Camera Relocalization by Computing Pairwise Relative Poses Using Convolutional Neural Network

Zakaria Laskar *[1], Iaroslav Melekhov *[1], Surya Kalia[2] and Juho Kannala[1]

[1]Aalto University, Finland , [2]IIT, Delhi, India

Abstract

We propose a new deep learning based approach for camera relocalization. Our approach localizes a given query image by using a convolutional neural network (CNN) for first retrieving similar database images and then predicting the relative pose between the query and the database images, whose poses are known. The camera location for the query image is obtained via triangulation from two relative translation estimates using a RANSAC based approach. Each relative pose estimate provides a hypothesis for the camera orientation and they are fused in a second RANSAC scheme. The neural network is trained for relative pose estimation in an end-to-end manner using training image pairs. In contrast to previous work, our approach does not require scene-specific training of the network, which improves scalability, and it can also be applied to scenes which are not available during the training of the network. As another main contribution, we release a challenging indoor localisation dataset covering 5 different scenes registered to a common coordinate frame. We evaluate our approach using both our own dataset and the standard 7 Scenes benchmark. The results show that the proposed approach generalizes well to previously unseen scenes and compares favourably to other recent CNN-based methods[†].

1. Introduction

Camera relocalization, or image-based localization is a fundamental problem in robotics and computer vision. It refers to the process of determining camera pose from the visual scene representation and it is essential for many applications such as navigation of autonomous vehicles, structure from motion (SfM), augmented reality (AR) and simultaneous localization and mapping (SLAM). Due to importance of these problems various relocalization approaches have been proposed. Point-based localization approaches find correspondences between local features extracted from an image by applying image descriptors (SIFT, ORB, etc [2, 20, 27]) and 3D point clouds of the scene obtained from SfM. In turn, such set of 2D-3D matches allows to recover the full 6-DoF (location and orientation) camera pose. However, this low-level process of finding matches does not work robustly and accurately in all scenarios, such as textureless scenes, large changes in illumination, occlusions and repetitive structures.

Recently, various machine learning methods [3, 32, 36], such as scene coordinate regression forest (SCoRF) [32, 36], have been successfully applied to camera localization problem. SCoRF utilize predicted 3D location of four pixels of an input image to generate an initial set of camera pose hypotheses which are subsequently refined by a RANSAC loop. However, all these methods require depth maps associated with input images at training time, thus the applicability of such approaches is restricted.

Inspired by the success in image classification [13, 17], semantic segmentation [14, 23] and image retrieval [1, 10], convolutional neural networks (CNNs) have also been used to predict camera pose from visual data [15, 16]. They cast camera relocalization as a regression problem, where camera location is directly estimated by CNN pre-trained on image classification data [8]. Although learning-based approaches overcome many disadvantages of point-based methods, they still have certain limitations. Directly regressing the absolute camera pose constrains the current machine learning models to be trained and evaluated scene-wise when the scenes are registered to different coordinate frames. The reason for this is that the trained model learns a mapping from image (pixels) to pose which is dependent on the coordinate frame of the training data belonging to a particular scene. This causes complications, especially if one is interested in localization across several scenes simultaneously, and also prevents transferring learnt knowledge of geometric relations between scenes. The second problem is the obviously limited scalability to large environments since

*Equal contribution: firstname.lastname@aalto.fi

a finite neural network has an upper bound on the physical area that it can learn, as pointed out by [16].

In this paper, we propose to decouple the learning process from the coordinate frame of the scene. That is, instead of directly regressing absolute pose like [15, 16, 21], we train a Siamese CNN architecture to regress the relative pose between a pair of input images and propose a pipeline for computing the absolute pose from several relative pose estimates between the query image and database images. This approach is flexible and has several benefits: *(a)* our CNN can learn from image pairs of any scene thereby being able to improve towards generic relative pose estimator; *(b)* a single network can be trained and used for localization in several disjoint scenes simultaneously, even in scenes whose training images are scarce or not available during the training time of the network; and *(c)* the approach is scalable because a single CNN can be used for various scenes and the full scene-specific database (*i.e*. training) images are not needed in memory at test time either as compact feature descriptors and fast large-scale image retrieval techniques can be utilized instead.

To summarize, we make the following contributions:

- We propose a new deep learning based approach for camera relocalization. Our approach is general and scalable alternative to previous models and compares favourably to other CNN-based methods.
- We show through extensive evaluation the generalization capability of the approach to localize cameras in scenes unseen during training of the CNN.
- We introduce a new challenging indoor dataset with accurate ground truth pose information and evaluate the proposed method also on this data.

The rest of the paper is organized as follows. Section 2 discusses related work in image-based localization. Section 3 describes the network structure and the whole pipeline of our approach. The details of a new large indoor dataset and evaluation results of our method are provided in Section 4 and Section 5 accordingly. Conclusion and some suggestions for future work are given in Section 6.

We will make the source code and the dataset publicly available upon publication of the paper.

2. Related work

Camera relocalization approaches largely belong to two classes: visual place recognition methods and 3D model-based localization algorithms. Visual place recognition methods cast image-based localization problem as an image retrieval task and apply standard techniques such as image descriptors (SIFT, ORB, SURF [2, 20, 27]), fast spatial matching [25], bag of visual words [7, 37] to find a representation of an unknown scene (a query image) in a database of geo-tagged images. Then, the geo-tag of the most relevant retrieved database image is considered as an approximation of a query location. The major limitation of visual recognition methods is that the images in database are often sparse, so that in situations where the query is far from database images the estimate would be inaccurate [36].

Structure-based localization methods utilize a 3D scene representation obtained from SfM and find correspondences between 3D points and local features extracted from a query image establishing a set of 2D-3D matches. Finally, the camera pose is established by applying RANSAC loop in combination with a Perspective-n-Point algorithm [4]. However, descriptor matching is expensive and time-consuming procedure making camera relocalization complicated problem for large scale scenes such as urban environment. In order to accelerate this stage, [19, 29] eliminate correspondence search as soon as enough matches have been found, and [28, 31] propose to perform matching with the 3D points of top-retrieved database images.

Sattler *et al*. [30] demonstrate that combining visual place recognition approaches with local SfM leads to better localization performance compared with 3D-based methods. However, the localization process itself is still very time-consuming.

It has also been shown that machine learning methods have potential for providing efficient solutions to the pose estimation problem. Similar to structure-based localization approaches, Shotton *et al*. [32] utilize a regression forest to predict a 3D point location for each pixel of an input RGB-D image. Thus, the method establishes 2D-3D matches which are then used to recover 6-DoF camera pose by applying RANSAC. Rather then finding point correspondences, Valentin *et al*. [36] propose to exploit the uncertainty of the predicted 3D point locations during pose estimation. Brachmann *et al*. [3] propose a differentiable RANSAC method for camera localisation from an RGB image. However, it still requires dense depth maps in the training stage.

Recently proposed CNNs-based approaches have shown great success in image-based localization. Originally, utilizing CNNs to directly regress camera relocalization was proposed by Kendall *et al*. [16]. Their method, named PoseNet, adapts GoogLeNet [34] architecture pre-trained on large-scale image classification data to reconstruct 6-DoF camera pose from an RGB image. In the recent paper [15], Kendall *et al*. propose a novel loss function based on scene reprojection error and show its efficiency in appearance-based localization. Contrary to [15, 16], HourglassPose [21] utilizes a symmetric encoder-decoder network structure with skip connections which leads to improvement in the localization accuracy outperforming PoseNet. Partly motivated by advances of Recurrent Neural Networks (RNN) in text classification, Clark *et al*. and

Walch *et al*. apply LSTM networks to determine the location from which a photo was taken [5, 38]. Following [16], both of these methods, called VidLoc [5] and LSTMPose [38], utilize GoogLeNet to extract features from input images to be localized, then feeding these features to a block of LSTM units. The regression part consisting of fully-connected layers utilizes output of LSTM units to predict camera pose. The major drawback of VidLoc is that it requires a sequence of adjacent image frames as input and is able to estimate camera translation only.

The proposed approach is related to all previously discussed CNN-based methods, but it is the first one solving image-based localization problem via relative camera pose. Inspired by [22, 24, 35], we apply Siamese neural network to predict relative orientation and relative translation between two views. These relative translation estimates are then triangulated to recover the absolute camera location. Compared with [5, 15, 16, 21, 38], our study provides more general and scalable solution to image-based localization task. That is, the proposed approach is able to estimate 6-DoF camera pose for scenes registered to different coordinates frames, unlike existing CNN-based methods. Finally, compared with traditional machine learning approaches, our method does not require depth maps for training, thus it is applicable for outdoor scenes as well. Further details of our approach will be given in the following sections.

3. Proposed approach

This section introduces the proposed localization approach for predicting camera pose. The method consists of two modules: a Siamese CNN network for relative pose computation and the localization pipeline. The input to the system is an RGB query image to be localized, and a database of images with their respective poses. At the first stage, we construct a set of training image pairs and use it to train a Siamese CNN to predict relative camera pose of each pair (Section 3.1). It should be noted that the training image pairs can be independent of the scenes present in the localization database.Then, each trained branch of the network is considered a feature extractor and the extracted feature vectors can be utilized to identify the database images that are nearest neighbours (NN) to the query image in the feature space. Finally, relative pose estimates between the query and its neighbours are computed and then coalesced with ground truth absolute location of the corresponding database images in a novel fusion algorithm (Section 3.2) producing the full 6-DoF camera pose.

3.1. Pairwise Pose Estimation

The first part of the proposed approach aims to directly estimate relative camera pose from an RGB image pair. The problem of regressing rigid body transformation between a pair of images has been well studied in recent years [9, 22, 35]. Following [22], we construct a Siamese neural network with two representation branches connected to a common regression part as shown in Fig. 1. The branches share the same set of weights and have ResNet34 architecture [13] truncated at the last average pooling layer. The weights are initialised from a network pre-trained for large-scale image classification task [8], and later fine-tuned for the relative pose estimation task as described below. The outputs of the representation branches are concatenated and passed through the regressor which consists of three fully-connected layers (FC), namely FC_i, FC_r and FC_t, where the latter two predict relative orientation and translation, respectively. Fig. 1 shows a detailed description. The fully-connected layers are initialized randomly.

The output of the regression part is parameterized as 4-dimensional quaternion for relative orientation Δr and 3-dimensional Δt for relative translation [22, 24]. As the quaternions lie on a unit sphere, enforcing unit norm constraint on any 4-D vector outputs a valid rotation. Also the distance between two quaternions $s(r_i, r_k)$ can be measured by the Euclidean l_2 norm $||r_i - r_k||_2$, unlike other rotation parameterizations such as rotation matrices that lie on a manifold and distance computation requires finding an Euclidean embedding. For a more elaborate discussion we guide the reader to [12, 15]. To regress relative camera pose, we train our CNN with the following objective criterion:

$$\mathcal{L} = \|\Delta t_{gt} - \Delta t\|_2 + \beta \|\Delta r_{gt} - \Delta r\|_2 \qquad (1)$$

where Δr_{gt} and Δt_{gt} are the ground-truth relative orientation and translation respectively. To balance the loss for orientation and translation we use the parameter $\beta > 0$ [22, 24]. The details of the training are described in Section 5.

At test time, a pair of images is fed to the regression neural network, consisting of two branches, which directly estimates the real-valued parameters of the relative camera pose vector. Finally, the estimated quaternion and translation vectors are normalized to unit length. The normalized translation vector gives the translation direction from the database image to the query camera location. Although the translation vector predicted by our network contains scale information, we found that in practice recovering the scale using the approach discussed in Section 3.2 is more accurate and reliable.

3.2. Localization Pipeline

Retrieving the nearest neighbours In order to find the nearest database images to the query, it is important to obtain a suitable representation for both the query q and the database images D. Considering recent advances of CNN-based approaches in the field of image retrieval [10, 26], we

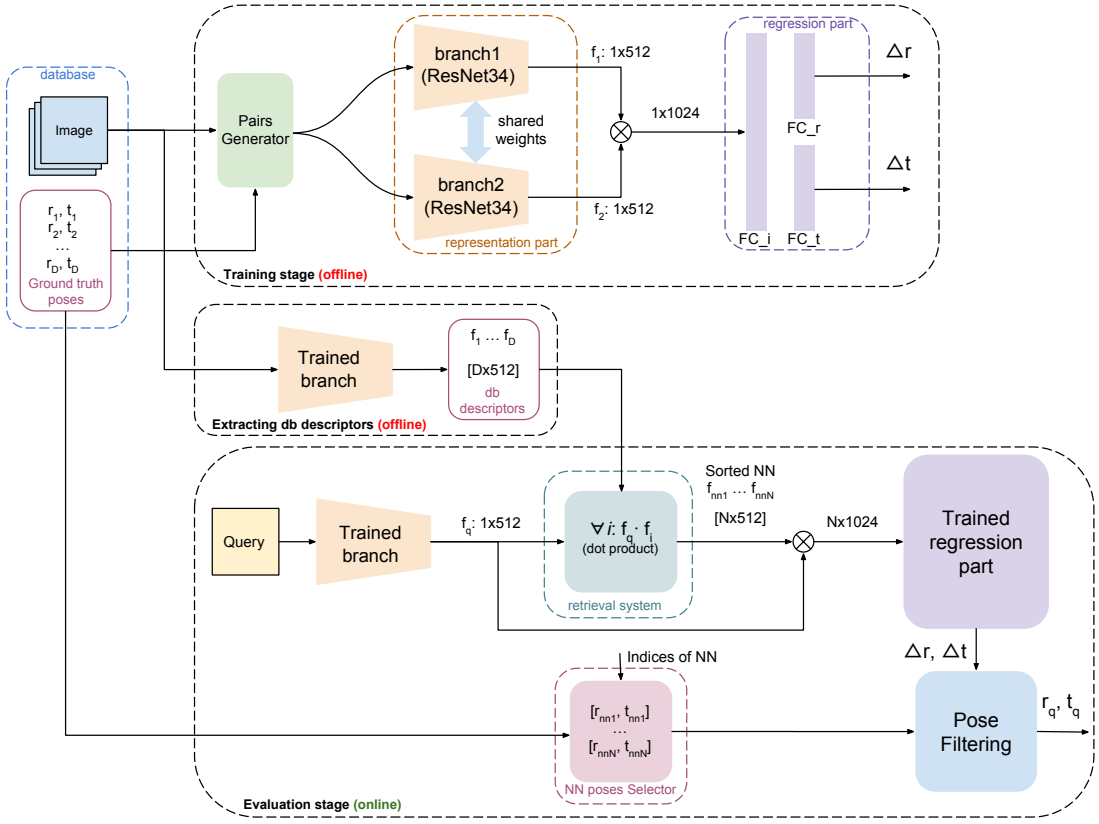

Figure 1: Pipeline of the proposed system. A Siamese based CNN network consisting of two branches pre-trained on ImageNet [8] is trained to directly regress relative pose between a pair of cameras (top). Once the training process is finished, we employ the branch as a descriptor to compute feature representations of database and query images. Then, the dot product is applied to these representations as a part of retrieval system and database descriptors are ranked according to higher similarity score. Consequently, query descriptor and its top N ranked database representations are concatenated and fed to the regression part of the network to predict pairwise relative pose. Finally, the proposed fusion algorithm naturally coalesces relative pose estimates and ground truth absolute poses to produce the full 6-DoF query location.

use the fine-tuned network (one branch of the model architecture) from the first stage (Section 3.1) as a feature extractor to encode the query and database images to fixed global representations (*i.e.* 512 dimensional feature vectors, see Fig. 1). In turn, the dot product of the query and database feature vectors is computed to obtain image similarity. Consequently, the database images are ranked according to similarity scores. Finally, the top N ranked database images, $d = \{d^j | d^j \in D, j = 1...N\}$ are selected as the nearest neighbours to the query image, q.

Although the retrieval stage is an important component in our pipeline, we adopted the simple approach above in this work and postpone deeper discussion and analysis to future work. The primary focus of our paper is not image retrieval, but camera localization. However, in Section 5, besides evaluating the performance of the complete pipeline, we also experimentally study how the system would perform with perfectly accurate retrieval stage.

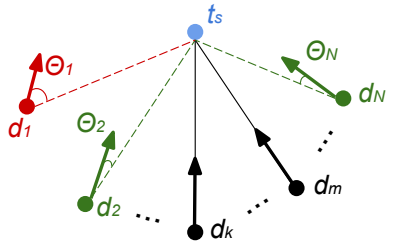

Figure 2: Estimation of query camera translation.

Compute Relative Pose In the next stage of the pipeline the Siamese CNN is used to regress relative camera pose for the image pairs $Q = q \times d$, $\Delta R = \{\Delta R^1, \Delta R^2 ... \Delta R^N\}$ and $\Delta \hat{t} = \{\Delta \hat{t}^1, \Delta \hat{t}^2, ... \Delta \hat{t}^N\}$. Here, ΔR^j represents the relative orientation between the j^{th} NN database image, $d^j \in d$ and the query q. Similarly, $\Delta \hat{t}^j$ is the relative translation

direction.

Pose hypotheses filtering The final part of the localization system is a novel fusion algorithm recovering the absolute query camera pose from these N relative pose estimations. This process is illustrated schematically in Fig. 2.

From the shortlisted top ranked database images d, select a pair $p^s = \{d^k, d^m\}$, where $p^s \subset d$ and $s = 1, 2, \ldots, \binom{N}{2}$. Now, the translation direction predictions to the query q from the images p^s are triangulated to obtain the location/translation parameter of query camera, t^s. This gives us $\binom{N}{2}$ hypotheses for the query translation, $\tilde{t}_q = \{t^s | s = 1, 2, \ldots, \binom{N}{2}\}$. Now, for each $t^s \in \tilde{t}_q$ the direction rays from the camera centers of the remaining images in d, $p^r = d \setminus p^s$ to this triangulated camera location t^s is obtained. If the direction ray associated with an image in p^r is within a pre-defined angular distance to the direction vector predicted by our network, then it is considered an inlier (d_2 and d_N in Fig. 2) to the estimate t^s. The translation estimate $t^s \in \tilde{t}_q$, $s = 1, 2, \ldots, \binom{N}{2}$ with the highest inlier count is assigned to the query camera. If two or more translation estimates have the same inlier count, then they are averaged to obtain the final query translation estimate.

Estimating rotation for the query camera is much simpler as the following equation can be used to obtain a hypothesis:

$$\Delta R^j = R_j^T R_q^j \tag{2}$$

where R_j is the orientation of the j^{th} camera in d, $d^j \in d$ available as input to our system, and R_q^j is the j^{th} hypothesis of the query camera orientation. These results in N hypotheses for query orientation, $\tilde{R}_q = \{R_q^1 ... R_q^N\}$. Instead of naively averaging the estimations, we apply a consensus based filtering similar to the process of estimating query translation. For each hypothesis, $R_q^j \in \tilde{R}_q$ a consensus set is formed from the remaining hypotheses in $\tilde{R}_q$ that lie within a pre-defined angular distance to R_q^j. The number of elements in the consensus set is defined as the inlier count for R_q^j. The hypothesis with the highest inlier count is assigned as the orientation estimate for the query camera. When two or more hypotheses share the same inlier count, a robust rotation averaging algorithm [11] is applied to obtain the final query camera rotation.

4. Datasets

We evaluate the proposed approach on two different dataset for indoor localization.

7Scenes Microsoft 7-Scenes dataset contains RGB-D images covering 7 different indoor locations [33]. The dataset has been widely used for image-based localization [5, 15, 16, 21] exhibiting significant variation in camera pose, motion blur and perceptual aliasing. In our experiments we utilize the same train and test datasets for each scene as provided in the original paper.

University The scenes in *7Scenes* dataset have their own coordinate system without being linked to each other in a global coordinate system. Therefore, all existing machine learning models are trained and evaluated scene-wise. This fundamental limitation restricts to widely apply these approaches to real life scenarios where a large-scale environment consists of multiple sub-scenes.

We release a challenging indoor localization dataset, *University*, with different locations that are all registered to a common global coordinate frame. For this paper, we consider a segment of the whole dataset, consisting of image sequences of 5 scenes, *Office, Meeting, Kitchen, Conference, and Coffee Room*. The scenes are structured in a similar way to *7Scenes*, with multiple traversals through each of the scenes. The training and test split of the sequences are provided. Overall, the proposed dataset contains 9694 training and 5068 test images respectively. Some challenging test cases of *University* dataset are presented in Figure 1 of the Supplementary material.

Ground-truth localization data of the dataset was generated using *Google Project Tango's* tablet and high-resolution image sequences were collected by iPhone 6S mounted on top of the tablet. The device outputs two types of odometry estimations: Start of Service (SOS), which is the raw odometry and thus suffers from drift, and Area Learning (AL), which uses device's drift correction engine. As the AL engine fails sometimes [18], we use the odometry estimates from SOS and manually generated location constraints in a pose-graph optimization framework to generate a globally consistent map.

5. Experiments

In this section we quantitatively demonstrate the effectiveness of the proposed system on *7Scenes* and *University* datasets. We compare our approach with the current state-of-the-art CNN-based methods, such as PoseNet [16], HourglassPose [21], LSTMPose [38], VidLoc [5], and PoseNet with reprojection loss [15], dubbed PoseNet2.

According to Fig.1, representation part of the proposed system is based on a Siamese architecture. In this work, we initialize our model using original ResNet34 model truncated at the last average pooling layer and pre-trained on ImageNet [8] as a structure of each branch.

Training data We start experiments by evaluating the system on *7Scenes* dataset. As it is mentioned in Section 4, the dataset consists of different indoor scenes and each of them provides training and testing image sequences. Since the system requires an image pair to learn relative pose, the following strategy is applied to obtain training dataset. For every image in the training set of each scene, we find a corresponding image from the same set so that they have sufficiently overlapping field of view. Total number of training pairs of a scene is equal to the number of images in the

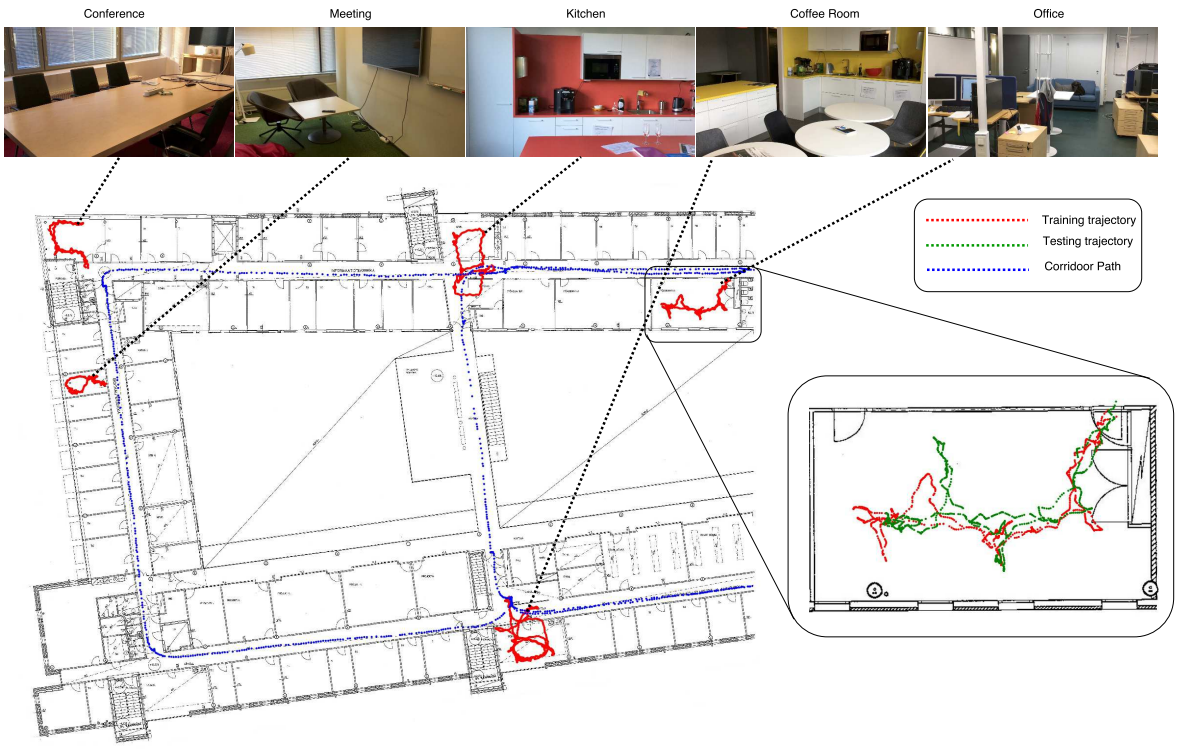

Figure 3: The University dataset. The proposed large-scale indoor localization dataset consists of 5 different scenes (segments) registered to a common global co-ordinate system.

Scene	Spatial Extent	PoseNet [16]	LSTM-Pose [38]	VidLoc [5]	Hourglass-Pose [21]	PoseNet2 [15]	ResNet34-Pose (baseline)	Ours
Chess	$3 \times 2 \times 1$m	0.32m, 8.12°	0.24m, 5.77°	0.18m, N/A	0.15m, 6.53°	0.13m, 4.48°	0.12m, 6.69°	0.13m, 6.46°
Fire	$2.5 \times 1 \times 1$m	0.47m, 14.4°	0.34m, 11.9°	0.26m, N/A	0.27m, 10.84°	0.27m, 11.3°	0.31m, 13.36°	0.26m, 12.72°
Heads	$2 \times 0.5 \times 1$m	0.29m, 12.0°	0.21m, 13.7°	0.14m, N/A	0.19m, 11.63°	0.17m, 13.0°	0.16m, 13.78°	0.14m, 12.34°
Office	$2.5 \times 2 \times 1.5$m	0.48m, 7.68°	0.30m, 8.08°	0.26m, N/A	0.21m, 8.48°	0.19m, 5.55°	0.21m, 8.78°	0.21m, 7.35°
Pumpkin	$2.5 \times 2 \times 1$m	0.47m, 8.42°	0.33m, 7.00°	0.36m, N/A	0.25m, 7.01°	0.26m, 4.75°	0.25m, 7.89°	0.24m, 6.35°
Red Kitchen	$4 \times 3 \times 1.5$m	0.59m, 8.64°	0.37m, 8.83°	0.31m, N/A	0.27m, 10.15°	0.23m, 5.35°	0.22m, 9.35°	0.24m, 8.03°
Stairs	$2.5 \times 2 \times 1.5$m	0.47m, 13.8°	0.40m, 13.7°	0.26m, N/A	0.29m, 12.46°	0.35m, 12.4°	0.37m, 14.45°	0.27m, 11.82°
Average		0.44m, 10.4°	0.31m, 9.85°	0.25m, N/A	0.23m, 9.53°	0.23m, 8.12°	0.23m, 10.61°	0.21m, 9.30°

Table 1: Camera localization performance of the proposed method and existing RGB-only CNN-based approaches for *7Scenes* datasets. We follow original notation presented in [16] and provide median translation and orientation errors. In terms of localization error, our approach is superior to other methods utilizing similar loss (1) such as PoseNet [16], LSTM-Pose [38], VidLoc [5] and Hourglass-Pose [21] for the all scenes. Furthermore, the proposed method outperforms PoseNet2 [15] in 4 scenes and achieves better localization accuracy in general. However, it is important to note that both methods are not directly comparable, due to more advanced loss function optimized in [15].

training sequence of this scene in the original dataset. Finally, the training pairs from all the scenes are merged into a single training set for training the CNN in our approach.

Training details As a preprocessing step, the training images are resized to 256 pixels on the smaller side while maintaining the aspect ratio of the resized image. The training images are further mean-centered and normalized using standard deviation computed over the whole training set. To ensure fixed sized input to our network, we use random crops (224×224) during training and perform the evaluation using central crops at test time. The network is trained for 200 epochs with an initial learning rate of 0.1 which is gradually decreased by 10 times after every 50 epochs. The loss (1) is minimized using stochastic gradient descent with a batch size of 64 training samples. The scale factor β is set to 1 after empirical evaluation for all our experiments.

Scene	Baseline	Proposed
Office	2.76m, 17.08°	0.57m, 17.09°
Meeting	2.13m, 14.13°	2.30m, 12.27°
Kitchen	1.65m, 12.92°	0.70m, 11.72°
Conference	4.97 m, 17.18°	2.74m, 15.00°
Average	2.88 m, 15.33°	1.58 m, 14.02°

Table 2: Camera relocalization performance of the proposed approach and the baseline on *University* dataset presented as median orientation and translation errors. Training is done using the training images of all scenes for both approaches. Evaluation is performed scene-wise.

Removed scene	Median error position [m]	Median error orientation [deg]
Chess	0.27	13.05
Heads	0.23	15.03
Red Kitchen	0.36	12.60

Table 3: Generalization performance of the proposed approach. Localization accuracy of the proposed method on unseen scenes of *7Scenes* dataset.

Scene	Median error position [m]	Median error orientation [deg]
Chess	0.31	15.05
Fire	0.40	19.00
Heads	0.24	22.15
Office	0.38	14.14
Pumpkin	0.44	18.24
Red Kitchen	0.41	16.51
Stairs	0.35	23.55
Average	0.36	18.38

Table 4: Generalization performance of the proposed approach. The network is trained on only *University* and evaluated on *7Scenes* dataset.

The weight decay is set to 10^{-5} and no dropout was used in our experiments. All experiments were evaluated on two NVIDIA Titan X GPUs using Torch7 [6] machine learning framework.

Evaluation stage The input to the system is a database containing the list of images from the training set of all scenes (for a given dataset) and their respective camera poses. The combined list of test images from all scenes constitute the query set. For each query, we retrieve its top 5 NN ($N = 5$) from the database images using neural representations from our trained representation branch.

The query and its NN are then fed sequentially to our Siamese model to obtain the relative camera pose estimates. From a practical standpoint it is not necessary to feed the query and its NN images through the full network model. The only component of our network that requires pairwise input is the regression part, which takes in input from the representation part of each branch of the Siamese model. Also, both the representation branches share the same parameters and the output of the representation part is already used in the first stage of our pipeline to compute image similarity. Thereby, to compute relative pose, we simply feed the representations of the query and its NN in a pairwise manner to the regression component.

The relative pose estimations are then robustly fused to obtain the query camera pose. The angular distance threshold of inliers for both translation and rotation is 20 degrees.

5.1. Quantitative Results

We compare our proposed system with the existing CNN based localization methods on *7Scenes*, while for *University* dataset we provide an evaluation of our proposed system and a baseline method. The results are shown in Table 1 and Table 2. For both datasets, the localization performance is measured as the median orientation and translation error over each scene.

For several scenes in the *7Scenes* dataset we outperform other CNN-based methods in camera relocalization. In particular, we perform favourably to the current best performing method, PoseNet2 on several scenes. However, PoseNet2 is not directly comparable to our work as it uses a more sophisticated loss function during training and a different CNN architecture.

For a fairer evaluation we compare our system with a baseline model consisting of pre-trained convolutional layers of ResNet34 architecture and a regression part replicating the one utilized in the proposed approach (but without the Siamese architecture). We entitle this model ResNet34-Pose. Following [15, 21], the baseline is trained and evaluated scene-wise. Table 1 shows that our proposed system has a consistent improvement over the baseline for both rotation and translation across all the scenes. Although the margin of improvement is not large, it is to be noted that all the existing methods (including the baseline) are trained in a scene-specific manner whereas our system was designed to inherently overcome this fundamental limitation and allows us to train and test our model jointly on all the scenes. That is, our approach uses the same network for all 7 scenes whereas other approaches of Table 1 have one network per scene (*e.g.* in total ResNet34-Pose has thus 7 times more parameters that are learnt).

The performance increase of our system compared to the baseline can be attributed to a number of factors: *i)* representation sharing across scenes during training, *ii)* generating multiple hypothesis for query camera pose followed by robust pose filtering, *iii)* larger training set. These factors,

although plausible have not been experimentally validated in this paper and we leave it for future work.

For the *University* dataset, our system and the baseline are trained jointly on the scenes: *Office, Meeting, Kitchen* and *Conference*. The scene *Coffee Room* is a recent addition to the database, and due to time constraints we could not train and evaluate our proposed system and ResNet34-Pose on this scene. However, we use it to evaluate our trained model in Section 5.2. The performance evaluation is presented in Table 2. The results show that the margin of translational error between the baseline and our system has increased significantly. In particular, it has increased from 2cm in *7 Scenes* (all scenes combined) to 130cm in *University*. Although it does demonstrate that the proposed system performs better even under similar training setup, it also provides additional insight on the scalability of our system. As mentioned in Section 4, the *University* dataset consists of several scenes spread over an area of 2500 m^2. Absolute pose prediction models like the baseline model need to maintain a track of the spread or scale of the map, while models like our system are not much affected by scale. Our system essentially removes the influence of scale by finding the NN and solving the relative pose problem which does not depend on the scale of the map/dataset.

5.2. Generalization Performance

Current machine learning models for camera localization are not only restricted to scene-wise training and evaluation, but also limited in their applicability to previously unseen scenes. In this section we experimentally demonstrate the generalization capability of our pipeline to data previously unseen during training.

We hold out one of the scenes in *7Scenes* dataset for evaluation and train our model on the remaining 6 scenes. In particular, we held out *Chess, Heads,* and *RedKitchen* separately as evaluation sets. Table 3 shows a graceful drop in performance on the held out test scene compared to the case where our model was trained on all the 7 scenes (Table 1). In *University* dataset, we evaluate on *Coffee Room* using the model trained on the remaining 4 scenes. The median position and orientation error were 1.44 m and 19.22 degrees.

We further evaluate the performance on the *7Scenes* dataset using our model trained on the *University* dataset (excluding *Coffee Room*). According to Table 4, the performance drop is not drastic, with the mean of the median error over all the scenes drop by 9 degrees and 15 cm for rotation and translation respectively.

5.3. Ideal Retrieval Results

We now evaluate the scenario when the retrieval stage of our pipeline returns only the true nearest neighbours to the query. We further evaluate the effect of image spacing between the query and the retrieved NN. These experiments are evaluated on *7Scenes* dataset.

Scene	Viewpoint 0	Viewpoint 3	Viewpoint 7
Chess	0.19m, 7.48°	0.16m, 7.26°	0.16m, 7.61°
Fire	0.13m, 6.61°	0.10m, 6.45°	0.11m, 6.32°
Heads	0.25m, 8.74°	0.25m, 8.54°	0.25m, 8.71°
Office	0.21m, 11.13°	0.19m, 11.14°	0.19m, 11.95°
Pumpkin	0.24m, 9.39°	0.26m, 9.50°	0.25m, 9.35°
Red Kitchen	0.21m, 7.59°	0.19m, 7.41°	0.19m, 7.45°
Stairs	0.23m, 7.92°	0.23m, 8.26°	0.23m, 8.56°
Average	0.21m, 8.41°	0.20m, 8.37°	0.20m, 8.44°

Table 5: Camera relocalization accuracy of the proposed system for different viewpoint changes between the query and the database image.

Due to low image spacing between training images across all the scenes in *7Scenes*, a query often has more than 300 true NN. Now, for a given query we use ground truth to sort all the training images from the corresponding scene using a metric similar to (1). We then create a sublist consisting of the top 355 images from this sorted list. From this sublist, we further select 8 sets of 5 images each at an interval of 50 ranks. That is, the first set (Viewpoint 0 in Table 5) contains images ranked 1-5, the next set (Viewpoint 1 in Supplementary) consisting of images ranked 51-55 and so on. We then evaluate our proposed system using these sets of true NN instead of the one obtained using neural representations. Table 5 (and Table in Supplementary) shows that the proposed system has a consistent performance across wide viewpoint variation in the true NN. This is an indication that the pipeline is robust to the quality of the nearest neighbours. They do not necessarily need to be the database images that have most overlap with the query. On the other hand, the result also shows that with *7Scenes* our choice of $N = 5$ might not be the optimal choice. Increasing N will increase the chances of retrieving the true NN, and the consistent performance across all scenes and viewpoint changes suggests that the true NN have a higher likelihood of forming a consensus set (*c.f*. Section 3.2).

6. Conclusion

We addressed some of the challenges and limitations of the current setup in which machine learning models are trained and evaluated for camera localization. By leveraging the training images both at training and test time, we are able to mitigate these limitations and achieve competitive results on challenging datasets. Results demonstrate that the scope of the proposed system is easily extendable to scenes without prior training.

As future work, possible directions include training the network simultaneously with both relative pose and image similarity objectives [39]. Also, learning a better generic relative camera pose estimator [35] can improve the generalization performance of the proposed system.

References

[1] A. Babenko, A. Slesarev, A. Chigorin, and V. S. Lempitsky. Neural codes for image retrieval. In *Proc. ECCV*, 2014. 1

[2] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (SURF). *Comput. Vis. Image Underst.*, 110(3):346–359, 2008. 1, 2

[3] E. Brachmann, A. Krull, S. Nowozin, J. Shotton, F. Michel, S. Gumhold, and C. Rother. DSAC - differentiable RANSAC for camera localization. In *Proc. CVPR*, 2017. 1, 2

[4] M. Bujnak, Z. Kukelova, and T. Pajdla. New efficient solution to the absolute pose problem for camera with unknown focal length and radial distortion. In *Proc. ACCV*, 2011. 2

[5] R. Clark, S. Wang, N. T. Andrew Markham, and H. Wen. VidLoc: A deep spatio-temporal model for 6-DoF video-clip relocalization. In *Proc. CVPR*, 2017. 3, 5, 6

[6] R. Collobert, K. Kavukcuoglu, and C. Farabet. Torch7: A matlab-like environment for machine learning. In *BigLearn, NIPS Workshop*, 2011. 7

[7] O. G. Cula and K. J. Dana. Compact representation of bidirectional texture functions. In *Proc. CVPR*, 2001. 2

[8] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *Proc. CVPR*, 2009. 1, 3, 4, 5

[9] D. DeTone, T. Malisiewicz, and A. Rabinovich. Deep image homography estimation. In *RSS Workshop on Limits and Potentials of Deep Learning in Robotics*, 2016. 3

[10] A. Gordo, J. Almazan, J. Revaud, and D. Larlus. Deep image retrieval: Learning global representations for image search. In *Proc. ECCV*, 2016. 1, 3

[11] R. Hartley, K. Aftab, and J. Trumpf. L1 rotation averaging using the weiszfeld algorithm. In *Proc. CVPR*, 2011. 5

[12] R. Hartley, J. Trumpf, Y. Dai, and H. Li. Rotation averaging. *International journal of computer vision*, 103(3):267–305, 2013. 3

[13] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proc. CVPR*, 2016. 1, 3

[14] S. Hong, H. Noh, and B. Han. Decoupled deep neural network for semi-supervised semantic segmentation. In *NIPS*, 2015. 1

[15] A. Kendall and R. Cipolla. Geometric loss functions for camera pose regression with deep learning. In *Proc. CVPR*, 2017. 1, 2, 3, 5, 6, 7

[16] A. Kendall, M. Grimes, and R. Cipolla. PoseNet: A convolutional network for real-time 6-dof camera relocalization. In *Proc. ICCV*, 2015. 1, 2, 3, 5, 6

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in NIPS*, 2012. 1

[18] Z. Laskar, S. Huttunen, D. Herrera, E. Rahtu, and J. Kannala. Robust loop closures for scene reconstruction by combining odometry and visual correspondences. In *Proc. ICIP*, 2016. 5

[19] Y. Li, N. Snavely, and D. P. Huttenlocher. Location recognition using prioritized feature matching. In *Proc. ECCV*, 2010. 2

[20] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vision*, 60(2):91–110, Nov. 2004. 1, 2

[21] I. Melekhov, J. Ylioinas, J. Kannala, and E. Rahtu. Image-based localization using hourglass networks. In *Proc. ICCV Workshop*, 2017. 2, 3, 5, 6, 7

[22] I. Melekhov, J. Ylioinas, J. Kannala, and E. Rahtu. Relative camera pose estimation using convolutional neural networks. In *Proc. ACIVS*, 2017. 3

[23] H. Noh, S. Hong, and B. Han. Learning deconvolution network for semantic segmentation. In *Proc. ICCV*, 2015. 1

[24] G. L. Oliveira, N. Radwan, W. Burgard, and T. Brox. Topometric localization with deep learning. *CoRR*, abs/1706.08775, 2017. 3

[25] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *Proc. CVPR*, 2007. 2

[26] F. Radenović, G. Tolias, and O. Chum. CNN image retrieval learns from BoW: Unsupervised fine-tuning with hard examples. In *Proc. ECCV*, 2016. 3

[27] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski. Orb: An efficient alternative to sift or surf. In *Proc. ICCV*, 2011. 1, 2

[28] T. Sattler, M. Havlena, F. Radenovic, K. Schindler, and M. Pollefeys. Hyperpoints and fine vocabularies for large-scale location recognition. In *Proc. ICCV*, 2015. 2

[29] T. Sattler, B. Leibe, and L. Kobbelt. Efficient effective prioritized matching for large-scale image-based localization. *IEEE TPAMI*, 2016. 2

[30] T. Sattler, A. Torii, J. Sivic, M. Pollefeys, H. Taira, M. Okutomi, and T. Pajdla. Are Large-Scale 3D Models Really Necessary for Accurate Visual Localization? In *Proc. CVPR*, 2017. 2

[31] T. Sattler, T. Weyand, B. Leibe, and L. Kobbelt. Image retrieval for image-based localization revisited. In *Proc. BMVC*, 2012. 2

[32] J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi, and A. Fitzgibbon. Scene coordinate regression forests for camera relocalization in RGB-D images. In *Proc. CVPR*, 2013. 1, 2

[33] J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi, and A. Fitzgibbon. Scene coordinate regression forests for camera relocalization in RGB-D images. In *Proc. CVPR*, 2013. 5

[34] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. E. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proc. CVPR*, 2015. 2

[35] B. Ummenhofer, H. Zhou, J. Uhrig, N. Mayer, E. Ilg, A. Dosovitskiy, and T. Brox. DeMoN: Depth and motion network for learning monocular stereo. In *Proc. CVPR*, 2017. 3, 8

[36] J. Valentin, M. Nießner, J. Shotton, A. Fitzgibbon, S. Izadi, and P. H. S. Torr. Exploiting uncertainty in regression forests for accurate camera relocalization. In *Proc. CVPR*, 2015. 1, 2

[37] M. Varma and A. Zisserman. Classifying images of materials: Achieving viewpoint and illumination independence. In *Proc. ECCV*, 2002. 2

[38] F. Walch, C. Hazirbas, L. Leal-Taixé, T. Sattler, S. Hilsenbeck, and D. Cremers. Image-based localization with spatial LSTMs. In *Proc. ICCV*, 2017. 3, 5, 6

[39] A. R. Zamir, T. Wekel, P. Agrawal, C. Wei, J. Malik, and S. Savarese. Generic 3d representation via pose estimation and matching. In *Proc. ECCV*, 2016. 8