility and scalability.

3.3. Online Pose Estimation

For a new captured image, we estimate the corresponding coordinates for each pixel, with the scene priors and few-shot scene representation. We use Perspective-n-Points (PnP) to calculate the pose using $n(\geq 4)$ pixel-to-coordinate pairs. As shown in Figure 2, pixel from input image is first converted to feature vector and then passed to Decision Tree, with each pixel mapping to a cluster containing multiple coordinates. We modify the standard RANSAC for pixel-cluster pair selection. Although pixel-cluster mapping increases the computational complexity comparing to one-to-one mapping, it improves the result of coordinate retrieving.

We follow a voting scheme as described in [27], where the problem of camera pose estimation is achieved by maximizing the total score over all putative camera poses with each represented by a matrix $H$. The target is to find as many inliers as possible while minimize the re-projection error of these inliers. We define the scoring function as

$$H* = \arg \max_H S(H),$$

with each hypothesis evaluated by

$$S(H) = \frac{|\text{Inliers}|}{\max (\sum_{(x_i, m) \in \text{Inliers}}\|x_i - PHm\|_2, \alpha)},$$

where $x_i$ is the 2D pixel in the image, $m \in \text{Cluster}$ is the scene coordinate in the cluster and $P$ is the camera projection matrix. $(x_i, m)$, the inlier, is selected among all 2D-3D pairs within a cluster, minimizes the re-projection error.

$$\arg \min_{m \in \text{Cluster}} \|x_i - PHm\|_2.$$  

In our experiments, if the error exceeds a pre-defined threshold $\tau$, we discard the corresponding candidate. We set $\tau = 20$ in our experiment.

Our RANSAC-based pixel-cluster selection takes into consideration of all coordinates within a cluster, and iteratively finds a best candidate with smallest re-projection error within cluster. Note that, given different poses, the best candidate should be different for most of the time. Our optimization steps is detailed in Algorithm 1. By following the setting in [3, 20], the initial number of hypotheses, $K_{\text{hypo}}$, is set to 256. We sampled 1600 validating pixels during each iteration. Note that, for each $i \in V$, only one pair with the lowest re-projection error is counted as inlier. We refine the pose via PnP over currently selected inliers. Hyper-parameter $\tau$ is used to control the movement of pose, and is decreased during iteration. Also, $\beta \in (0, 1)$ is used to represent the confidence of the score. Specifically, when $\tau$ is smaller, the confidence is higher.

Algorithm 1 Pseudocode for preemptive RANSAC

1: $K \leftarrow K_{\text{hypo}}$
2: extract feature map $F'_z$ using $F$ from image $I$
3: generate initial hypotheses $H_1, ..., H_K$
4: initialize scores $S_k \leftarrow 0$ for $k = 1, ..., K$
5: while $K > 1$ do
6: sample validating pixels $V$ from image $I$
7: for all $i \in V$ do
8: $M_i = \text{DecisionTree}(F'_z(x_i))$
9: end for
10: for all $k = 1, ..., K$ do
11: $S_k \leftarrow \beta \cdot S_k + S(H_k, V, M)$
12: end for
13: sort hypotheses $(H_k, S_k)$ by $S_k$
14: $K \leftarrow \lfloor K \rfloor$
15: refine hypotheses $H_k$ for $k = 1, ..., K$
16: end while
17: return $H$

4. Experiments

In this section, we extensively evaluate our proposed methods from different aspects, including ablation study to verify each proposed module, pose estimation accuracy, computational cost, and robustness in the difficult situations. Due to space limitation, please refer to the supplementary for more details of the experiments.

4.1. Datasets and Settings

We compare with state-of-the-art methods on 7-Scenes datasets [27]. This dataset was designed for RGBD relocalization, and contains significant variation in camera view. As it contains many ambiguous repeated patterns, texture-less features, motion blur and reflections, it is challenging for pure RGB relocalization. We take each of the seven scenes as an unknown scene in turn, and use the other six scenes as known scenes, forming seven sets of experiments. For each set of experiments, our proposed pipeline consists of three steps, i.e. offline pre-training, on-the-spot scene regressor training, and online pose estimation. In the offline step, with our pre-calculated positive and negative pairs from raw training data [27], we first pre-train the feature extraction network on six scenes. Then, We select 5 to 10 images from the other scene to perform on-the-spot scene coordination regression. Due to the efficiency of our methods, this step takes only a few seconds. With pre-trained feature extractor and on-the-spot trained scene regressor, we then can perform online pose estimation from either images or video.
4.2. Implementation Details

We make use of U-Net with totally 28 layers as feature extractor. We apply ELU as activate function between layers, and sigmoid for the last layer, to fit the follow-up binary tree structure. For all experiments, we set the output features dimension as $n = 16$. The height of our tree regressor is equal to the length of feature vector. In P3SAC, we use 256 dedicated threads for initial hypothesis generation. Each thread attempts to generate a single validate hypothesis by randomly sampled 4 2D-3D correspondences retrieved using our tree regressor. If the sampled pose induce less than 4 inliers, or the projected pixel is deviation from ground truth by a threshold, the hypothesis is rejected, and the thread starts a new attempt. During scoring and refinement iteration, $\tau$ is decreased by multiply 0.9 until smaller than 10, and $\beta$ is set to 0.5.

| Scene  | ORB  | DSAC | Full-Frame | Full-Frame w/ MAML | Ours  |
|--------|------|------|------------|--------------------|-------|
| Chess  | 1.68m| 0.28m| 0.46m      | 1.21m              | 0.14m |
| Fire   | 1.73m| 0.67m| 0.63m      | 1.58m              | 0.23m |
| Heads  | 0.96m| 0.05m| 0.61m      | 0.97m              | 0.23m |
| Office | 1.75m| 1.63m| 2.03m      | 1.81m              | 0.38m |
| Pumpkin| 2.85m| 0.51m| 2.06m      | 2.45m              | 0.35m |
| Kitchen| 2.79m| 2.68m| 1.12m      | 1.69m              | 0.25m |
| Stairs | 2.34m| 2.29m| 1.52m      | 1.22m              | 0.31m |
| Average| 2.01m| 1.16m| 1.20m      | 1.56m              | 0.27m |

Table 1. Median translation error of estimated camera pose.

4.3. Comparison

In this section, we compare our approach (Ours) with state-of-the-art camera relocalization methods. The first baseline (ORB) represents the state-of-the-art conventional bag-of-words model, which takes few-shot images as keyframes. The second baseline (DSAC) and the third baseline (Full-Frame) represent the state-of-the-art deep learning based method, one uses image patches as input, while the other inputs full images. The fourth baseline (Full-Frame w/ MAML) represents a combination of state-of-the-art deep learning based camera relocalization with few-shot learning approach.

ORB [25] is a state-of-the-art non-learning based baseline method. ORB feature is designed to retrieve 2D-3D correspondences for camera relocalization. Few-shot images of the test scene are regarded as keyframes, and used to extract ORB features and build feature-3D correspondences. Then PnP is utilized to estimate the camera pose.

DSAC [3] is a representative work of state-of-the-art RGB patch-based method. The model in each experiment set is trained on few-shot images from the unknown scene for 50k updates to get the training loss converge.

Full-Frame [20] is a representative work of state-of-the-art RGB full-frame based method. The model is trained on few-shot images from the unknown scene for 6k updates to get the training loss converge. We combine MAML [6], a representative meta-learning method for few-shot learning, with the Full-Frame [20]. As described in MAML, the network is pre-trained on the 6 known scenes, an then fine-tuned on the few-shot frames from the unknown scene with the meta-learning strategy.

| Scene  | ORB  | DSAC | Full-Frame | Full-Frame w/ MAML | Ours  |
|--------|------|------|------------|--------------------|-------|
| Chess  | ≫10  | 1.74 | 4.46       | 0.92               | 0.13  |
| Fire   | ≫10  | 1.35 | ≫10        | 1.00               | 0.99  |
| Heads  | ≫10  | 0.67 | 0.58       | 0.52               | 0.78  |
| Office | ≫10  | 2.51 | 1.11       | 1.01               | 1.63  |
| Pumpkin| ≫10  | 1.92 | ≫10        | 1.06               | 0.52  |
| Kitchen| ≫10  | 3.19 | 1.11       | 0.91               | 1.70  |
| Stairs | ≫10  | 2.25 | 1.12       | ≫10                | 1.06  |
| Average| ≫10  | 1.95 | ≫10        | ≫10                | 0.97  |

Table 2. Standard deviation of camera pose translation error for different methods.

Accuracy. To verify the effectiveness of the proposed methods, we compare the camera pose estimation accuracy in Table 1. We also provide the comparison of output images rendered by re-projecting scene coordinates from estimated camera poses, as shown in Figure 5. From the results we can see that, our method outperforms all the baseline methods in most scenes.

Because of the sparsity of keyframes, ORB almost totally failed in the few-shot setting, which verifies that Handcrafted features are sensitive to viewpoint changes. Thus ORB cannot correctly match 2D pixels between test images and keyframes, leading to the wrong retrieval of 3D scene coordinates. For DSAC, even though the training images are only a few, it crops lots of patches as training data. However, overfitting still occurs due to the lack of training data. For the unknown scene Head whose test images are close to the training images, DSAC can achieve accurate relocalization with low error. But in general, DSAC performs bad generalization in most unknown scenes that are different from the known scenes. Figure 4 further visualizes the overfitting of DSAC.

Full-Frame also suffers from overfitting due to the lack of training data and obtains poor results in most scenes. For Full-Frame w/ MAML, although the network is pre-trained on large amount of images from various known scenes, it still fails to adapt to new scene with only few-shot images when features and coordinates are coupled. We also fine-tune more than 1 step for MAML, while the results remains similar. It is difficult for MAML to converge due to the feature-coordinate coupling, because the feature-coordinate training may conflict under different coordinate systems. Benifit from the decoupling of feature learning and coordinate regression, we achieve higher accuracy than the state-of-the-arts with two orders of magnitude less training time.
Since the training of Decision Tree based coordinate regressor only need few-shot samples in specific scene under specific coordinate system, our model successfully avoids overfitting and generalizes to unknown scene.

| Training Time | ORB | DSAC | Full-Frame | Full-Frame w/ MAML | Ours |
|---------------|-----|------|------------|--------------------|------|
| 1s            | 30min | 3.5h | 100ms     | 15s                |

Table 3. Comparisons of few-shot training time.

**Computation Efficiency.** Camera relocalization is in high demand of training efficiency, including both the number of training samples and the training time. However, most learning based methods need long training time to converge, even when the training data is only a few. To verify the computation efficiency of our method, we evaluate the training time as shown in Table 3. For ORB, the number represents the time of extracting ORB features, computing 3D coordinates, and build the bag-of-words dictionary. For DSAC, Full-Frame, and Full-Frame w/ MAML, and our method, the number represents the network training time on few-shot samples of unknown scenes. Note that for Full-Frame w/ MAML, we only finetune for 1 step, to be consistent with [6]. From Table 3, we see that our method only needs second-level training time to build the Decision Tree and achieve accurate pose estimation, which is two orders of magnitude less than DSAC and Full-Frame. Since ORB only needs to extract features and compute 3D coordinates without training, it only costs 1s for this process.

**Robustness.** To verify the robustness and stability of our model, we compare the standard deviation of camera pose translation error as shown in Table 2. From the results we can see that our method is quite stable and robust with low standard deviation, while all the other baselines suffer from large standard deviation. The high standard deviation of Full-Frame w/ MAML further demonstrates that our decoupling mechanism is effective and robust in the few-shot camera relocalization setting.

### 4.4. Ablation Study

To justify the efficacy of each component in our approach, we conduct ablation study to compare our full solution (Full) against the following ablated variants.

**Ablate learned scene prior (w/o Scene Prior)** Instead of pre-training our model on different known scenes, we apply a U-Net with randomly initialized parameters, to ablate the prior feature learning from our full solution.

**Ablate decision tree regression (w/o Decision Tree)** Instead of learning a Decision Tree on few-shot frames, we train a pixel-wise network to learn the mapping from features to coordinates. The network contains 5 1x1 convolution layers, with about 60k parameters, to be consistent with our Decision Tree.

**Ablate preemptive RANSAC (w/o P3SAC)** Instead of P3SAC, we apply a conventional RANSAC, as described in [20]. We initialize 256 camera pose hypotheses, select the one with the most inliers, and then refine remained hypothesis at most 8 times until meeting the terminate condition.

| Scene      | w/o Scene Prior | w/o Decision Tree | w/o P3SAC | Full          |
|------------|----------------|-------------------|-----------|---------------|
| Chess      | 0.28m          | 4.68m             | 0.61m     | 0.14m         |
| Fire       | 0.97m          | 4.96m             | 0.62m     | 0.23m         |
| Heads      | 0.29m          | 2.34m             | 0.34m     | 0.23m         |
| Office     | 3.25m          | 3.75m             | 1.31m     | 0.38m         |
| Pumpkin    | 0.61m          | 6.36m             | 0.60m     | 0.35m         |
| Kitchen    | 0.44m          | 7.30m             | 0.94m     | 0.25m         |
| Stairs     | 1.21m          | 10.32m            | 1.61m     | 0.31m         |
| Average    | 1.01m          | 5.67m             | 0.86m     | 0.27m         |

Table 4. Median translation error from variants of our model.

Table 4 shows that the our full solution consistently outperforms the variants of our model with ablated components, verifying the efficacy of each component. Here we provide detailed analysis as below.

**Scene prior is important.** To ablate the scene prior feature learning, we apply a network with randomly initialized parameters. As shown in Table 4, random initialized network can generalize to some scenes, such as Chess and Heads, because these scenes have small scanning view changes and are easy to be adapted to. However, when comes to scenes with large change of scanning view, like Office, the random initialized network fails. It shows that our learned prior is helpful to extract view invariant features, which leads to more accurate camera pose estimation.

**Decision Tree is more suitable for coordinate regression.** The mapping from feature space to coordinate space is crucial, for it predicts the 2D-3D correspondences, which is the keypoint for camera pose estimation. Yet it is often challenged by either overfitting or difficulty to converge. Since there are often similar patterns in the scenes, like Stairs, similar feature may appear in different coordinates. Under this circumstance, Networks trend to converge to a central coordinate among training samples. However, Decision Tree can provide one-to-many correspondences, and achieves diverse and accurate correspondence predictions. It is also tolerant to feature noise, which makes coordinate retrieval more robust to output the correct correspondence.

**Preemptive RANSAC is necessary.** When a camera (viewpoint) pose hypothesis is generated, humans tend to find more inlier landmarks to prove or deny it, which is the key insight of RANSAC. Because of the one-to-many correspondence predictions, RANSAC generates lots of false positive camera pose hypotheses. When correspondence samples are sparse (because of few-shot), it is hard to find the most accurate camera pose if only validated once. The comparison to w/o P3SAC shows that P3SAC is a well tailored version for our one-to-many correspondences setting.
Figure 4. Comparisons of overfitting and generalization. To show the generalizing capability of our learned model, we select 100-frame from 3 scenes. The 50th image is used as training image, while others are testing images. The plot shows the translation error of estimated camera pose from these 100 frames. On 'Chess' dataset, our method performs robust pose estimation with consistent low error, while DSAC fails around the 0 to 20th frames. On 'Heads' dataset, which contains tiny view change, the two method achieve similar performance. Since DSAC overfits the training data, it obtains lower error than ours. On 'Office' dataset, which contains large view change, we get lower error than DSAC most of the time. For each scene, we runs 5 times with different random samples (solid curve shows the average error, and shadow color shows standard deviation).

Figure 5. Comparison of rendered RGB images from estimated pose on 'Pumpkin' from 7-scenes.

5. Conclusions

In this paper, we present a novel framework for few-shot camera relocalization. With seeing only a few images, our method is able to estimate reliable pose from novel view observations. This is achieved by a view-coordinate decoupling design, such that, we can learn a view-invariant scene prior independent to local coordinate systems of specific scenes. We incorporate scene prior to coordinate regression, and optimize the pixel-to-coordinate correspondence.
for pose estimation. Since few-shot acquisition is simple and our scene adaptation is performed on-the-spot, our method is practical and convenient to be applied to novel environments for localization driven applications.

References

[1] Marcin Andrychowicz, Misha Denil, Sergio Gomez, Matthew W Hoffman, David Plaut, Tom Schaul, Brendan Shillingford, and Nando De Freitas. Learning to learn by gradient descent by gradient descent. In NeurIPS, pages 3981–3989, 2016.

[2] Andreas Antoniou, Amos Storkey, and Harrison Edwards. Data augmentation generative adversarial networks. arXiv preprint arXiv:1711.04340, 2017.

[3] Eric Brachmann, Alexander Krull, Sebastian Nowozin, Jamie Shotton, Frank Michel, Stefan Gumbold, and Carsten Rother. Dsac-differentiable ransac for camera localization. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 6684–6692, 2017.

[4] Tommaso Cavallari, Stuart Golodetz, Nicholas Lord, Julien Valentin, Victor 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, 2019.

[5] Tommaso Cavallari, Stuart Golodetz, Nicholas A Lord, Julien Valentin, Luigi Di Stefano, and Philip HS 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.

[6] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In Proceedings of the 34th International Conference on Machine Learning-Volume 70, pages 1126–1135. JMLR. org, 2017.

[7] Dorian Galvez-Lopez and Juan D Tardos. Real-time loop detection with bags of binary words. In 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, pages 51–58. IEEE, 2011.

[8] Victor Garcia and Joan Bruna. Few-shot learning with graph neural networks. arXiv preprint arXiv:1711.04043, 2017.

[9] Ben Glocker, Shahram Izadi, Jamie Shotton, and Antonio Criminisi. Real-time rgb-d camera relocalization. In 2013 IEEE International Symposium on Mixed and Augmented Reality (ISMAR), pages 173–179. IEEE, 2013.

[10] Ben Glocker, Jamie Shotton, Antonio Criminisi, and Shahram Izadi. Real-time rgb-d camera relocalization via randomized ferns for keyframe encoding. IEEE transactions on visualization and computer graphics, 21(5):571–583, 2014.

[11] Erin Grant, Chelsea Finn, Sergey Levine, Trevor Darrell, and Thomas Griffiths. Recasting gradient-based meta-learning as hierarchical bayes. arXiv preprint arXiv:1801.08930, 2018.

[12] Michelle Guo, Edward Chou, De-An Huang, Shuran Song, Serena Yeung, and Li Fei-Fei. Neural graph matching networks for fewshot 3d action recognition. In Proceedings of the European Conference on Computer Vision (ECCV), pages 653–669, 2018.

[13] Bharath Hariharan and Ross Girshick. Low-shot visual recognition by shrinking and hallucinating features. In ICCV, pages 3018–3027, 2017.

[14] R Hartley and A Zisserman. Multiple view geometry in computer vision (cambridge university, 2003).

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

[16] 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, pages 5974–5983, 2017.

[17] Alex Kendall, Matthew Grimes, and Roberto 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.

[18] Yoonho Lee and Seungjinn Choi. Gradient-based meta-learning with learned layerwise metric and subspace. arXiv preprint arXiv:1801.05558, 2018.

[19] Shuda Li and Andrew Calway. Rgbd relocalisation using pairwise geometry and concise key point sets. In 2015 IEEE International Conference on Robotics and Automation (ICRA), pages 6374–6379. IEEE, 2015.

[20] Xiaotian Li, Juha Ylioinas, and Juho Kannala. Full-frame scene coordinate regression for image-based localization. arXiv preprint arXiv:1802.03237, 2018.

[21] Tsenduren Munkhdalai and Hong Yu. Meta networks. In ICML, pages 2554–2563. JMLR. org, 2017.

[22] Raül Mur-Artal and Juan D Tardós. Fast relocalisation and loop closing in keyframe-based slam. In 2014 IEEE International Conference on Robotics and Automation (ICRA), pages 846–853. IEEE, 2014.

[23] Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. 2016.

[24] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In International Conference on Medical image computing and computer-assisted intervention, pages 234–241. Springer, 2015.

[25] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary R Bradski. Orb: An efficient alternative to sift or surf. In ICCV, volume 11, page 2. Citeseer, 2011.

[26] 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.

[27] 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.
[28] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In NeurIPS, pages 4077–4087, 2017. 3

[29] 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. 2

[30] Oriol Vinyals, Charles Blundell, Tim Lillicrap, Daan Wierstra, et al. Matching networks for one shot learning. In NeurIPS, 2016. 3

[31] 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. 2

[32] Brian Williams, Georg Klein, and Ian Reid. Automatic relocalization and loop closing for real-time monocular slam. IEEE transactions on pattern analysis and machine intelligence, 33(9):1699–1712, 2011. 2

[33] Andy Zeng, Shuran Song, Matthias Nießner, Matthew Fisher, Jianxiong Xiao, and Thomas Funkhouser. 3dmatch: Learning local geometric descriptors from rgb-d reconstructions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 1802–1811, 2017. 4

[34] Guofeng Zhang, Haomin Liu, Zilong Dong, Jiaya Jia, Tien-Tsin Wong, and Hujun Bao. Efficient non-consecutive feature tracking for robust structure-from-motion. IEEE Transactions on Image Processing, 25(12):5957–5970, 2016. 2

[35] Ruixiang Zhang, Tong Che, Zoubin Ghahramani, Yoshua Bengio, and Yangqiu Song. Metagan: An adversarial approach to few-shot learning. In NeurIPS, pages 2365–2374, 2018. 3