IEEE Copyright Notice

Pre-print of article that will appear at the **2019 International Conference on Computer Vision (ICCV 2019) - 2nd Workshop on Deep Learning for Visual SLAM.**

Adversarial Networks for Camera Pose Regression and Refinement

Mai Bui[1,*], Christoph Baur[1,*], Nassir Navab[1,3], Slobodan Ilic[1,2,†] and Shadi Albarqouni[1,†]

[1] Technical University of Munich, Germany
[2] Siemens AG, Munich, Germany
[3] Johns Hopkins University, Baltimore, USA

Abstract

Despite recent advances on the topic of direct camera pose regression using neural networks, accurately estimating the camera pose of a single RGB image still remains a challenging task. To address this problem, we introduce a novel framework based, in its core, on the idea of implicitly learning the joint distribution of RGB images and their corresponding camera poses using a discriminator network and adversarial learning. Our method allows not only to regress the camera pose from a single image, however, also offers a solely RGB-based solution for camera pose refinement using the discriminator network. Further, we show that our method can effectively be used to optimize the predicted camera poses and thus improve the localization accuracy. To this end, we validate our proposed method on the publicly available 7-Scenes dataset improving upon the results of direct camera pose regression methods.

1. Introduction

Camera re-localization is an important topic in computer vision applications such as simultaneous localization and mapping (SLAM) [29, 44] in case of tracking failure, augmented reality [26] or in robotics for navigation [4]. Current methods have focused on computing the camera pose given 2D-3D correspondences between the input image and a 3D model of the scene, in essence predicting the camera pose by solving the perspective-n-point problem. Most often correspondences are computed using for example SIFT [19] features or implicitly learned using regression forests [36, 41] as well as deep learning methods [5, 6, 8]. On the other hand, direct camera pose estimation approaches have been developed, that regress the camera pose using convolutional neural networks (CNNs), providing a very fast solution to solve this task and, in contrast to previous methods, solely relying on RGB information [22, 20, 21, 42].

---

† S. Ilic and S. Albarqouni are joint senior authors
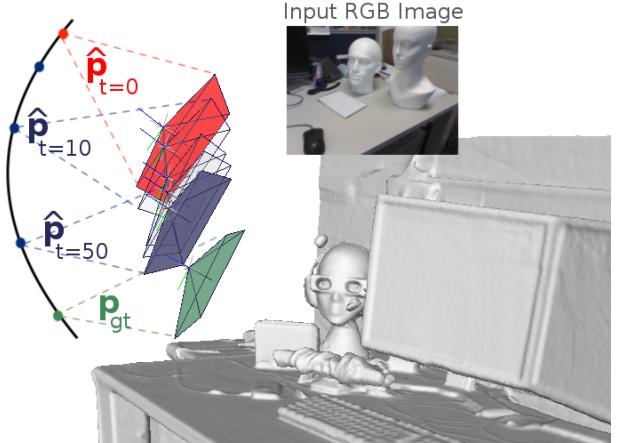* M. Bui and C. Baur contributed equally to this work

Figure 1: Given an RGB input image, our method regresses a camera pose estimate $\hat{\mathbf{p}}_{t=0}$ (red) in reference to a known scene. By incorporating adversarial training and following pose refinement, the regressed pose is updated and pushed further towards the ground truth pose $\mathbf{p}_{gt}$ (green), resulting in the final prediction $\hat{\mathbf{p}}_{t=50}$ (blue).

This advantage makes such methods easily applicable in indoor as well as outdoor scenarios without requiring a 3D model or depth information. However, despite recent advances of these methods, accurately regressing the camera pose of a corresponding RGB image still remains a difficult task, especially if very little training data is available. The performance of correspondence-based methods, in comparison, can most often be accounted to an iterative pose refinement step using RANSAC, that due to the absence of a 3D model has not yet been investigated in the context of direct camera pose regression frameworks. Therefore, in this paper we make an attempt at providing a deep learning based solution for RGB-based camera pose refinement.

For this aim, we draw inspiration from the framework of Generative Adversarial Networks (GANs) [16], which has recently shown great success in improving the performance of deep neural networks trained for tasks such as object detection [43], human pose estimation [12, 45] or realistic image composition [25]. Such GANs consist of two networks, a generator that captures the underlying data distribution and a discriminator that estimates the probability of a sample coming from the actual distribution or the generated one, i.e. can tell the real distribution and distribution of generated data apart. During training, the two networks are in competition with each other as the generator tries to better mimic the ground truth data distribution such that it becomes more and more difficult for the discriminator to correctly classify a sample representation. More precisely, in every training step, the generator is updated in a way such that it is more likely to fool the discriminator.

In order to improve direct camera pose regression models, and to better model the connection between RGB images and their camera poses, we first follow the training procedure of GANs and combine a camera pose regression network and a pose discriminator network that learns to distinguish between accurate real and potentially erroneously regressed poses and the input RGB image. This way, we attempt to implicitly model the joint distribution between an RGB image and the corresponding camera pose capturing the geometric mapping between the two in the discriminator network. Once learned, we show how the information contained in the discriminator network can be leveraged to further refine the predicted poses during inference. To summarize our contributions, we propose a novel framework for camera pose regression, that 1) includes the effect of adversarial learning in the aforementioned frameworks and 2) introduces a solely RGB-based solution for refining the resulting camera poses giving an additional boost in performance. An example result of our method is shown in Figure 1, where we visualize regressed and optimized camera poses in comparison to the ground truth pose.

2. Related Work

Methods working on the topic of camera pose estimation can mainly be divided into three groups: correspondence-based, image-retrieval-based and direct pose regression approaches.

Correspondence-Based. In classical SLAM or structure from motion scenarios the camera is tracked in an unknown environment and a corresponding sparse 3D map or reconstruction of the environment is built. Each 3D point in the map has a corresponding image feature descriptor associated to it. Therefore, the main component of these methods is the detection of key-points in a query image and feature extraction, e.g. SIFT features, at these respective points.

2D to 3D point correspondences between the image and the 3D model can then be established using feature descriptor matching. Finally, given these correspondences, the camera pose can be computed by solving the perspective-n-point problem. However, despite usually providing good camera localization accuracy, these methods can easily fail in case of texture-less surfaces and require efficient feature matching techniques to achieve reasonable computational times for camera re-localization applications. For this purpose, Sattler et al. [32, 33, 34] propose an optimized prioritization scheme based on vocabulary-based quantization for efficient feature matching. Additionally, by using co-visibility constraints or semantic consistency checks [39], wrong matches can be removed, which further improves the methods accuracy. In contrast, Schmidt et al. [35] focus on optimizing extracted features used for correspondence matching. Here, a deep learning method is applied and a neural network is trained on a contrastive loss function, pushing features of pixels to be similar only if they correspond to the same 3D point. Implicitly giving a mapping between image pixels and 3D points, Shotton et al. [36] train a regression forest on RGB and depth features extracted at pixel locations to estimate the corresponding scene coordinate directly. Further extensions and analysis of this method have been proposed, including uncertainty of the forests predictions [41], online adaption of the regression forest [11], ensemble prediction [17], backtracking schemes [28] and a comparison to neural networks [27]. Switching from regression forests to convolutional neural networks, Brachmann et al. [5, 6] propose an end-to-end trainable pipeline, consisting of a scene coordinate regression and a pose hypothesis scoring CNN, connected by a differentiable version of RANSAC, which they call DSAC. These methods have shown remarkable results in retrieving accurate camera poses. They, however, usually require depth information or a 3D model.

Image-Retrieval-Based. In contrast to correspondence-based methods, image retrieval methods focus on computing a lower dimensional representation of a full query image, which can then efficiently be matched against a database of images with corresponding camera poses. Finding the nearest neighbor according to the resulting features will in this case also retrieve the closest camera pose, but therefore also restricts the search space to the camera poses contained in the database, especially if RGB images are the only source of information available. Glocker et al. [15] rely on a fern-based encoding approach, which computes a binary encoding for each frame and thus enables fast similarity comparisons based on the Hamming distance. Relja et al. [1] construct a new feature aggregation layer, inspired by VLAD [19], which can be included in any existing convolutional neural network and shows great capa-
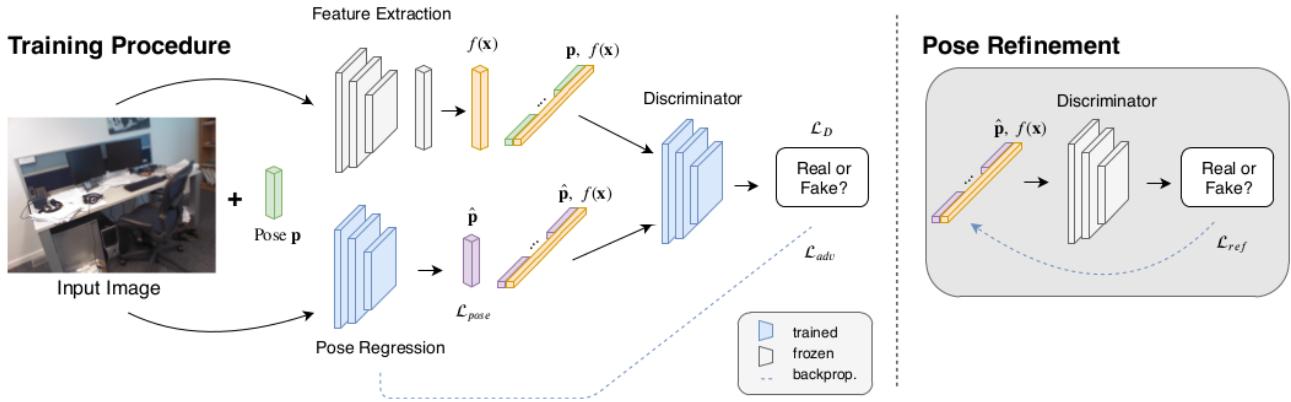
Figure 2: Given an RGB image, a corresponding camera pose is estimated with a pose regression network. Alongside the estimated pose, a feature representation of the corresponding image is extracted and used to train a discriminator network. This network is trained to distinguish between ground truth and regressed poses considering the input image and can then be leveraged to refine the regressed camera pose.

bilities in aiding image retrieval tasks in the context of camera pose estimation. Additionally, Taira et al. [38] propose to learn dense features using a convolutional neural network for camera pose estimation. After retrieval of nearest neighbor database images, dense features at different layers of the network are used to find 2D-3D correspondences, given that depth information is available for the database images, from which the pose can be computed. Additionally, the estimated pose is verified by comparing the query image to the synthesized view obtained using the 3D model and retrieved camera pose.

Direct Pose Regression. Very recently, direct camera pose regression approaches have emerged, mainly using CNNs to estimate the rotation, most often represented as quaternions, and position of the camera given a single RGB image as input. Starting with the introduction of PoseNet [22], Kendall et al. presented a computationally very fast solution for solving the camera pose estimation problem relying solely on RGB information and also showing great capabilities when applied on large-scale scenes. This, on the other hand, came at a large drop in general accuracy compared to earlier state-of-the-art methods. Thus, several extensions and modifications of this method have been proposed, including uncertainty estimation [20, 21, 10], LSTM units [42], frame-to-frame information [2, 24, 13, 7] and previous pose fusion [40, 30]. The latter, however, more closely resembles a camera tracking scenario rather than re-localization as the method relies on pose information of the previous frame.

Since in this work, we want to explore re-localization methods utilizing RGB information only, we build on top of recent research on direct camera pose regression meth-

ods. However, we additionally attempt to model the connection between an RGB image and its camera pose implicitly, rather than trying to simply learn this mapping directly. For this purpose, we show the advantage that leveraging an adversarial network can have on such methods. To the best of our knowledge, we are the first to investigate adversarial learning in the context of camera pose estimation. Therefore, we propose a novel framework based on a camera pose regression network and a discriminator network that, given a regressed pose and the RGB input image, learns to distinguish between regressed and ground truth poses. Further, once the model has learned a representation of this connection, as our main contribution, we show how the trained model can be used for camera pose refinement. By leveraging the learned information encoded in the discriminator network, the localization accuracy can be improved beyond the one of a simple camera pose regression network.

3. Methodology

Following previous camera pose regression approaches, we attempt to train a convolutional neural network, hereby referred to as the pose regressor, to learn the mapping $\Omega : \mathbf{x} \rightarrow \mathbf{p}$ between an input image $\mathbf{x}$ and a camera pose $\mathbf{p}$.

However, we additionally attempt to learn the distribution of camera poses and their respective RGB images captured by the camera. More precisely, we train a pose discriminator network to distinguish between regressed and ground truth pose with respect to the input image. The pose regressor and discriminator are trained in an alternating manner, where the pose regressors goal is to fool the discriminator, such that it can not clearly distinguish between regressed and real camera poses anymore. Finally, once the discriminator has learned the geometric mapping between

an input image and a camera pose, the information captured by the discriminator can be leveraged to update and refine the regressed camera pose. By freezing the discriminator networks weights and optimizing solely the regressed camera pose, we aim at pushing the regressed pose closer towards the manifold of real poses to ultimately better fit the input image. An overview of our method can be seen in Figure 2.

3.1. Camera Pose Regression

Given an RGB image $\mathbf{x} \in \mathbb{R}^{h \times w \times 3}$, our objective is to predict the camera pose $\mathbf{p} = [\mathbf{q}, \mathbf{t}]$ given as orientation, represented as vector $\mathbf{q}$, and translation $\mathbf{t} \in \mathbb{R}^3$. For this aim, a CNN, is trained on the following loss function

$$\mathcal{L}_{pose} = \|\mathbf{t} - \hat{\mathbf{t}}\| e^{-\beta} + \beta + \|\mathbf{q} - \hat{\mathbf{q}}\| e^{-\alpha} + \alpha, \quad (1)$$

where $\hat{\mathbf{t}}$ and $\hat{\mathbf{q}}$ represent the predicted translation and rotation, respectively, β and α are trainable parameters to balance both distances, and $\| \cdot \|$ is chosen to be the ℓ_1 norm. Readers are referred to [21] for further details about the loss function, and its derivation.

The parameterization used to regress the rotational component of an object or a camera pose has been extensively addressed in many literature [7, 14]. In this work, first, we choose to evaluate our method on the representation of quaternions, which is already well established in image-based localization. Here, a quaternion can be described as $\mathbf{q} = [w, \mathbf{u}] \in \mathbb{R}^4$ where w is a real valued scalar and $\mathbf{u} \in \mathbb{R}^3$. To ensure that the resulting quaternions lie on the unit sphere, they are normalized during the training. As shown in [21], no additional constraints are enforced while training the pose regression network, as the resulting quaternions become sufficiently close to the ground truth so that there is no significant difference in ℓ_1 norm and spherical distance. Second, we use the logarithm of a unit quaternion, which is computed as

$$\mathbf{q}_{log} = \log \mathbf{q} = \begin{cases} \frac{\mathbf{u}}{\|\mathbf{u}\|} \arccos(w), & \text{if } \|\mathbf{u}\| \neq 0 \\ \mathbf{0}, & \text{otherwise} \end{cases}, \quad (2)$$

and has the advantages of not being over-parameterized. Further, it relaxes the need of normalization during the training. The the unit quaternion can be recovered by $\mathbf{q} = [\cos(\|\mathbf{q}_{log}\|), \frac{\mathbf{q}_{log}}{\|\mathbf{q}_{log}\|} sin(\|\mathbf{q}_{log}\|)]$.

3.2. Discriminator

Both the regressed poses $\hat{\mathbf{p}}$, and the ground-truth poses $\mathbf{p}$, and a lower dimensional representation, $f(\mathbf{x})$, of the corresponding input images, form "fake" and "real" examples, respectively, used to train the discriminator network. The aim of this network is to minimize the following loss function defined as

$$\mathcal{L}_D = \sigma(\{f(\mathbf{x}), \mathbf{p}\}, c_{real}) + \sigma(\{f(\mathbf{x}), \hat{\mathbf{p}}\}, c_{fake}), \quad (3)$$

where $\sigma(\cdot, \cdot)$ is the binary cross-entropy loss, c_{real} and c_{fake} are set to 1 and 0, respectively. Therefore, the discriminator models the conditional distribution $P(y | \mathbf{p}, f(\mathbf{x}))$ of $y \in \{c_{real}, c_{fake}\}$ conditioned on the pose $\mathbf{p}$ and image features $f(\mathbf{x})$, and thus implicitly captures the joint distribution of $\mathbf{p}$ and $\mathbf{x}$. Our framework is, in fact, inspired by GANs to ensure that the geometric mapping between camera poses and the corresponding RGB images are exploited in the network, however differs from the original GAN framework as our pose regression network is purely discriminative.

3.3. Feature Extraction

A pre-trained network architecture on ImageNet [31], see Section 4.1.3, is used to extract a feature representation $f(\mathbf{x})$ given an RGB input image. The weights of the network are frozen during the training, as its purpose is mainly to provide the discriminator with a lower dimensional representation of the image. Given the fact that most of the state-of-the-art network architectures produce a rather high dimensional feature representation (compared to the six or seven dimensional camera pose vector), and inspired by the concept of dimensionality reduction, we apply a linear mapping to better balance the dimensionality between feature representation and camera pose. To easily integrate this linear mapping to the network architecture, we simply add one additional fully-connected layer, without bias or activation function, right after the last layer, and keep its weights frozen during training. This way, the discriminator is discouraged to adapt the extracted features during training and solely base its decision on the features themselves. The camera pose vector is then copied, to fit the dimensionality of the extracted feature representation, and concatenated with said representation to form a feature map that is used as the input to the discriminator network. Intuitively we would want the discriminator to learn the connection between RGB images and corresponding camera poses. Therefore, such that the network is discouraged to solely focus on the information provided by either one, the design choices described above were made. However, in addition we have experimented with fine-tuning the feature extraction network as well as only fine-tuning individual layers. Both resulted in worse performance.

3.4. Adversarial Learning

Following the training procedure introduced for generative adversarial networks, we alternate between training the camera pose regressor and the discriminator network, updating the regressor on

$$\mathcal{L}_G = \mathcal{L}_{pose} + \lambda \underbrace{\sigma(\{f(\mathbf{x}), \hat{\mathbf{p}}\}, c_{real})}_{\mathcal{L}_{adv}}, \quad (4)$$

such that the network learns to predict more and more realistic poses and thus eventually is able to fool the discriminator. Here, the parameter λ balances the influence of the adversarial loss on the pose regressor.

3.5. Pose Refinement

Once the model is trained and the discriminator is successfully "fooled", meaning it can not distinguish properly between regressed and ground truth poses with respect to the input image, the discriminator network can be used during testing to refine the regressed camera poses. For this aim, the test image is fed to the pose regression network to obtain an initial pose estimate. Then, the predicted pose together with the extracted feature representation of the image is used as input to the discriminator. In succession, however, the weights of the discriminator are frozen, and the initially regressed pose $\hat{\mathbf{p}}$ for the image $\mathbf{x}$ is updated iteratively by minimizing the loss function as

$$\mathcal{L}_{ref} = \sigma(\{f(\mathbf{x}), \hat{\mathbf{p}}\}, c), \tag{5}$$

where the class label c is set to 0.5. This stems from the fact, that at the end of training, the discriminator will not be able to distinguish between regressed and ground truth camera pose anymore, thus predicting values close to 0.5 in both cases. Intuitively, this amounts to moving along the manifold towards a region where the discriminator reliably confuses real and regresses poses. Therefore, any predicted pose of an unseen query image should be pushed towards this manifold. As the gradients coming from the discriminator do not necessarily follow a geometrically meaningful direction, in case of using the quaternion representation, we restrict the quaternion update, so that its movement along the unit sphere is ensured [9, 3]. Thus, the update for one iteration is described by

$$\mathbf{q}_t = \mathbf{q}_{t-1} \cos(\gamma l) + \frac{\mathbf{v}}{\gamma} \sin(\gamma l), \tag{6}$$

with $\gamma = \|\mathbf{v}\|_2$, l being the step size, and $\mathbf{v} \in \mathbb{R}^4$ being the projection of the quaternion gradient $\nabla \mathbf{q}$ into the tangent space, given as

$$\mathbf{v} = (I - \nabla \mathbf{q} \nabla \mathbf{q}^T) \nabla \mathbf{q}, \tag{7}$$

where $I \in \mathbb{R}^{4 \times 4}$ is the identity matrix. To further ensure that the resulting poses are valid, the updated quaternion is normalized after each iteration. However, no such constraints have to be be enforced to update the translational component of the camera pose. Though, for simplicity, it is updated with the same step size l.

3.6. Training Procedure

As a first step, the pose regression network is trained for a few epochs to initially give reasonable poses, before including the adversarial loss in the training procedure, where

Table 1: Effect of adversarial training and pose refinement on the camera pose accuracy, evaluated on the *Heads* scene. Median rotation and translation errors are reported. Optimizing the camera pose regression network with the adversarial loss results in an improvement in accuracy, which is further increased by our proposed camera pose refinement.

Scene	Base Model	Ours	Ours+Ref.
Heads	$14.5°, 0.18\,\mathrm{m}$	$14.1°, 0.17\,\mathrm{m}$	$12.4°, 0.16\,\mathrm{m}$

the parameters β and α are set following the state-of-the-art [7] and λ is set to $1 \cdot 10^{-3}$. Afterwards, the pose regressor and discriminator are alternately trained on the $\mathcal{L}_G$ and $\mathcal{L}_D$ loss functions, respectively.

Implementation Details. Following the state-of-the art [7], input RGB images are down-sampled to a resolution of 341×256 pixels, normalized, and then fed in mini batches of size 64 to train the neural networks. As a camera pose regressor, a ResNet-34 network architecture is used as the base network, where the classification layers are removed and two fully connected layers for camera pose regression are placed after the average pooling layer. The discriminator consists of three convolutional layers followed by exponential linear units as activation function. All networks are implemented in PyTorch. For training the networks, we use the Adam Optimizer with a learning rate of $1 \cdot 10^{-4}$ and optimize for 300 epochs on an 11GB NVIDIA GeForce RTX 2080 graphics card. Once the networks are trained, the regressed camera poses are refined as described in Section 3.5 until convergence, but up to a maximum of 50 iterations at a step size of $l = 1 \cdot 10^{-3}$. The effect of the step size and the number of iterations on the resulting pose accuracy can also be found in more detail in Section 4.1.2.

4. Experiments and Evaluation

We evaluate our method on the publicly available 7-Scenes [36] dataset. This dataset from Microsoft consists of RGB-D frames of seven indoor scenes, captured with a hand-held Kinect camera, and corresponding ground truth camera poses computed using Kinect Fusion. The scenes are of varying spatial extent and also differ significantly in the amount of training data available. Training and test data are specified and consist of distinct camera trajectories. It has been widely used to evaluate camera re-localization methods as it contains several challenging scenarios such as motion blur, repeating structures and texture-less surfaces.

For evaluation, we utilize the recent state-of-the-art method and implementation of MapNet [7], focusing on di-
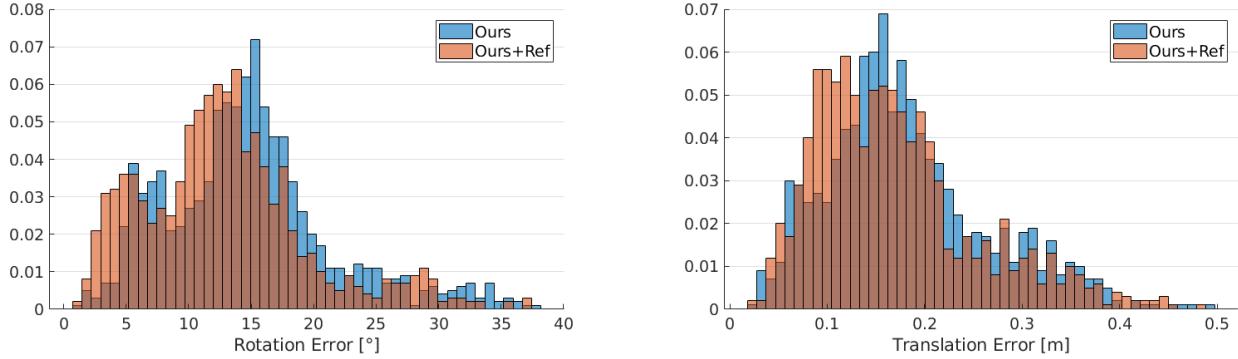
Figure 3: Normalized histograms of rotation and translation errors before and after pose refinement on the *Heads* scene. Results without refinement (Ours) are shown in blue, whereas errors after refinement (Ours+Ref.) are displayed in orange, resulting in an overlap in brown.

rectly regressing the camera pose without the aid of temporal or geometric information. We investigate the effect of our method on models either regressing quaternions themselves or the logarithm of a quaternion (baseline models of [7]). Further, for evaluation of our framework, we introduce the following models:

- **Baseline:** As a baseline model, we train the camera pose regression network on the $\mathcal{L}_{pose}$ loss, which, as already mentioned, effectively results in the state-of-the art baseline method of [7]. However, we abbreviate this model as "Base Model" whenever experiments are conducted by us to explicitly highlight re-trained models and to better analyze the effect of our contributions.

- **Adversarial Pose Regression:** To analyze the effect of adversarial training on the camera pose regression, the regression model is trained on the $\mathcal{L}_G$ loss function (Eq.4), abbreviated as "Ours".

- **Pose Refinement:** Finally, during testing, the trained discriminator network is used to further improve the regressed poses using $\mathcal{L}_{ref}$. The models are then abbreviated as "Ours+Ref".

In the remainder of this section, these models will be used to validate our contributions. We start by investigating the effect of optimizing a camera pose regression network including the adversarial loss, after which we analyze the effect of the proposed pose refinement on the localization accuracy. Finally, setting our method in the context of recent research, we compare our results to the current state-of-the-art methods on direct camera pose regression.

4.1. Ablation Studies

4.1.1 Adversarial Learning

First, to investigate the effect of adversarial learning on the camera pose regression framework, we compare rota-

tion and translation errors of our baseline, "Base Model", and the model "Ours". The results can be seen in Table 1, showing median rotation and translation errors of the described models on the *Heads* scene. That adversarial training can help in training deep networks has already been shown, for example in [45] for the task of human pose estimation, which, however differs significantly from the task of predicting the camera pose from a corresponding image. Nevertheless, we found slight improvements in rotation, as well as in translation accuracy by simply including adversarial training into a camera pose regression framework due to better and more stable convergence of the model.

4.1.2 Pose Refinement

As a second step, we evaluate our proposed pose refinement based on the trained discriminator network. Surprisingly, even though the gradients coming from the discriminator have not specifically been trained to have geometric meaningful information, it turns out that this information has implicitly been encoded in the network. Thus, we can use the gradients to update the regressed poses for any test image, given the constraints described in Section 3.5 on the quaternion update. Table 1 and Figure 3 summarize our findings, where we report the median rotation and translation error as well as the overall distribution of the aforementioned errors on the *Heads* scene of the 7-scenes dataset. Overall we found improvements in pose accuracy by applying the proposed pose refinement, examples of which are also visualized in Figures 1 and 4. Further examples for the remaining scenes of the 7-Scenes dataset can be found in the supplementary material. It can be seen both quantitatively and qualitatively that the regressed pose can effectively be pushed further towards the ground truth pose by the proposed refinement step, resulting for example in a relative improvement in rotation of 12.0% and 31.1% for the Heads

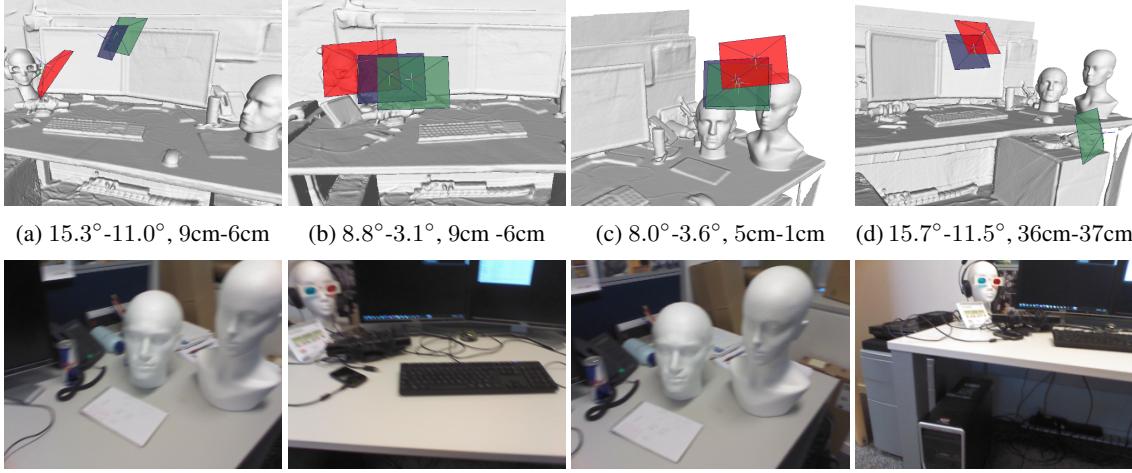| (a) 15.3°-11.0°, 9cm-6cm | (b) 8.8°-3.1°, 9cm -6cm | (c) 8.0°-3.6°, 5cm-1cm | (d) 15.7°-11.5°, 36cm-37cm |

Figure 4: RGB input images (second row) and the corresponding camera poses (first row), visualized in a reconstruction of the given scene. For each frame, the ground truth (green), initially regressed pose (red) and optimized pose using the proposed refinement (blue) are displayed. Below each visualization the respective rotation and translation errors before and after refinement are given.
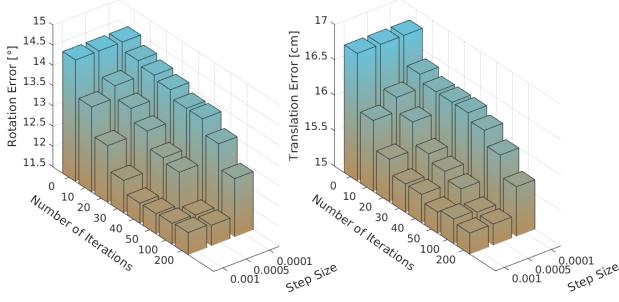

Figure 5: Effect of different numbers of iterations as well as step sizes on the median rotation and translation errors for the proposed refinement, shown on the *Heads* scene. Our refinement can significantly improve the localization accuracy even in a few iterations of optimization.

and Stairs scene respectively.

Further, we investigate the effect of the step size l as well as the number of iterations on the localization accuracy of the proposed pose refinement. The results of our investigation are summarized in Figure 5, where we show median rotation and translational error on the *Heads* scenes for different numbers of refinement iterations as well as step sizes. A lower step size usually leads to smaller changes in the pose, but, therefore, can also require a higher number of iterations to converge to the desired pose. Since this optimization process is required during testing, increasing the number of iterations is directly proportional to an increase in computational time. Experiments with larger step sizes ($l > 10^{-3}$) resulted in deterioration of the camera poses due

to the optimization procedure becoming unstable. Usually only a few iterations of refinement are sufficient, though, to improve the regressed poses and provide a good improvement in camera pose accuracy, whereas the run-time of RANSAC-based methods, for example, depends on the quality of correspondences found. As a trade-off, we chose the parameter setting described in Section 3.6. For example, on average the refinement has a computational time of 42ms for 30 iterations, but grows linearly with the number of iterations. Although we were able to achieve promising results with the proposed pose refinement strategy, it should be noted that it remains an optimization procedure itself, and thus depends on factors such as the quality of initialization. Therefore, in some cases the refinement might result in a solution that is not preferable to the initially regressed pose or difficult to recover from, if the predicted pose is far away from the ground truth one, an example of which is shown in Figure 4 d).

4.1.3 Influence of Feature Extractor

To evaluate the effect of the feature extraction network on the discriminator and thus the camera pose refinement, we evaluated our method using several different network architectures, namely AlexNet [23], VGG16 [37] and ResNet-18 [18]. Initialization is kept the same for all models and refinement is run for 30 iterations. Additionally we experiment with feeding only the regressed camera poses to train the discriminator network. For this experiment we replace the convolutional layers of the discriminator network with fully connected layers of roughly equal number of trainable

Table 2: Comparison between recent state-of-the-art direct camera pose regression methods and our results without (Ours) and with pose refinement (Ours+Ref.). Following the state of the art, displayed is the median rotation and translation error evaluated on the 7-Scenes dataset.

Scene	DSAC++ RGB [6]	PoseNet RGB [21]	MapNet [7]	Ours	Ours+Ref.	MapNet [7] log q	Ours log q	Ours+Ref. log q
Chess	0.02m, 0.7°	0.14m, 4.5°	0.11m, 4.2°	0.13m, 4.9°	0.12m, 4.8°	0.11m, 4.3°	0.13m, 5.0°	0.12m, 4.8°
Fire	0.03m, 1.1°	0.27m, 11.8°	0.29m, 11.7°	0.30m,11.0°	0.29m, 10.2°	0.27m, 12.1°	0.28m, 11.8°	0.27m, 11.6°
Heads	0.12m, 6.7°	0.18m, 12.1°	0.20m, 13.1°	0.17m, 14.5°	0.15m, 12.0°	0.19m, 12.2°	0.17m, 14.1°	0.16m, 12.4°
Office	0.03m, 0.8°	0.20m, 5.7°	0.19m, 6.4°	0.22m, 6.7°	0.21m, 6.6°	0.19m, 6.4°	0.20m, 7.1°	0.19m, 6.8°
Pumpkin	0.05m, 1.1°	0.25m, 4.8°	0.23m, 5.8°	0.23m, 6.7°	0.22m, 6.5°	0.22m, 5.1°	0.22m, 5.4°	0.21m, 5.2°
Red Kitchen	0.05m, 1.3°	0.24m, 5.5°	0.27m, 5.8°	0.27m, 5.9°	0.26m, 5.8°	0.25m, 5.3°	0.26m, 6.2°	0.25m, 6.0°
Stairs	0.29m, 5.1°	0.37m, 10.6°	0.31m, 12.4°	0.32m, 13.5°	0.30m, 12.2°	0.30m, 11.3°	0.29m, 12.2°	0.28m, 8.4°
Average	0.08m, 2.4°	0.24m, 7.9°	0.23m, 8.5°	0.23m, 9.0°	0.22m, 8.3°	0.22m, 8.1°	0.22m, 8.8°	0.21m, 7.9°

Table 3: Relative decrease, in percentage, of the median rotation and translation error after refinement in comparison to initially regressed poses. Evaluated are different network architectures used to obtain a feature representation of the RGB image input, showing the influence of the feature extractor on the proposed refinement. Higher values correspond to improved pose accuracy.

Heads	Without $f(x)$	AlexNet [23]	VGG-16 [37]	ResNet-18 [18]
Rotation	4.25%	3.56%	8.32%	12.18%
Translation	-3.0%	2.88%	4.7%	4.39%

parameters as the convolutional variant of the discriminator. Since a separate training is required for each architecture, we report the relative decrease in rotation and translation error over the initially regressed pose quality of the respective model. The results are summarized in Table 3. We found that our proposed refinement is fairly robust to the extracted features and were able to obtain improved pose accuracy regardless of the network architecture used, except when using pose information only, without additional information about the corresponding image representation. Nevertheless, we found an increase in localization performance depending on the choice of network architecture with the best performing model resulting in the ResNet-18 [18] network architecture.

4.2. Comparison to the State of the Art

As our main focus in this work is to investigate the effect of our proposed framework on direct camera pose regression methods using RGB information only, we show a comparison to recent methods working on this topic, namely PoseNet [21] and MapNet [7], which also forms our baseline model. We choose PoseNet and MapNet versions solely relying on single image and RGB information, for which we show the results in Table 2. We evaluate both models trained to predict quaternions as well as the logarithm of quaternions to show the effectiveness of our method regardless of the baseline representation used. In comparison to both [21] and [7], we found overall improvements in pose accuracy using the proposed refinement, where the effect of our method seems to be most profound on scenes for which only a small number of training images is available, such as *Heads* and *Stairs*. In addition we include a recent scene coordinate regression method, DSAC++ [6], that given an initial depth estimate, can be trained solely relying on RGB information. As can be seen, the regressed 3D information, and following pose refinement, greatly improve the accuracy of the predicted camera poses, which leads to the method outperforming direct camera pose regression methods and ours. This, however, comes at a significant drop in computational time. Lastly, although we focus on RGB only solutions in this paper, it should be mentioned that our core regression method could be easily extended to include further information, like relative pose information or geometric constraints as in [7].

5. Conclusion

In conclusion, we have presented a novel approach for camera re-localization applications solely relying on RGB information. Building on top of direct camera pose regression methods, we use the regressed camera poses and features extracted from the input image to train a discriminator network that tries to distinguish between regressed and ground truth poses, and thus implicitly tries to learn the geometric connection between RGB image and the corresponding camera pose. We have analyzed each component of our framework to evaluate this assumption and were able to achieve promising results. Further, we proposed a novel RGB-based pose refinement, where we use the trained dis-

criminator network to update and optimize the initially regressed poses, showing that the network can learn a meaningful representation of the camera poses and image space, and in turn can use this information to further improve localization accuracy.

A. Qualitative Results

6.9°-1.5°, 0.15 m-0.14 m   13.9°-9.2°, 0.15 m-0.05 m   13.4°-9.5°, 0.38 m-0.17 m   7.4°-3.0°, 0.32 m-0.35 m

2.9°-2.2°, 0.14 m-0.11 m   15.4°-3.9°, 0.1 m-0.06 m   6.7°-2.3°, 0.16 m-0.06 m   5.0°-3.7°, 0.24 m-0.26 m

10.8°-8.1°, 0.12 m-0.04 m   9.8°-6.2°, 0.13 m-0.07 m   14.2°-12.1°, 0.27 m-0.23 m   30.9°-29.8°, 0.41 m-0.42 m
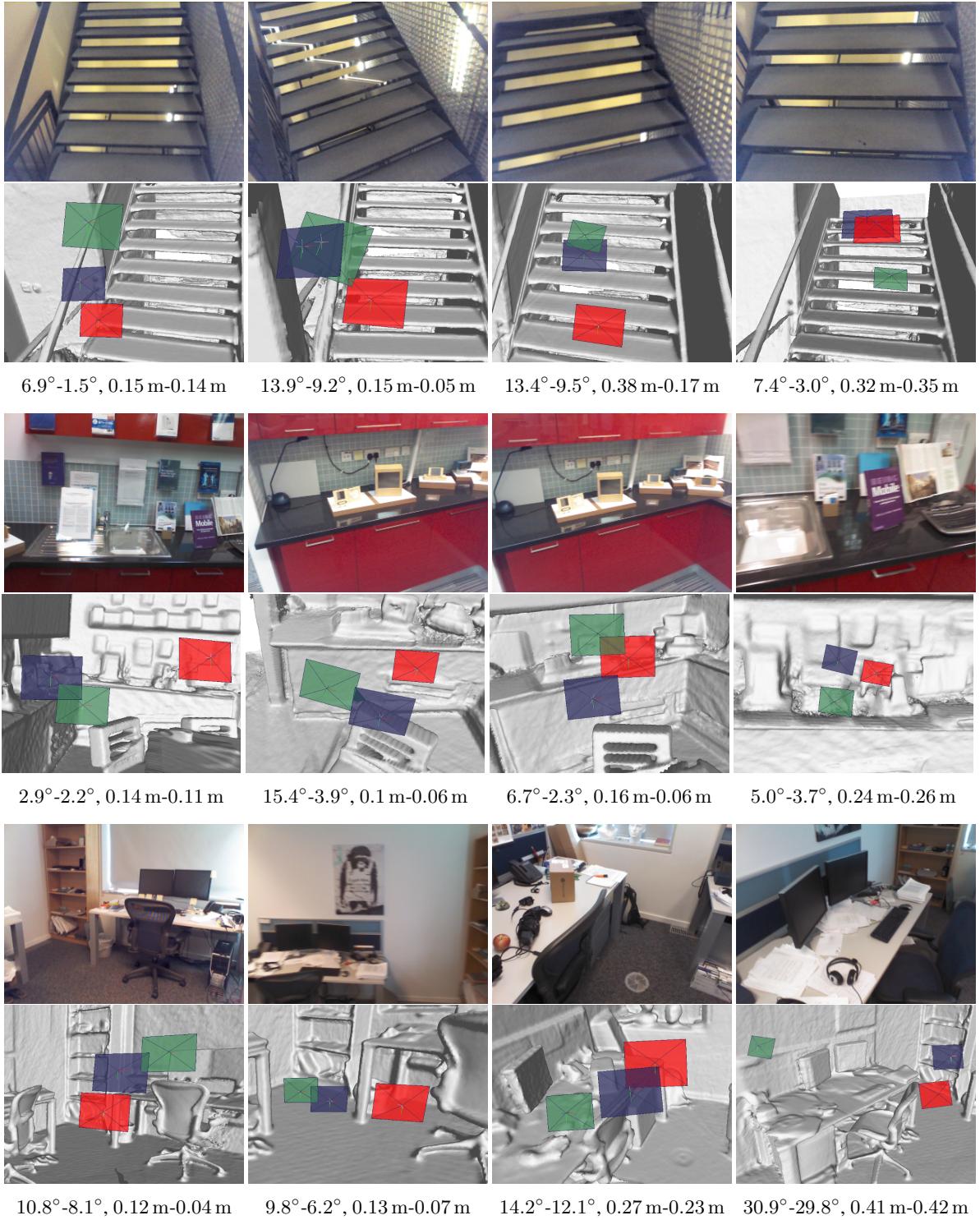
Figure 6: RGB input images (first row) and the corresponding resulting camera poses (second row), visualized in a reconstruction of the given scene (Stairs, Red Kitchen, Office). For each frame the ground truth (green), initially regressed pose (red) and optimized pose using the proposed adversarial refinement (blue) are displayed. Below each image initially regressed (left values) and refined (right values) rotation and translation errors are given.
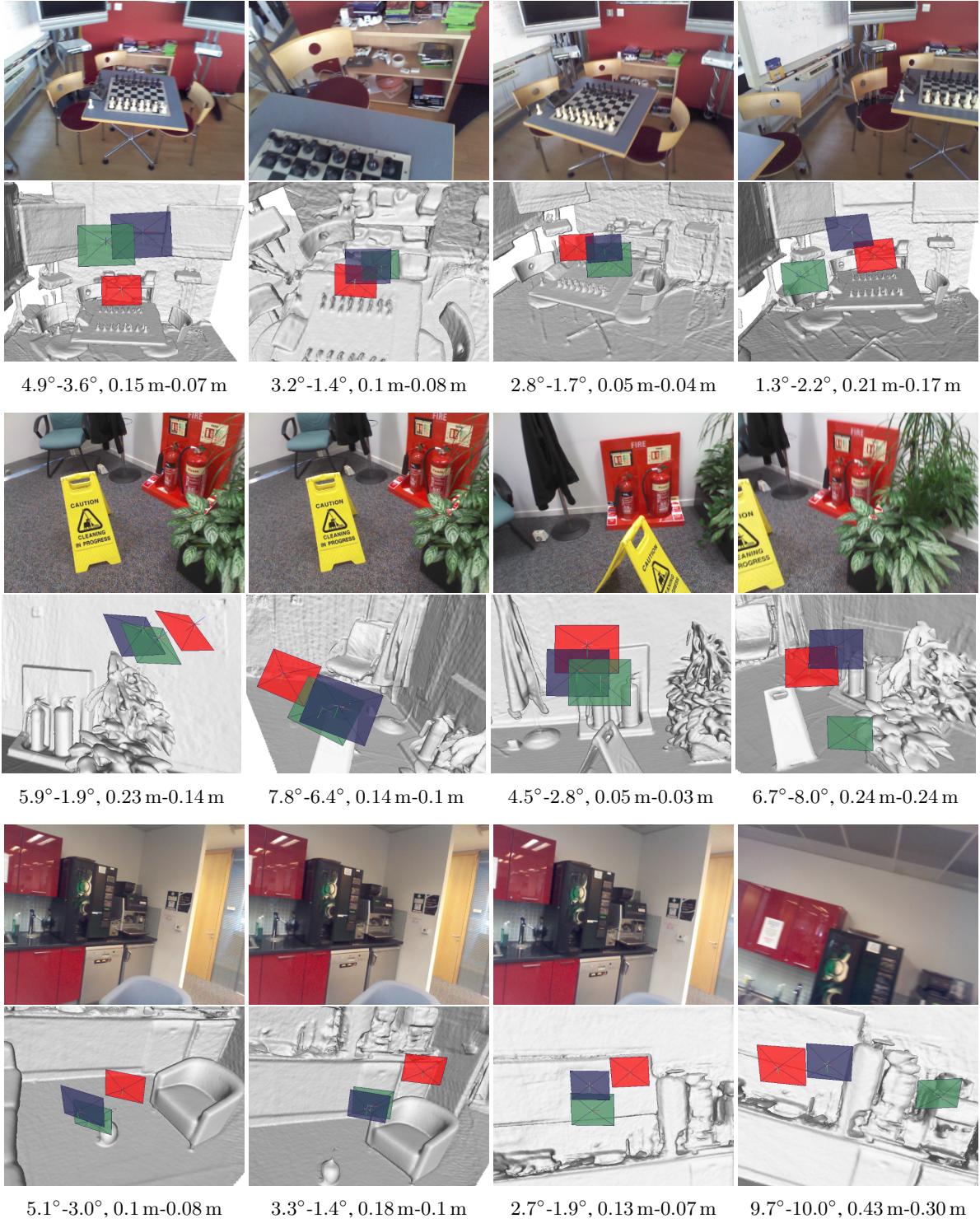
Figure 7: RGB input images (first row) and the corresponding resulting camera poses (second row), visualized in a reconstruction of the given scene (Chess, Fire, Pumpkin). For each frame the ground truth (green), initially regressed pose (red) and optimized pose using the proposed adversarial refinement (blue) are displayed. Below each image initially regressed (left values) and refined (right values) rotation and translation errors are given.

We show additional qualitative results of our method evaluated on the remaining scenes of the 7-Scenes dataset. RGB input images and the corresponding resulting camera poses, visualized in a reconstruction of the given scene, are shown in Figures 6 and 7. For each frame the ground truth, initially regressed pose and optimized pose using the proposed pose refinement are displayed. Rotation and translation errors of the regressed and refined camera poses are shown in the caption of each image pair.

B. Network Architectures

Further, the network architectures used to train the models described in this paper are given in more detail. For simplicity, we abbreviate fully-connected layers as FC, convolutional layers as C and average pooling layers as AP, where the resulting feature dimensionality is given as numbers after the respective layer. Further ELU stand for the exponential linear unit, whereas S describes the sigmoid function.

Camera Pose Regression Network The camera pose regression network consists of a ResNet-34 - $AP2048$ - $FC2048$ after which two fully connected layers for rotation $FC3/4$ and translation $FC3$ follow.

Feature Extraction Network This network consists of a pre-trained ResNet-18 - $GP512$ - $FC60/FC70$. All parameters of this network are fixed during training.

Discriminator Network The extracted features are concatenated with the replicated 6 or 7-dimensional pose vector and fed to the discriminator network, consisting of $C32$ - ELU - $C16$ - ELU - $C1$ - S.

C. Runtime Evaluation

Table 4 shows the computational times of the individual parts of our method evaluated for one frame. Pose refinement is calculated for 30 iterations. The method is implemented in Python and PyTorch and run on a 11GB NVIDIA GeForce RTX 2080 graphics card and 64 GB Intel Core i7.

Table 4: Computational times.

Pose regression	Feature extraction	Pose refinement	Overall
4.5ms	3ms	42ms	~ 50ms

References

[1] Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. Netvlad: Cnn architecture for weakly supervised place recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5297–5307, 2016.

[2] Vassileios Balntas, Shuda Li, and Victor Prisacariu. Relocnet: Continuous metric learning relocalisation using neural nets. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 751–767, 2018.

[3] Tolga Birdal, Umut Simsekli, Mustafa Onur Eken, and Slobodan Ilic. Bayesian pose graph optimization via bingham distributions and tempered geodesic mcmc. In *Advances in Neural Information Processing Systems*, pages 306–317, 2018.

[4] Francisco Bonin-Font, Alberto Ortiz, and Gabriel Oliver. Visual navigation for mobile robots: A survey. *Journal of intelligent and robotic systems*, 53(3):263–296, 2008.

[5] Eric Brachmann, Alexander Krull, Sebastian Nowozin, Jamie Shotton, Frank Michel, Stefan Gumhold, and Carsten Rother. Dsac-differentiable ransac for camera localization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 3, 2017.

[6] Eric Brachmann and Carsten Rother. Learning less is more-6d camera localization via 3d surface regression. In *Proc. CVPR*, volume 8, 2018.

[7] Samarth Brahmbhatt, Jinwei Gu, Kihwan Kim, James Hays, and Jan Kautz. Geometry-aware learning of maps for camera localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2616–2625, 2018.

[8] Mai Bui, Shadi Albarqouni, Slobodan Ilic, and Nassir Navab. Scene coordinate and correspondence learning for image-based localization. In *British Machine Vision Conference*, 2018.

[9] Simon Byrne and Mark Girolami. Geodesic monte carlo on embedded manifolds. *Scandinavian Journal of Statistics*, 40(4):825–845, 2013.

[10] Ming Cai, Chunhua Shen, and Ian Reid. A hybrid probabilistic model for camera relocalization. In *British Machine Vision Conference*, 2018.

[11] Tommaso Cavallari, Stuart Golodetz, Nicholas A Lord, Julien PC Valentin, Luigi Di Stefano, and Philip HS Torr. On-the-fly adaptation of regression forests for online camera relocalisation. In *CVPR*, volume 2, page 7, 2017.

[12] Yu Chen, Chunhua Shen, Xiu-Shen Wei, Lingqiao Liu, and Jian Yang. Adversarial posenet: A structure-aware convolutional network for human pose estimation. In *IEEE International Conference on Computer Vision (ICCV)*, pages 1212–1221, 2017.

[13] Ronald Clark, Sen Wang, Andrew Markham, Niki Trigoni, and Hongkai Wen. Vidloc: A deep spatio-temporal model for 6-dof video-clip relocalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, volume 3, 2017.

[14] Cai Ming Do Thanh-Toan, Pham Trung and Reid Ian. Real-time monocular object instance6d pose estimation. In *British Machine Vision Conference*, 2018.

[15] Ben Glocker, Jamie Shotton, Antonio Criminisi, and Shahram Izadi. Real-time rgb-d camera relocalization via randomized ferns for keyframe encoding. *IEEE transactions on visualization and computer graphics*, 21(5):571–583, 2015.

[16] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[17] Abner Guzman-Rivera, Pushmeet Kohli, Ben Glocker, Jamie Shotton, Toby Sharp, Andrew Fitzgibbon, and Shahram Izadi. Multi-output learning for camera relocalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1114–1121, 2014.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[19] Hervé Jégou, Matthijs Douze, Cordelia Schmid, and Patrick Pérez. Aggregating local descriptors into a compact image representation. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 3304–3311. IEEE, 2010.

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

[21] Alex Kendall, Roberto Cipolla, et al. Geometric loss functions for camera pose regression with deep learning. In *Proc. CVPR*, volume 3, page 8, 2017.

[22] Alex Kendall, Matthew Grimes, and Roberto Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *Proceedings of the IEEE international conference on computer vision*, pages 2938–2946, 2015.

[23] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[24] Zakaria Laskar, Iaroslav Melekhov, Surya Kalia, and Juho Kannala. Camera relocalization by computing pairwise relative poses using convolutional neural network. *IEEE International Conference on Computer Vision Workshop (ICCVW)*, pages 929–938, 2017.

[25] Chen-Hsuan Lin, Ersin Yumer, Oliver Wang, Eli Shechtman, and Simon Lucey. St-gan: Spatial transformer generative adversarial networks for image compositing. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9455–9464, 2018.

[26] Eric Marchand, Hideaki Uchiyama, and Fabien Spindler. Pose estimation for augmented reality: a hands-on survey. *IEEE transactions on visualization and computer graphics*, 22(12):2633–2651, 2016.

[27] Daniela Massiceti, Alexander Krull, Eric Brachmann, Carsten Rother, and Philip HS Torr. Random forests versus neural networkswhat's best for camera localization? In *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, pages 5118–5125. IEEE, 2017.

[28] Lili Meng, Jianhui Chen, Frederick Tung, James J Little, Julien Valentin, and Clarence W de Silva. Backtracking regression forests for accurate camera relocalization. In *Intelligent Robots and Systems (IROS), 2017 IEEE/RSJ International Conference on*, pages 6886–6893. IEEE, 2017.

[29] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015.

[30] Noha Radwan, Abhinav Valada, and Wolfram Burgard. Vlocnet++: Deep multitask learning for semantic visual localization and odometry. *IEEE Robotics and Automation Letters*, 3(4):4407–4414, 2018.

[31] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.

[32] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Fast image-based localization using direct 2d-to-3d matching. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 667–674. IEEE, 2011.

[33] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Improving image-based localization by active correspondence search. In *European conference on computer vision*, pages 752–765. Springer, 2012.

[34] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Efficient & effective prioritized matching for large-scale image-based localization. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (9):1744–1756, 2017.

[35] Tanner Schmidt, Richard Newcombe, and Dieter Fox. Self-supervised visual descriptor learning for dense correspondence. *IEEE Robotics and Automation Letters*, 2(2):420–427, 2017.

[36] Jamie Shotton, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew Fitzgibbon. Scene coordinate regression forests for camera relocalization in rgb-d images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2930–2937, 2013.

[37] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[38] Hajime Taira, Masatoshi Okutomi, Torsten Sattler, Mircea Cimpoi, Marc Pollefeys, Josef Sivic, Tomas Pajdla, and Akihiko Torii. Inloc: Indoor visual localization with dense matching and view synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7199–7209, 2018.

[39] Carl Toft, Erik Stenborg, Lars Hammarstrand, Lucas Brynte, Marc Pollefeys, Torsten Sattler, and Fredrik Kahl. Semantic match consistency for long-term visual localization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 383–399, 2018.

[40] Abhinav Valada, Noha Radwan, and Wolfram Burgard. Deep auxiliary learning for visual localization and odometry. In *International Conference on Robotics and Automation (ICRA 2018)*. IEEE, 2018.

[41] Julien Valentin, Matthias Nießner, Jamie Shotton, Andrew Fitzgibbon, Shahram Izadi, and Philip HS Torr. Exploiting uncertainty in regression forests for accurate camera relocal-

ization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4400–4408, 2015.

[42] Florian Walch, Caner Hazirbas, Laura Leal-Taixe, Torsten Sattler, Sebastian Hilsenbeck, and Daniel Cremers. Image-based localization using lstms for structured feature correlation. In *Int. Conf. Comput. Vis.(ICCV)*, pages 627–637, 2017.

[43] Xiaolong Wang, Abhinav Shrivastava, and Abhinav Gupta. A-fast-rcnn: Hard positive generation via adversary for object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

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

[45] Wei Yang, Wanli Ouyang, Xiaolong Wang, Jimmy Ren, Hongsheng Li, and Xiaogang Wang. 3d human pose estimation in the wild by adversarial learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, 2018.