RelocNet: Continuous Metric Learning Relocalisation using Neural Nets

Vassileios Balntas[1], Shuda Li[1], and Victor Prisacariu[1]

Active Vision Lab, University of Oxford, UK
www.robots.ox.ac.uk/~lav
{balntas,shuda,victor}@robots.ox.ac.uk

Abstract. We propose a method of learning suitable convolutional representations for camera pose retrieval based on nearest neighbour matching and continuous metric learning-based feature descriptors. We introduce information from camera frusta overlaps between pairs of images to optimise our feature embedding network. Thus, the final camera pose descriptor differences represent camera pose changes. In addition, we build a pose regressor that is trained with a geometric loss to infer finer relative poses between a query and nearest neighbour images. Experiments show that our method is able to generalise in a meaningful way, and outperforms related methods across several experiments.

1 Introduction

Robust 6-DoF camera relocalisation is a core component of many practical computer vision problems, such as loop closure for SLAM [13, 37, 4], reuse a pre-built map for augmented reality [16] or autonomous multi- agent exploration and navigation [39].

Specifically, given some type of prior knowledge base about the world, the relocalisation task aims to estimate the 6-DoF pose of a novel (unseen) frame in the coordinate system given by the prior model of the world. Traditionally, the world is captured using a sparse 3D map built from 2D point features and some visual tracking or odometry algorithm [37]. To relocalise, another set of features is extracted from the query frame and is matched with the global model, establishing 2D to 3D correspondences. The camera pose is then estimated by solving the perspective-n-point problem [29, 32, 47, 30]. While this approach provides usable results in many scenarios, it suffers from exponentially growing computational costs, making it unsuitable for large-scale applications.

More recently, machine learning methods, such as the random forest RGB-D approach of [5] and the neural network RGB method of [25] have been shown to provide viable alternatives to the traditional geometric relocalisation pipeline, improving on both accuracy and range. However, this comes with certain downsides. The former approach produces state-of-the-art relocalisation results but requires depth imagery and has only been shown to work effectively indoors. The latter set of methods has to be retrained fully and slowly for each novel scene, which means that the learnt internal network representations are not transferable, limiting its practical deployability.

Our method (Fig. 1) leverages the ability of neural networks to deal with large-scale environments, does not require depth and aims to be transferable i.e. produce

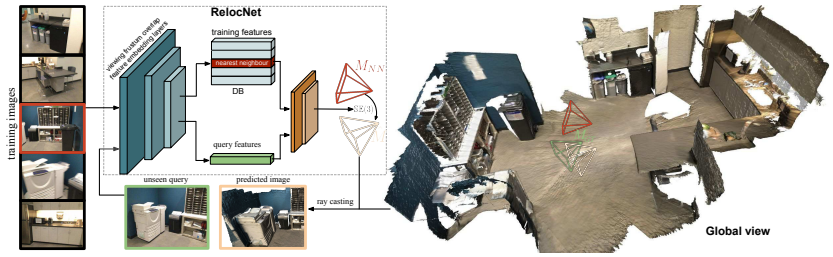


Fig. 1: Our system is able to retrieve a relevant item from a database, which presents high camera frustum overlap with an unseen query. Subsequently, we can use the pose information from the images stored in a database, to compute the pose of a previously unseen query by applying a transformation produced by a deep neural network. Note that the differential nature of our method enables the successful transfer of our learnt representation to previously unseen novel sequences (best viewed on screen).

accurate results on novel sequences and environments, even when not trained on them. Inspired by the image retrieval literature, we build a database of whole-image features, but, unlike in previous works, these are trained specifically for camera pose retrieval, and not holistic image retrieval. At relocalisation time, a nearest neighbour is identified using simple brute-forcing of L2 distances. Accuracy is further improved by feeding both the query image and the nearest neighbour features, in a Siamese manner, to a neural network, that is trained with a geometric loss and aims to regress the 6-DoF pose difference between the two images.

Briefly, our main contributions are:

- we employ a continuous metric learning-based approach, with a camera frustum overlap loss in order to learn global image features suitable for camera relocalisation;
- retrieved results are further improved by being fed to a network regressing pose differences, which is trained with exponential and logarithmic map layers directly in the pose homogeneous matrices space, without the need for separate translation and orientation terms;
- we introduce a new RGBD dataset with accurate ground truth targeting experiments in relocalisation.

The remainder of the paper is structured as follows: Section 2 describes related work. Section 3 discusses our main contributions, including the train and test methodologies and Section 4 shows our quantitative and qualitative evaluations. We conclude in Section 5.

2 Related work

Existing relocalisation methods can be generally grouped into five major categories: appearance similarity based, geometric, Hough transform, random forest and deep learning approaches.

Appearance similarity based approaches rely on a method to measure the similarity between pairs of images, such as Normalised Cross Correlation [15], Random Ferns [16] and bag of 2D features [14]. The similarity measurement can identify one or multiple reference images that match the query frame. The pose is then be estimated e.g. by a linear combination of poses from multiple neighbours, or simply by using the pose corresponding to the best match. However, these methods are often not accurate if the query frame is captured from a viewing pose that is far from those in the reference database. For this reason, similarity-based approaches, such as DBoW [14], are usually used as an early warning system to trigger a geometric approach for pose estimation [37]. The first stage of our own work is inspired by this category of methods, with pose-specific descriptors representing the database and query images.

Geometric relocalisation approaches [6, 21, 30] tackle the relocalisation problem by solving either the absolute orientation problem [20, 1, 41, 35, 31] or the perspective-n-point problem [29, 32, 47] given a set of point correspondences between the query frame and a global reference model. The correspondences are usually provided using 2D or 3D local feature matching. Matching local features can be noisy and unreliable, so pairwise information can be utilised to reduce feature matching ambiguity [30]. Geometric approaches are simple, accurate and especially useful when the query pose has large $\mathbb{SE}(3)$ distance to the reference images. However, such methods are restricted to a relatively small working space due to the fact that matching cost, depending on the matching scheme employed, can grow exponentially with respect to the number of key points. In contrast, our approach scales (i) linearly with the amount of training data, since each image needs a descriptor built, and (ii) logarithmically with the amount of test data, since database searches can usually be done with logarithmic complexity.

Hough Transform methods [11, 2, 40] rely entirely on pairwise information between pairs of oriented key points, densely sampled on surfaces. The pose is recovered by voting in the Hough Space. Such approaches do not depend on textures, making them attractive in object pose estimation for minimally-textured objects [40]. However, sampling densely on a 3D model for the point pair features is computationally expensive and not scalable. In addition, since the pose relocalisation requires both a dense surface model and a depth map, it is unsuitable for vison-only sensors. In contrast, our method only requires RGB frames for both training and testing.

Random forest based methods [42, 17, 45] deliver state-of-the-art accuracy, by regressing the camera location for each point in an RGBD query frame. Originally, such approaches required expensive re-training for each novel scene, but [5] showed that this can be limited to the leaf nodes of the random forest, which allowed for real-time performance. However, depth information is still required for accurate relocalisation results.

Convolutional neural network methods, starting with PoseNet [25], regress camera poses from single RGB images. Subsequent works (i) examined the use of recurrent neural networks (i.e. LSTMs) to introduce temporal information to the problem [46, 7], and (ii) trained the regression with geometric losses [24].

Most similar to our own approach are the methods of [44, 28], with the former assuming the two frames are given, and regressing depth and camera pose jointly, and

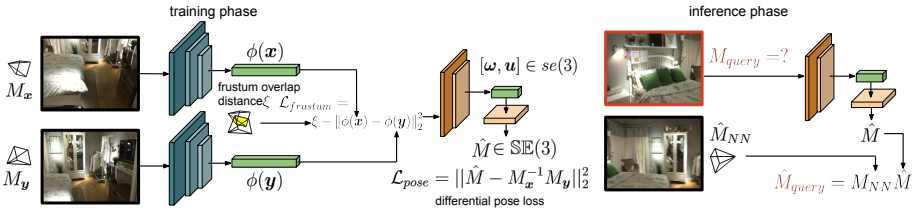


Fig. 2: (left) *Training stage*. We use a Siamese architecture to train global feature descriptors driven by a continuous metric learning loss based on camera frustum overlaps. This forces the representations that are learnt to be relevant to fine-grained camera pose retrieval. In addition, a final query pose is learnt based on a loss on a subsequent set of layers which are trained to infer the differential pose between two inputs. (right) *Inference stage*. Given an unseen image, and its nearest neighbour retrieved using our optimised frustum feature descriptors, we are able to compute a pose estimation for the unseen query based on the output of our differential pose network, and the stored nearest neighbour pose.

the later using ImageNet-trained ResNet feature descriptor similarity to identify the nearest neighbouring frame.

Compared to these approaches, we use a simpler geometric pose loss, and introduce a novel continuous metric learning method to train full frame descriptors specifically for camera pose-oriented retrieval.

3 Methodology

In this section, we present a complete overview of our method (Fig. 2), consisting of learning (i) robust descriptors for camera pose-related retrieval, and (ii) a shallow differential pose regressor from pairs of images.

3.1 Learning camera pose descriptors for retrieval using camera frustum overlaps

The first part of our method deals with learning suitable feature descriptors for retrieval of nearest neighbours that are consistent with the camera movement.

Motivation Several methods use pre-trained models for retrieval of relevant images, because such models are trained on large datasets such as ImageNet [9] or Places [48], and are able to capture relevant image features in their penultimate layers. With no significant effort, such models can be used for several other transfer learning scenarios. However, such features are trained for detection and recognition of final objectives, and might not be directly relevant to our problem, i.e. understanding the camera movement.

Recent work has shown that features that are learnt guided from object poses [3] can lead to a more successful object pose retrieval. To tackle the equivalent issue in terms of camera poses, we make use of the camera frustum overlaps as described below.

Algorithm 1: *Frustum overlap distance* between a pair of camera poses

Input : Relative pair pose $\boldsymbol{M} \in \mathbb{SE}(3)$, camera intrinsics $\boldsymbol{K}$, maximum clipping depth D, sampling step τ

1 Use $\boldsymbol{K}$ to sample a uniform grid $\mathcal{V}$ of voxels with size τ inside the first frustum with max clipping distance D.

2 Compute the subset of voxels $\mathcal{V}_+ \subseteq \mathcal{V}$ which lie inside the second frustum.

Return: Frustum overlap distance $\xi = 1 - \frac{|\mathcal{V}_+|}{|\mathcal{V}|},\ with\ \xi \in [0, 1]$

Frustum Overlap Loss To capture relevant features in the layers of our network, our main idea is to use a geometric quantity, which is the overlap between two camera frusta. Retrieval of nearest neighbours with high overlap will improve results of high-accuracy methods that are based on appearance matching such as [31], since there is a stronger probability that a consistent set of feature points will be visible in both images.

Given a pair of images, $\{\boldsymbol{x}, \boldsymbol{y}\}$, with known poses $\{\boldsymbol{M_x}, \boldsymbol{M_y}\}$, and camera internal parameters $\boldsymbol{K}$, the geometry of frusta can be calculated efficiently by sampling a uniform grid of voxels. Based on this, we compute a camera frustum overlap distance ξ according to Algorithm 1. Thus, we can define a frustum-overlap based loss, as follows

$$\mathcal{L}_{frustum} = \{||\phi(\boldsymbol{x}) - \phi(\boldsymbol{y})||_2^2 - \xi\}^2 \tag{1}$$

Intuitively, this loss aims to associate camera frusta overlaps between two frames, with their respective distance in the learnt embedding space.

Some sample pairs of images from random sequences (e.g. taken from the ScanNet Dataset [8]), which are similar to the ones that are used in our optimisation process, are shown in Fig. 3. We can observe that the frustum intersection ratio is a very good proxy for visual image similarity. Note that the number written below each image pair is the frustum overlap ratio ($1 - \xi$), and not the frustum overlap distance (ξ). The results in Fig. 3 are computed with D being 4 meters which is a reasonable selection for indoors scenes. The selection of D is dependent on the scale of the scene since the camera frustum clipping plane is related to the distance of the camera to the nearest object. Thus, if this method is to be applied on outside large-scale scenes, this parameter would need to be adjusted accordingly.

3.2 Pose Regression

While retrieval of nearest neighbours is the most important step in our pipeline, it is also crucial to refining the estimations that are given by the neighbours to improve the final inference stage of the unknown query pose.

To improve the estimation that is given from the retrieved nearest neighbours, we add a shallow neural network on top of the feature network, that is trained for regressing differential camera poses between two neighbouring frames.

The choice of the camera pose representation is very important, but the literature finds no ideal candidate [26]: *unit quaternions* were used in [25, 24], *axis-angle* representations in [44, 33] and *Euler angles* [34, 36].

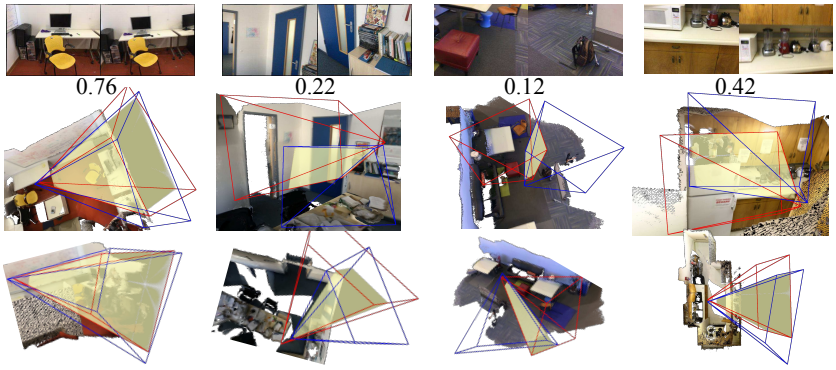


Fig. 3: Samples of our frustum overlap score that is inverted and used as a loss function for learning suitable camera pose descriptors for retrieval. We show pairs of images, together with their respective frustum overlap scores, and two views of the 3D geometry of the scene that lead to the RGB image observations. We can observe that the frustum overlap score is a good indicator of the covisibility of the scenes, and thus a meaningful objective to optimise.

Below, we adopt the matrix representation of rotation with its extension to represent the $\mathbb{SE}(3)$ transformation space similarly to [18]. Specifically, $\boldsymbol{M} = \begin{pmatrix} \boldsymbol{R} & \boldsymbol{t} \\ \boldsymbol{0} & 1 \end{pmatrix} \in \mathbb{SE}(3)$ with $\boldsymbol{R} \in \mathbb{SO}(3)$ and $\boldsymbol{t} \in \mathbb{R}^3$. We adopt the $\mathbb{SE}(3)$ matrix for both transformation amongst different coordinate systems but also for measuring the loss, which shows great convenience in training the network. In addition, since our network directly outputs a camera pose, the validity of the regressed pose is guaranteed, unlike the quaternion method used in [24, 25] where a valid rotation representation for a random $\boldsymbol{q} \in \mathbb{R}^4$ is enforced a-posteriori by normalising the quaternion $\boldsymbol{q}$ to have unit norm.

Our goal is to learn a differential pose regression that is able to use a pair of feature descriptors in order to regress the differential camera poses between them. To that end, we build our pose regression layers on top of the feature layers of RelocNet allowing for a joint forward operation during inference, thus significantly reducing computational time.

The D-dimensional feature descriptors that are extracted from the feature layers of RelocNet, are concatenated into a single feature vector, and are forwarded through a set of fully connected layers which performs a transformation from $\mathbb{R}^D$ to $\mathbb{R}^6$. Afterwards, we can use an exponential map layer to convert this to an element in $\mathbb{SE}(3)$ [18].

Given an input image $\boldsymbol{q}$, we can denote the computed output from the fully connected layers as $\gamma(\phi(\boldsymbol{q}), \phi(\boldsymbol{t})) = (\boldsymbol{\omega}, \boldsymbol{u}) \in \mathbb{R}^6$, where $\phi(\boldsymbol{q})$ and $\phi(\boldsymbol{t})$ are two feature embeddings and $(\boldsymbol{\omega}, \boldsymbol{u})$ is the relative motion from $\phi(\boldsymbol{t})$ to the query image. Our next step is to convert this to a valid $\mathbb{SE}(3)$ pose matrix, which we then use in the training process together with the loss introduced in Eq. 10. By considering the $\mathbb{SE}(3)$ item for the final loss of the training process, the procedure can be optimised for valid camera

poses without needing to normalise quaternions. To convert between $\boldsymbol{se}(3)$ items to $\mathbb{SE}(3)$ we utilise the following two specialised layers:

$exp_{\mathbb{SE}(3)}$ **layer.** we implement an exponential map layer to regress valid camera pose matrices. This accepts a vector $(\boldsymbol{\omega}, \boldsymbol{u}) \in \mathbb{R}^6$ and outputs a valid $\boldsymbol{M} \in \mathbb{SE}(3)$ by using the exponential map from the $\boldsymbol{se}(3)$ element $\boldsymbol{\delta}$ to the $\mathbb{SE}(3)$ element $\boldsymbol{M}$ and can be computed as follows [12]:

$$exp((\boldsymbol{\omega}, \boldsymbol{u})) = \left[\begin{array}{c|c} \boldsymbol{R} & \boldsymbol{V}\boldsymbol{u} \\ \hline \boldsymbol{0} & 1 \end{array}\right] \tag{2}$$

with

$$\theta = \sqrt{\boldsymbol{\omega}^\top \boldsymbol{\omega}} \tag{3}$$

$$\boldsymbol{R} = \boldsymbol{I} + \frac{sin(\theta)}{\theta}[\boldsymbol{\omega}]_\times + \frac{1 - cos(\theta)}{\theta^2}[\boldsymbol{\omega}]_\times^2 \tag{4}$$

$$\boldsymbol{V} = \boldsymbol{I} + \frac{1 - cos(\theta)}{\theta^2}[\boldsymbol{\omega}]_\times + \frac{\theta - sin(\theta)}{\theta^3}[\boldsymbol{\omega}]_\times^2 \tag{5}$$

where $[\boldsymbol{\omega}]_\times$ represents the skew symmetric matrix generator for the vector $\boldsymbol{\omega} \in \mathbb{R}^3$ [12].

Subsequently, we are able to do a forward pass in this layer, using the output of the network $\gamma(\boldsymbol{q}, \boldsymbol{t}) = (\boldsymbol{\omega}, \boldsymbol{u})$, and passing it through as per Eq. 2.

$\log_{\mathbb{SE}(3)}$ **layer.** To return from $\mathbb{SE}(3)$ items to $\boldsymbol{se}(3)$, we implement a logarithmic map layer, which is defined as follows:

$$\log(\left[\begin{array}{c|c} \boldsymbol{R} & \boldsymbol{V}\boldsymbol{u} \\ \hline \boldsymbol{0} & 1 \end{array}\right]) = (\log(\boldsymbol{R}), \boldsymbol{V}^{-1}\boldsymbol{u}) \tag{6}$$

$$\log(\boldsymbol{R}) = \frac{\theta}{2\sin(\theta)}(\boldsymbol{R} - \boldsymbol{R}^T) \tag{7}$$

As suggested by [12], the Taylor expansion of $\frac{\theta}{2\sin(\theta)}$ should be used when the norm of $\boldsymbol{\omega}$ is below the machine precision. However, in our training process, we did not observe elements suffering from this issue.

Joint learning of feature descriptors and poses with a Siamese network As previously discussed, one of the main issues with the recent work on CNN relocalisers is the need to use the global world coordinate system as a training label. This strongly restricts the learning process and thus requires re-training for each new sequence that the system encounters. To address this issue, we instead propose to focus on learning a shallow differential pose regressor, which returns the camera motion between two arbitrary frames of a sequence. In addition, by expanding the training process to pairs of frames, we expand the amount of information, since we can use exponentially more training samples than when training with individual images. We thus design our training process as a Siamese convolutional regressor [10].

For training the Siamese architecture, a pair of images $(\boldsymbol{q}_L, \boldsymbol{q}_R)$ is given as input and the network outputs a single estimate $\tilde{\boldsymbol{M}} \in \mathbb{SE}(3)$. Intuitively, this $\tilde{\boldsymbol{M}}$ represents the differential pose between the two pose matrices. More formally, let $\boldsymbol{M}_{wL}$ represent the

pose of an image $\boldsymbol{q}_L$, and $\boldsymbol{M}_{wR}$ the pose of an image $\boldsymbol{q}_R$, with both poses representing the transformation from the camera coordinate system to the world. The differential transformation matrix that transfers the camera from $R \rightarrow L$ is given by $\boldsymbol{M}_{RL} = \boldsymbol{M}_{wR}^{-1}\boldsymbol{M}_{wL}$.

Assuming we have a set of K training items inside a mini-batch,

$$\{\boldsymbol{q}_L^{(i)}, \boldsymbol{M}_{wL}^{(i)}, \boldsymbol{q}_R^{(i)}, \boldsymbol{M}_{wR}^{(i)}, \boldsymbol{M}_{RL}^{(i)}, \xi_{LR}\}\ i \in [1, K] \tag{8}$$

we train our network with the following loss

$$\mathcal{L} = \alpha\mathcal{L}_{\mathbb{SE}(3)} + \beta\mathcal{L}_{frustum} \tag{9}$$

with

$$\mathcal{L}_{\mathbb{SE}(3)} = \sum_{i=0}^{K} ||log_{\mathbb{SE}(3)}\{\tilde{\boldsymbol{M}}^{(i)^{-1}}(\boldsymbol{M}_{wR}^{(i)-1}\boldsymbol{M}_{wL})\}||_1 \tag{10}$$

which considers the L_1 norm of the $log_{\mathbb{SE}(3)}$ map of the composition of the inverse of the prediction $\tilde{\boldsymbol{M}}$ and the ground truth $\boldsymbol{M}_{wR}^{(i)-1}\boldsymbol{M}_{wL}$. Intuitively, this will become 0 when the $\boldsymbol{M}_{wR}^{(i)-1}\boldsymbol{M}_{wL}$ becomes $\boldsymbol{I}_{4\times4}$ due to the fact that the logarithm of the identity element of $\mathbb{SE}(3)$ is 0. Note that we can extend the above method, to focus on single image based regression, where for each training item $\{\boldsymbol{q}_i, \boldsymbol{M}_i\}$ we infer a pose $\hat{\boldsymbol{M}}_i$, and we instead modify the loss function to optimise $\hat{\boldsymbol{M}}_i^{-1}\boldsymbol{M}_i$. We provide a visual overview of the training stage on Fig. 2 (left).

3.3 Inference Stage

In this section, we discuss our inference framework, starting by using one nearest neighbour (*NN*) for pose estimation, and subsequently using multiple nearest neighbours.

Pose from a Single Nearest Neighbour During inference, we assume that there exists a pool of images in the database $\boldsymbol{q}_{db}^{(i)}$, together with their corresponding poses $\boldsymbol{M}_{db}^{(i)}$for $i \in [0, N_{db}]$. Let s_{NN1} represent the index of the nearest neighbour in the D-dimensional feature space for the query $\boldsymbol{q}_q$, with unknown pose $\boldsymbol{M}_q$.

After computing the estimate $\tilde{\boldsymbol{M}} = \gamma(\boldsymbol{q}_q, \boldsymbol{q}_{db}^{(NN1)})$, we can infer a pose $\tilde{\boldsymbol{M}}_{db}$ for the unknown ground-truth pose $\boldsymbol{M}_{db}$ by a simple matrix multiplication, since $\tilde{\boldsymbol{M}} = \boldsymbol{M}_{db}^{-1}\tilde{\boldsymbol{M}}_q$. We provide a visual overview of the inference stage on Fig. 2 (right).

Pose from Multiple Nearest Neighbours We also briefly discuss a method to infer a prediction from multiple candidates. As shown in Fig. 6, for each pose query we can obtain top *K-NN*, and use each one of them to predict a distinct pose for the query using our differential pose regressor. We aim to aggregate these matrices into a single estimate $\tilde{\boldsymbol{M}}^{(e)}$.

We consider the $(\boldsymbol{\omega}, \boldsymbol{u})$ representation of a pose matrix in $\boldsymbol{se}(3)$ as discussed before, and compute

$$\log(\hat{\boldsymbol{M}}^{(e)}) = \sum_k \beta_k \log(\boldsymbol{M}^{(k)}) + k\log(\boldsymbol{M}^{(e)}) \sum_K \beta_k + k \tag{11}$$

with $\beta_k = \frac{\sqrt{2t\hat{r}-t^2}}{\hat{r}}$ and $\hat{r} = \max(||\log(\boldsymbol{M}^{(e)}) - log(\hat{\boldsymbol{M}}^{(e)})||, t)$, resulting from the robust Huber error norm, with t denoting the outlier threshold, and k the number of nearest neighours that contribute to the estimation $\boldsymbol{M}^{(e)}$. We then use iteratively reweighted least squares, to estimate $\log(\boldsymbol{M}^{(e)})$ and the inliers amongst the set of the k neural network predictions [22, 38]. For our implementation we use $k = 5$ and $t = 0.5$.

3.4 Training Process

We use ResNet18 [19] as a feature extractor, and we run our experiments for the training of the retrieval stage with maximum clipping depth $D = 4m$ and grid step $0.2m$. In addition, to avoid the fact that most pairs in a sequence are not covisible, we limit our selection of pairs to cases where the translation distance is below $0.3m$ and the rotation is below 30°.

We append three fully connected layers of sizes $(512 \rightarrow 512)$, $(512 \rightarrow 256)$ and $(256 \rightarrow 6)$ to reduce the 512 dimensional output of the Siamese output feature layer $\phi(\boldsymbol{x}) - \phi(\boldsymbol{y})$ of the network to a valid element in $\mathbb{R}^6$. This is then fed to the $exp_{\mathbb{SE}(3)}$ layer to produce a valid 4×4 pose matrix. For training, we use Adam [27], with a learning rate of 10^{-4}. We also use weight decay that we set to 10^{-5}. We provide a general visual overview of the training process in Fig. 2 (left). For our joint training loss, we set $a = 0.1$ and $\beta = 0.9$.

4 Results

In this section, we briefly introduce the datasets that are used for evaluating our method, and we then present experiments that show that our feature descriptors are significantly better at relocalisation compared to previous work. In addition, we show that the shallow differential pose regressor is able to perform meaningfully when transferred to a novel dataset, and is able to outperform other methods when trained and tested on the same dataset.

4.1 Evaluation Datasets

We use two datasets to evaluate our methods, namely 7scenes[16], and our new RelocDB which is introduced later in this paper. Training is done primarily on the ScanNet dataset [8].

ScanNet. The ScanNet dataset[8] consists of over $1k$ sequences, with respective ground truth poses. We keep this dataset for training since there do not exist multiple sequences for each scene globally aligned such that they can be used for relocalisation purposes. In addition, the size of the dataset makes it easy to examine the generalisation capabilities of our method.

7Scenes. The 7Scenes dataset consists of 7 scenes each containing multiple sequences that are split into train and test sets. We use the train set to generate our database of stored features, and we treat the images in the test set as the set of unknown queries.

Fig. 4: Sample sequences from our RelocDB dataset.

RelocDB dataset. While 7Scenes has been widely used, is it significantly smaller than ScanNet and other datasets that are suitable for training deep networks. ScanNet aims to address this issue, however, it is not designed for relocalisation. To that end, we introduce a novel dataset, RelocDB that is aimed at being a helpful resource at evaluating retrieval methods in the context of camera relocalisation.

We collected 500 sequences with a Google Tango device, each split into train and test parts. The train and test set are built by moving two times over a similar path, and thus are very similar in terms of size. These sets are aligned to the same global coordinate framework, and thus can be used for relocalisation. In Fig. 4, we show some examples of sequences from our RelocDB dataset.

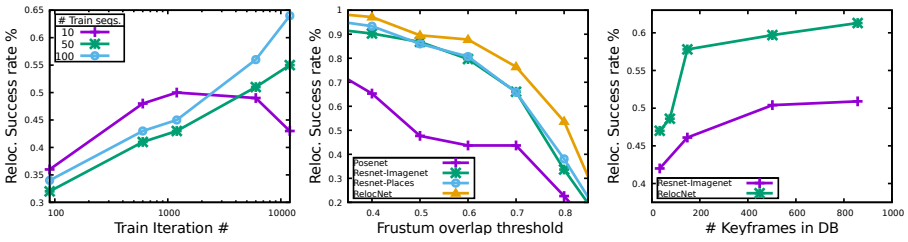
4.2 Frustum Overlap Feature Descriptors

Below we discuss several experiments demonstrating the retrieval performance of our feature learning method. For each of these cases, the frusta descriptors are trained on ScanNet and evaluated on 7Scenes sequences. In all cases, we use *relocalisation success rate* as a performance indicator, which simply counts the percentage of query items that were relocalised from the test set to the saved trained dataset by setting a frustum overlap threshold.

We compare with features extracted from ResNet18[19], VGG [43], PoseNet [25], and a non-learning based method [16]. Fig. 5 (a) indicates that the size of the training set is crucial for the good generalisation of the learnt descriptors for the heads sequence in 7Scenes. It is clear that descriptors that are learnt with a few sequences quickly overfit and are not suitable for retrieval. In Fig. 5 (b) we plot the performance of our learnt descriptor across different frustum overlap thresholds, where we can observe that our method outperforms other methods across all precisions. It is also worth noting, that the features extracted from the penultimate PoseNet layer does not seem to be relevant for relocalisation, presumably due to the fact that they are trained for direct regression, and more importantly are over-fitted to each specific training sequence. To test the effect of the size of a training set that is used as a reference DB of descriptors

in the performance of our method, we increasingly reduce the number of items in the training set, by converting the 1000 training frames to a sparser set of keyframes based on removing redundant items, according to camera motion thresholds of $0.1m, 10°$. Thus, the descriptor for a new frame will be added in the retrieval descriptors pool, only if it presents larger values in both threshold than all of the items already stored. In Fig. 5 (c), we show results in terms of accuracy versus retrieval pool size for our method compared to a standard pre-trained ImageNet retrieval method. We can observe that our descriptor is more relevant across several different keyframes training set sizes. We can also see that our method is able to deal with smaller retrieval pools in a more efficient way.

In Table 1, we show a general comparison between several related methods. As we can observe, our descriptors are very robust and can generalise in a meaningful way between two different datasets. The low performance of the features extracted from PoseNet is also evident here. It is also worth noting that our method can be used instead of other methods in several popular relocalisers and SLAM systems, such as [38], where Ferns [16] are used.



(a) (b) (c)

Fig. 5: (a) Relation of training dataset size and relocalisation performance. We can observe that there is a clear advantage of using more training data for training descriptors relevant to relocalisation (b) Relocalisation success rate in relation with the frustum overlap threshold. Our RelocNet is able to outperform pre-trained methods with significantly more training data, due to the fact that it is trained with a relevant geometric loss. (c) Relation of number of keyframes stored in the database with relocalisation success rate. Our retrieval descriptor shows consistent performance over datasets with different amounts of stored keyframes.

4.3 Pose Regression Experiments

In Table 2 we show the results of the proposed pose regression method, compared to several state-of-the-art CNN based methods for relocalisation. We compare our work with the following methods: PoseNet [25] which uses a weighted quaternion and translation loss, the Bayesian and geometric extensions to PoseNet [23, 24] which uses geometric re-projection error for training, and an approach that extends regression to the

		ResNet18	ResNet18	VGG11	VGG19	PoseNet	Ferns	**RelocNet**
	train set	ImageNet	Places	ImageNet	ImageNet	Cambr. Land.	-	ScanNet
DB	sequence							
7scenes	heads	48.6 %	46.6 %	37.7 %	39.8 %	29.1%	30.63%	**70.33%**
	fire	67.3 %	73.1 %	64.9 %	66.8%	33.70%	37.03%	**79.01%**
	redkitchen	65.0%	62.6%	64.8%	61.1%	30.9%	40.47%	**73.42%**
	chess	71.25%	69.50%	67.90%	74.90%	18.6%	51.73%	**78.95%**
	stairs	32.5%	54.6%	42.7%	41.0%	7.8%	28.16%	**62.77%**
	pumpkin	73.1%	68.8%	69.2%	69.8%	12.2 %	52.17%	**79.25%**
	office	69.0%	69.3%	64.0%	57.5%	10.3%	47.34 %	**72.41%**
RelocDB	caution	78.1%	78.1%	72.1%	70.9%	30.1%	61.8%	**83.6%**
	desk	59.6%	61.5%	59.6%	61.5%	31.3%	47.3%	**68.4%**
	lecture	66.6%	62.0%	55.1%	64.3%	29.40%	40.2%	**70.1%**
	meetingroom	57.2%	56.7%	54.3%	53.5%	12.81%	36.4%	**62.5%**
	posters	62.6%	67.3%	58.1%	62.6%	39.94%	49.3%	**74.3%**
	printer	67.0%	70.8%	63.2%	70.8%	27.69%	31.0%	**72.1%**

Table 1: Nearest neighbour matching success rate using a brute force approach. We show the success rate of relocalising when using a frustum overlap threshold of 0.7 across 7Scenes and sequences from our new RelocDB. We can observe that our feature descriptors significantly outperform all other methods in terms of relocalisation success rate, by a significant margin.

temporal domain using recurrent neural networks [46]. We can observe that even by using the descriptors and the pose regressors learnt on ScanNet, we are able to perform on par with methods that are trained and tested on the same sequences. This is a significant result as it shows the potential of large-scale training in relocalisation. In addition, we can observe that when we apply our relocalisation training framework by training and testing on the same sequence as the other methods do, we are able to outperform several related methods.

4.4 Fusing Multiple Nearest Neighbours

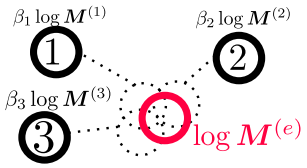
In Fig. 6 we show results comparing the single NN performance with the fusing method from Eq. 11. We can observe that in most cases, fusing from multiple NNs slightly improves the performance. The fact that the improvement is not significant and consistent is potentially attributed to the way the nearest neighbours are extracted from the dataset, which might lead to significantly similar candidates. One possible solution to this, would be to actively enforce some notion of dissimilarity between the retrieved nearest neighbours, therefore ensuring that the fusion operates on a more diverse set of proposals.

4.5 Qualitative Examples

In the top two rows of Fig. 7, we show examples of a synthetic view of the global scene model using the predicted pose from the first nearest neighbour, while the bottom row

Scene	PoseNet (β weight) [25]	Bayesian PoseNet [23]	PoseNet Spatial LSTM [46]	PoseNet Geometric [24]	**RelocNet** ScanNet	**RelocNet** 7scenes
Chess	0.32m, 6.60°	0.37m, 7.24°	0.24m, 5.77°	0.13m, 4.48°	0.21m, 10.9°	**0.12m**, **4.14°**
Fire	0.47m, 14.0°	0.43m, 13.7°	0.34m, 11.9°	0.27m, 11.3°	0.32m, 11.8°	**0.26m**, **10.4°**
Heads	0.30m, 12.2°	0.31m, 12.0°	0.21m, 13.7°	0.17m, 13.0°	0.15m, 13.4°	**0.14m**, **10.5°**
Office	0.48m, 7.24°	0.48m, 8.04°	0.30m, 8.08°	0.19m, 5.55°	0.31m,10.3°	**0.18m**, **5.32°**
Pumpkin	0.49m, 8.12°	0.61m, 7.08°	0.33m, 7.00°	**0.26m**, 4.75°	0.40m,10.9°	**0.26m**, **4.17°**
Red Kitchen	0.58m, 8.34°	0.58m, 7.54°	0.37m, 8.83°	**0.23m**, 5.35°	0.33m,10.3°	**0.23m**, **5.08°**
Stairs	0.48m, 13.1°	0.48m, 13.1°	0.40m, 13.7°	0.35m, 12.4°	0.33m, 11.4°	**0.28m**, **7.53°**

Table 2: Median localisation errors in the *7Scenes* [42] dataset. We can observe that we can outperform the original version of PoseNet even by training and testing on separate datasets. This indicates the potential of our method in terms of transferability between datasets. In addition, we can outperform other methods when we train and test our method on the same datasets. Finally, it is also worth noting that the performance boost from using temporal information (LSTM) is smaller than the one given by using our method.



scene	Diff. training NN	Diff. training kNN
chess	0.12m, 4.14 °	0.12m, 3.95 °
heads	0.14m, 10.5 °	0.13m, 10.5 °
fire	0.26m, 10.4 °	0.25m, 10.1 °
stairs	0.28m, 7.53 °	0.27m, 7.31°

Fig. 6: Effect of fusing multiple nearest neighbours. We can observe that we are able to improve performance over single nearest neighbour, by incorporating pose information from multiple nearest neighbours.

shows the query image whose pose we are aiming to infer. Note that for this experiment, we use the high accuracy per-database trained variant of our network. From the figure, we can see that in most of the cases the predicted poses are well aligned with the query image (first 5 columns). We also show some failure cases for our method (last 3 columns). The failure cases might be characterised by the limited overlap between the query and training frames, something that is an inherent disadvantage of our method.

In Fig. 7 (bottom), we show typical cases of the camera poses of the nearest neighbours (red) selected by the feature network, as well as the estimated query pose for each nearest neighbour (cyan). Note that these results are sample test images when using the network that is trained on the non-overlapping train set. In addition, we show the ground truth query pose which is indicated by the blue frustum. Surprisingly, we see that the inferred poses are significantly stable even for cases where the nearest neighbours that are retrieved are noisy (eg 1^{st} and 2^{nd} columns). In addition, we can observe that in the majority of the cases, the predicted poses are significantly closer to the ground truth than the retrieved poses of the nearest neighbours. Lastly, we show a failure case (last column) where the system was not able to recover, due to the fact that the nearest neighbour is remarkably far from the ground truth, something that is likely due to the limited overlap between train and test poses.

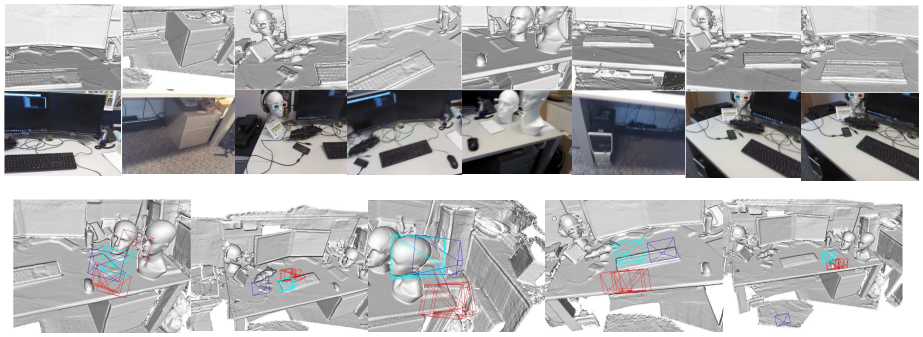


Fig. 7: (top 2 rows) Examples of the global map rendered using our predicted pose (top 1^{st} row) compared to the actual ground truth view (top 2^{nd} row) (bottom) Examples of how our network "corrects" the poses of the nearest neighbours (red frusta) to produce novel camera poses (cyan frusta). We can observe that in most cases, the corrected poses are significantly closer to the ground truth (blue frustum).

5 Conclusions

We have presented a method to train a network using frustum overlaps that is able to retrieve nearest pose neighbours with high accuracy. We show experimental results that indicate that the proposed method is able to outperform previous works, and is able to generalise in a meaningful way to novel datasets. Finally, we illustrate that our system is able to predict reasonably accurate candidate poses, even when the retrieved nearest neighbours are noisy. Lastly, we introduce a novel dataset specifically aimed at relocalisation methods, that we make public.

For future work, we aim to investigate more advanced methods of training the retrieval network, together with novel ways of fusing multiple predicted poses. Significant progress can also be made in the differential regression stage to boost the good performance of our fine-grained camera pose descriptors. In addition, an interesting extension to our work would be to address the scene scaling issue, using some online estimation of the scene, and adjusting the learning method accordingly.

Acknowledgments

We gratefully acknowledge the Huawei Innovation Research Program (HIRP) FLAGSHIP grant and the European Commission Project Multiple-actOrs Virtual Empathic CARegiver for the Elder (MoveCare) for financially supporting the authors for this work.

References

1. S. K. Arun, T. S. Huang, and S. D. Blostein. Least-Squares Fitting of Two 3-D Point Sets. *IEEE Trans. Pattern Anal. Machine Intell. (PAMI)*, 9:698–700, 1987.

2. S. H. B, V. Lepetit, N. Rajkumar, and K. Konolige. Going Further With Point Pair Features. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 834–848, 2016.
3. V. Balntas, A. Doumanoglou, C. Sahin, J. Sock, R. Kouskouridas, and T.-K. Kim. Pose Guided RGB-D Feature Learning for 3D Object Pose Estimation. In *Proceedings of Intl. Conf. on Computer Vision (ICCV)*, 2017.
4. C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. D. Reid, and J. J. Leonard. Simultaneous Localization And Mapping: Present, Future, and the Robust-Perception Age. *IEEE Trans. on Robotics (ToR)*, pages 1–27, 2016.
5. T. Cavallari, S. Golodetz, N. A. Lord, J. Valentin, L. Di Stefano, and P. H. Torr. On-the-fly Adaptation of Regression Forests for Online Camera Relocalisation. In *Proceedings of IEEE Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017.
6. D. Chekhlov, M. Pupilli, W. Mayol, and A. Calway. Robust Real-Time Visual SLAM Using Scale Prediction and Exemplar Based Feature Description. In *Proceedings of IEEE Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2007.
7. R. Clark, S. Wang, A. Markham, N. Trigoni, and H. Wen. 6-DoF Video-Clip Relocalization. In *Proceedings of IEEE Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017.
8. A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner. ScanNet: Richly-annotated 3D Reconstructions of Indoor Scenes. In *Proceedings of IEEE Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2017.
9. J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *Proceedings of IEEE Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2009.
10. A. Doumanoglou, V. Balntas, R. Kouskouridas, and T. Kim. Siamese Regression Networks with Efficient mid-level Feature Extraction for 3D Object Pose Estimation. *arXiv preprint arXiv:1607.02257*, 2016.
11. B. Drost, M. Ulrich, N. Navab, and S. Ilic. Model Globally, Match Locally: Efficient and Robust 3D Object Recognition. In *Proceedings of IEEE Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 998–1005, 2010.
12. E. Eade. Lie Groups for 2D and 3D Transformations. Technical report, University of Cambridge, 2017.
13. J. Engel, T. Schops, and D. Cremers. LSD-SLAM: Large-Scale Direct Monocular SLAM. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 1–16, 2014.
14. D. Galvez-Lopez and J. D. Tardos. Bags of Binary Words for Fast Place Recognition in Image Sequences. In *Proceedings of IEEE Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, volume 28, pages 1188–1197, 2012.
15. A. Gee and W. Mayol-Cuevas. 6D Relocalisation for RGBD Cameras Using Synthetic View Regression. In *Proceedings of British Machine Vision Conference (BMVC)*, 2012.
16. B. Glocker, S. Izadi, J. Shotton, and A. Criminisi. Real-Time RGB-D Camera Relocalization. In *Proceedings of IEEE/ACM Intl. Symposium on Mixed and Augmented Reality (ISMAR)*, volume 21, pages 571–583, 2013.
17. A. Guzman-Rivera, P. Kohli, B. Glocker, J. Shotton, T. Sharp, A. Fitzgibbon, and S. Izadi. Multi-Output Learning for Camera Relocalization. In *Proceedings of IEEE Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014.
18. A. Handa, M. Bloesch, V. Patraucean, S. Stent, J. McCormac, and A. Davison. gvnn: Neural Network Library for Geometric Computer Vision. In *Proceedings of the European Conference on Computer Vision Workshops*, 2016.
19. K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In *Proceedings of IEEE Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

20. B. K. Horn. Closed-form Solution of Absolute Orientation Using Unit Quaternions. *Journal of the Optical Society of America A*, 6:422, 1987.
21. A. S. Huang, A. Bachrach, P. Henry, M. Krainn, D. Maturana, D. Fox, and N. Roy. Visual Odometry and Mapping for Autonomous Flight Using an RGB-D Camera. In *Proceedings of Intl. Symposium on Robotics Research (ISRR)*, 2011.
22. O. Kähler, V. A. Prisacariu, and D. W. Murray. Real-time Large-scale Dense 3D Reconstruction with Loop Closure. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 500–516. Springer, 2016.
23. A. Kendall and R. Cipolla. Modelling Uncertainty in Deep Learning for Camera Relocalization. In *Proceedings of IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 4762–4769, 2016.
24. A. Kendall and R. Cipolla. Geometric Loss Functions for Camera Pose Regression with Deep Learning. In *Proceedings of IEEE Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 6555–6564, 2017.
25. A. Kendall, M. Grimes, and R. Cipolla. PoseNet: A Convolutional Network for Real-Time 6-DOF Camera Relocalization. In *Proceedings of Intl. Conf. on Computer Vision (ICCV)*, pages 2938–2946, 2015.
26. H. Kengo, T. Satoko, T. Toru, R. Bisser, K. Kazufumi, and A. Toshiyuki. Comparison of 3 DOF Pose Representations for Pose Estimations. *Korea-Japan Joint Workshop on Frontiers of Computer Vision (FCV)*, 2010.
27. D. Kingma and J. Ba. Adam: A Method For Stochastic Optimization. In *Proceedings of Intl. Conf. on Learning Representations (ICLR)*, 2015.
28. Z. Laskar, I. Melekhov, S. Kalia, and J. Kannala. Camera Relocalization by Computing Pairwise Relative Poses Using Convolutional Neural Network. *arXiv preprint arXiv:1707.09733*, 2017.
29. V. Lepetit, F. Moreno-Noguer, and P. Fua. EPnP: An Accurate O(n) Solution to the PnP Problem. *Intl. Journal of Computer Vision (IJCV)*, 81:155–166, 2009.
30. S. Li and A. Calway. RGBD Relocalisation Using Pairwise Geometry and Concise Key Point Sets. In *Proceedings of IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2015.
31. S. Li and A. Calway. Absolute Pose Estimation Using Multiple Forms of Correspondences from RGB-D Frames. In *Proceedings of IEEE Intl. Conf. on Robotics and Automation (ICRA)*, pages 4756–4761, 2016.
32. S. Li, C. Xu, and M. Xie. A Robust O(n) Solution to the Perspective-n-Point Problem. *IEEE Trans. Pattern Anal. Machine Intell. (PAMI)*, 34:1444–1450, 2012.
33. S. Mahendran, H. Ali, and R. Vidal. 3D Pose Regression Using Convolutional Neural Networks. In *Proceedings of IEEE Intl. Conf. on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 494–495, 2017.
34. F. Massa, R. Marlet, and M. Aubry. Crafting a Multi-task CNN for Viewpoint Estimation. In *Proceedings of British Machine Vision Conference (BMVC)*, pages 91.1–91.12, 2016.
35. R. J. Micheals and T. E. Boult. On the Robustness of Absolute Orientation. In *Proceedings of IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2000.
36. K. Moo Yi, Y. Verdie, P. Fua, and V. Lepetit. Learning to Assign Orientations to Feature Points. In *Proceedings of IEEE Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 107–116, 2016.
37. R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos. ORB-SLAM : A Versatile and Accurate Monocular SLAM System. *IEEE Trans. on Robotics (ToR)*, 2015.
38. V. A. Prisacariu, O. Kähler, S. Golodetz, M. Sapienza, T. Cavallari, P. H. Torr, and D. W. Murray. InfiniTAM v3: A Framework for Large-Scale 3D Reconstruction with Loop Closure. *arXiv preprint arXiv:1708.00783*, 2017.
39. S. Saeedi, M. Trentini, H. Li, and M. Seto. Multiple-robot Simultaneous Localization and Mapping - A Review. *Journal of Field Robotics*, 2015.

40. R. F. Salas-Moreno, R. a. Newcombe, H. Strasdat, P. H. Kelly, and A. J. Davison. SLAM++: Simultaneous Localisation and Mapping at the Level of Objects. In *Proceedings of IEEE Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1352–1359, 2013.
41. U. Shinji. Least-squares Estimation of Transformation Parameters between Two Point Patterns. *IEEE Trans. Pattern Anal. Machine Intell. (PAMI)*, 1991.
42. J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi, and A. Fitzgibbon. Scene Coordinate Regression Forests for Camera Relocalization in RGB-D Images. In *Proceedings of IEEE Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2930–2937, 2013.
43. K. Simonyan and A. Zisserman. Very Deep Convolutional Networks for Large-scale Image Recognition. *arXiv preprint arXiv:1409.1556*, 2014.
44. 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 *Proceedings of IEEE Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 5622–5631, 2017.
45. J. Valentin, A. Fitzgibbon, M. Nießner, J. Shotton, and P. Torr. Exploiting Uncertainty in Regression Forests for Accurate Camera Relocalization. In *Proceedings of IEEE Intl. Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2015.
46. F. Walch, C. Hazirbas, L. Leal-Taixé, T. Sattler, S. Hilsenbeck, and D. Cremers. Image-based Localization with Spatial LSTMs. *arXiv preprint arXiv:1611.07890*, 2016.
47. Y. Zheng, Y. Kuang, S. Sugimoto, K. Astrom, and M. Okutomi. Revisiting the PnP Problem: A Fast, General and Optimal Solution. In *Proceedings of Intl. Conf. on Computer Vision (ICCV)*, pages 2344–2351, 2013.
48. B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba. Places: A 10 Million Image Database for Scene Recognition. *IEEE Trans. Pattern Anal. Machine Intell. (PAMI)*, 2017.