Relative Camera Pose Estimation Using Convolutional Neural Networks

Iaroslav Melekhov[1], Juha Ylioinas[1], Juho Kannala[1], and Esa Rahtu[2]

[1] Aalto University, Finland,
firstname.lastname@aalto.fi
[2] Tampere University of Technology, Finland
esa.rahtu@tut.fi

Abstract. This paper presents a convolutional neural network based approach for estimating the relative pose between two cameras. The proposed network takes RGB images from both cameras as input and directly produces the relative rotation and translation as output. The system is trained in an end-to-end manner utilising transfer learning from a large scale classification dataset. The introduced approach is compared with widely used local feature based methods (SURF, ORB) and the results indicate a clear improvement over the baseline. In addition, a variant of the proposed architecture containing a spatial pyramid pooling (SPP) layer is evaluated and shown to further improve the performance.

Keywords: relative camera pose estimation, deep neural networks, spatial pyramid pooling

1 Introduction

The ability to estimate the relative pose between camera views is essential for many computer vision applications, such as structure from motion (SfM), simultaneous localization and mapping (SLAM) and visual odometry. Due to its practical importance, plenty of research effort has been devoted to the topic over the years. One popular approach to the problem is based on detecting and matching local feature points and using the obtained correspondences to determine the relative poses. The performance of such system is highly dependent on the accuracy of the local feature matches, which are commonly determined using methods like SIFT [1], DAISY [2], or SURF [3]. Unfortunately, there are many practically important cases where these hand-crafted descriptors are not able to find sufficient amount of correspondences. Particularly, repetitive structures, textureless objects, and extremely large viewpoint changes are difficult to handle. We highlight such cases in Fig. 1. An alternative solution would be to utilize all photometric information from the images to determine the poses. However, such methods (e.g. [4]) are usually not applicable to wide baseline settings, where there is large viewpoint change, or they may be computationally expensive.

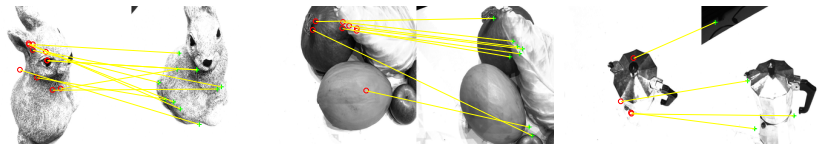


Fig. 1: Scenarios where traditional approaches are not able to estimate relative camera pose precisely. *Left*: very large viewpoint changes, thus most of inliers (correspondences) are not correct; *center*: the correct inliers concentrate on a small region; *right*: there is insufficient number of correspondences due to textureless scene (object with reflecting surface).

Recently, methods based on convolutional neural networks (CNNs) have clearly outperformed previous state-of-the-art results in many computer vision problems, such as image classification, object recognition, and image retrieval. In this work, we show how CNNs can also be applied to estimate the relative camera poses. Our contributions are as follows: 1) we propose a CNN-based method, which takes RGB images from both cameras as input and directly produces the relative rotation and translation as output; 2) we explore several network architectures and evaluate their performance on the DTU dataset [5]; 3) we study how different training strategies affect the results and make comparisons to popular keypoint based approaches. In addition, we investigate how spatial pyramid pooling [6] could be applied in the context of relative camera pose estimation problem.

The rest of the paper is organized as follows. Section 2 describes the related work focusing on relative camera pose estimation. The proposed approach and details related to network architectures and objective functions are introduced in Section 3. Finally, sections 4 and 5 present the baseline methods, experimental setup, evaluation results, discussion, and possible directions for future investigations.

2 Related Work

Over the years, a large variety of different local feature-based methods, such as SIFT [1], SURF [3], ORB [7], BRIEF [8], have been utilized in structure from motion, image-based localization, and visual SLAM contexts for estimating camera poses. The main disadvantage of these methods is their limited ability to cope with nuisance factors such as variations in viewpoint, reflections, and lack of distinguishable texture. As also noted in [9], some recent efforts indicate promise in approaching geometric computer vision tasks with a dense, featureless methods based on using full images. Taking this into account, one of the most prominent solutions is to apply convolutional neural networks (CNNs). While they have recently been applied in many computer vision problems, there are only a few works that consider them in the context of relative pose estimation.

Konda et al. [10] proposed a CNN architecture for predicting change in velocity and local change in orientation using short stereo video clips. They used a

rather shallow CNN architecture together with unsupervised pre-training of the filters in early layers. Partly because of the shortage of training data in their path prediction application, they were forced to discretize the space of velocities and local changes for a softmax-based classification instead of continuous estimates with regression. Mohanty et al. [11] tried to solve the same problem as in [10] using a monocular approach. In detail, they used an architecture based on two AlexNet-like CNN branches acting as inputs to a stack of fully connected layers coupled with a regression layer.

Ummenhofer et al. [12] proposed a CNN architecture for depth and relative camera motion estimation. They utilized multiple tasks in the learning phase to provide additional supervision in order to get more accurate depth maps and camera motion estimates. DeTone et al. [9] proposed a CNN architecture for estimating the relative homography between two images by regressing a 4-point homography parameterization with an Euclidean loss. Finally, instead of relative camera pose, Kendall et al. [13] proposed a CNN-based method for absolute 6-DoF camera pose estimation.

Our proposal is related to all previously discussed works, but it is the first one investigating the suitability of Siamese network architectures in the relative camera pose estimation problem. Compared with [10,11], our study aims at more general treatment of the camera pose estimation problem. That is, our approach is applicable for general unrestricted camera motion and for wide baseline view pairs, unlike [10,11]. Compared with [9], we are trying to solve relative camera pose, which can be regarded as a more general problem than solving the relative homography between two views. Regarding [13], we adopt the same learning objective but concentrate on solving a different problem. In particular, unlike prediction of absolute pose [13], relative pose estimation provides means for relation and representation learning for *previously unseen* scenes and objects. Finally, compared with [12], our study focuses on analyzing the differences in traditional and CNN-based approaches for relative camera pose estimation and does not consider the role of additional supervisory signals. That is, our approach does not require depth maps for training which is beneficial in practice. Further details of our approach will be given in the following sections.

3 Methodology

Our goal is to estimate relative camera pose directly by processing a pair of images captured by two cameras. We propose a convolutional neural network based method that predicts a 7-dimensional relative camera pose vector $\Delta\mathbf{p}$ containing the relative orientation vector $\Delta\mathbf{q}$ (4-dimensional quaternion), and the relative position, i.e. translation vector $\Delta\mathbf{t}$ (3-dimensional), so that $\Delta\mathbf{p} = [\Delta\mathbf{q}, \Delta\mathbf{t}]$.

3.1 Network Architectures

To estimate the relative camera pose between a pair of images, we apply a Siamese network architecture [14] (see Fig. 2). In detail, our network consists of

two blocks termed as the representation and the regression part. The representation part incorporates two identical CNN branches sharing the weights and other parameters. In general, both of these branches are composed of convolutional layers and rectified linear units (ReLU). The regression part in turn is composed of two fully-connected (FC1 and FC2) layers, where FC1 and FC2 have 4 and 3 connections respectively for estimating the 7-dimensional pose vector.

Following [13], we apply transfer learning. In detail, we take the Hybrid-CNN [15] as a base network for both of the branches and initialize them based on weights learned usin large-scale classification data. More specifically, Hybrid-CNN is AlexNet trained on both image classification *ImageNet* [16] and a scene-centric *Places* [15] datasets.

Extracting features from convolutional layers instead of fully-connected layers has shown to produce more accurate results in image retrieval problem [17,18,19]. Therefore, we removed the last three fully-connected layers (FC6, FC7 and FC8) from the original Hybrid-CNN preserving only convolutional, max-pooling layers and ReLU. More precisely, the network architecture for one branch has the following blocks: $\text{convB1}_{[96,11,4,0]}$, $\text{pool}_{[3,2]}$, $\text{convB2}_{[256,5,1,2]}$, $\text{pool}_{[3,2]}$, $\text{convB3}_{[384,3,1,1]}$, $\text{convB4}_{[384,3,1,1]}$, $\text{convB5}_{[256,3,1,1]}$, $\text{pool}_{[3,2]}$. The notation $\text{convB}_{[N,\omega,s,p]}$ consists of a convolution layer with N filters of size $\omega \times \omega$ with stride s and padding p and a regularization layer (ReLU), $\text{pool}_{[k,s]}$ is a max-pooling layer of size $k \times k$ applied with a stride s. The last layer of this baseline architecture dubbed *cnnA* is a pooling layer producing a tiny feature map (6×6) as an output. Therefore, due to reduction of spatial resolution, image details that are beneficial for relative camera pose estimation may have been lost at this part of the network. In order to limit such information loss, we remove the last max-pooling layer extracting features from convB5, which allows to have slightly larger feature maps (size 13×13). This modified version of the *cnnA* architecture is called *cnnB*.

Each branch of our representation part has an AlexNet-like structure originally designed for a fixed-size (227×227) input image. Such limitation may lead to reduced accuracy in the relative camera pose estimation degrading the performance of the system in general. To have more accurate estimations, it might be beneficial to process larger images to be able to extract more information from the scene structure. Theoretically, the convolutional layers accept arbitrary input image sizes, but they also produce outputs of variable dimensions. However, the FC layer of the regression part (see Fig. 2) requires to have fixed-length vectors as input. To make our pipeline to accept arbitrary image sizes, we apply a spatial pyramid pooling (SPP) layer which can maintain spatial information by pooling in local spatial bins [6]. An SPP layer consists of a set of pooling layers of $n \times n$ bins with the window size $w = ceil(a/n)$ and a stride $str = floor(a/n)$, where a is a size of the input feature map ($a \times a$) of the SPP layer. Therefore, the number of output bins of the SPP layer is fixed regardless of the image size.

We modified the original architectures *cnnA* and *cnnB* by adding an SPP layer to the end of each branch. Obtained networks (*cnnAspp* and *cnnBspp*) have 4-level (1×1, 2×2, 3×3, 6×6) and 5-level
(1×1, 2×2, 3×3, 6×6, 13×13) pyramid pooling respectively. An *cnnBspp*

structure is illustrated in Fig. 2. More detailed evaluation of the proposed network architectures is presented in Sec. 4.2.

3.2 Learning and inference

To regress the relative pose, the network was designed to compute the Euclidean loss between estimated vectors and ground truth. Following [13], we predict the relative orientation and position together using only one neural network learnt in a multi-task fashion.

Our loss function for training is as follows

$$\mathcal{L} = \left\| \Delta\hat{\mathbf{t}} - \Delta\mathbf{t} \right\|_2 + \beta \left\| \Delta\hat{\mathbf{q}} - \Delta\mathbf{q} \right\|_2 \tag{1}$$

where $\Delta\hat{\mathbf{q}}$ and $\Delta\hat{\mathbf{t}}$ are the ground-truth relative orientation and translation, and β is a parameter to keep the estimated values to be nearly equal. As described in [13], β must balance the orientation and translation estimations and can be set by grid search. In our experiments we set β equal to 10. The network is trained via back-propagation using stochastic gradient descent. The details of the training are described in Section 4.2.

It should be noted that quaternion vectors have unit length by definition and therefore $||\Delta\hat{\mathbf{q}}|| = 1$ in (1). Further, since the absolute scale of the translation can not be recovered, we normalize the ground truth translations to unit length as well, i.e. $||\Delta\hat{\mathbf{t}}|| = 1$. However, the norm constraints are not explicitly enforced during training. Instead, the estimates are normalized to unit length as a post-processing step like in [13].

Thus, at test time, a pair of images is fed to a regression neural network, consisting of two branches, which directly estimates the real-valued parameters of the relative camera pose vector. Finally, the estimated quaternion and translation vectors are normalized to unit length.

3.3 Error metrics

The error metrics that we use for evaluation of the performance are: (a) relative orientation error (ROE) in degrees and (b) relative translation error (RTE) in degrees. The latter error is the angle between the estimated translation vector and the ground truth. The former error is measured by determining the rotation angle for the rotation between the estimated orientation and the ground truth (i.e. we determine the rotation which rotates the estimated orientation onto the ground truth).

3.4 Datasets

It is essential to have a large and consistent dataset for training CNNs for relative camera pose problem. However, collecting such data may be expensive and laborious task. We overcome this by utilizing a crowd-sourced image collection provided by [20], where the ground truth camera poses are obtained by using an

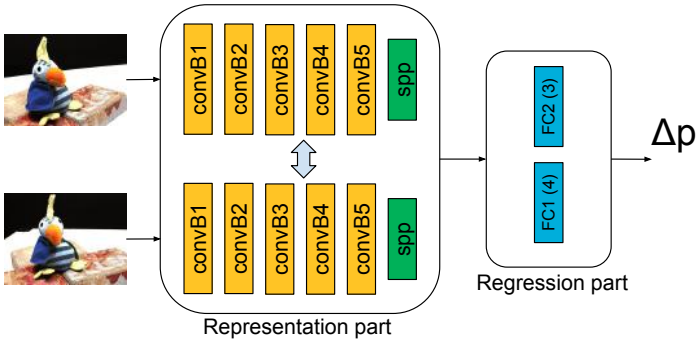


Fig. 2: Model structure (*cnnBspp*). Both network branches (representation part) have identical structure with shared weights. Pre-trained Hybrid-CNN [15] neural network was utilized to initialize the proposed architecture. Representation part maps an image pair to a low dimensional feature vector which is processed by regression part of the network. Regression part consists of 2 fully-connected layers (FC1 and FC2) and estimates relative camera pose.

.

(a) Roman Forum (b) Gendarmenmarkt (c) Montreal Notre Dame

(d) Piccadilly (e) Vienna Cathedral (f) Yorkminster

Fig. 3: Examples of the training (3a-3e) and the validation (3f) sets representing image pairs of six landmarks. The images were taken under different lighting and weather conditions, with variations of appearance and camera positions. Additionally, the dataset has a lot of occluded image pairs, so the problem of estimation relative camera pose becomes more challenging.

automatic structure from motion pipeline based on local feature matching. The collection consists of 13 subsets of images representing different landmarks and the numerical data of the global structure from motion reconstruction for each subset.

To evaluate the proposed CNN architectures we construct datasets for training and validation. The training set is composed of samples of five landmarks (Montreal Notre Dame, Piccadilly, Roman Forum, Vienna Cathedral and Gen-

Fig. 4: Example scenes from the DTU Robot Image Dataset [5]. The images show different objects which have been used in the evaluation dataset to estimate relative camera poses. In the dataset, camera positions are estimated very accurately as the camera was mounted on an industrial robot.

darmenmarkt) covering altogether 581k image pairs (see examples of each landmark in Fig. 3). For the validation set, we used the Yorkminster subset covering 22k image pairs in total. The ground truth labels (relative orientation and translation) are provided by [20] and were computed by applying the SIFT keypoint detector and descriptor followed by the structure and motion estimation via RANSAC, triangulation and iterative optimization.

In order to obtain a fair comparison between our approach and point-based methods, we need to specify an evaluation dataset where the ground truth orientation and translation are accurate and reliable, and not dependent on the success of traditional point-based motion estimation methods. For this, we utilize the DTU Robot Image Dataset provided by [5], where the ground truth is obtained by attaching the camera to a precisely positioned robot arm. The dataset covers 124 scenes containing different number of camera positions. Several object scenes of this dataset are illustrated in Fig. 4. See [5] for further details about the pipeline that was used to collect the DTU dataset.

4 Experiments

We evaluated the performance of our proposal on the DTU dataset comparing it with two traditional feature based methods, namely SURF [3] and ORB [7].

4.1 Preprocessing of the Evaluation Dataset

As explained, the DTU dataset consists of 124 scenes covering different number of camera positions. More specifically, it contains 77 scenes (type-I) with 49 camera positions and 47 scenes (type-II) with 64 camera positions. In order to estimate relative camera poses between pairs of views, we first determine the camera pairs which have overlapping fields of view.

Assuming that the pairwise relationships between cameras are represented in a symmetric $n \times n$ adjacency matrix, it is easy to see that the upper bound for the number of overlapping view pairs is $n\,(n-1)\,/2$, where n is the number of camera positions in the scene (49 or 64). Depending on the scene, this equals to 1176 and 2016, respectively. However, we compute relative poses only for the subset of pairs which have overlapping fields of view. These camera pairs can be determined easily since the intrinsic camera parameters are known for the

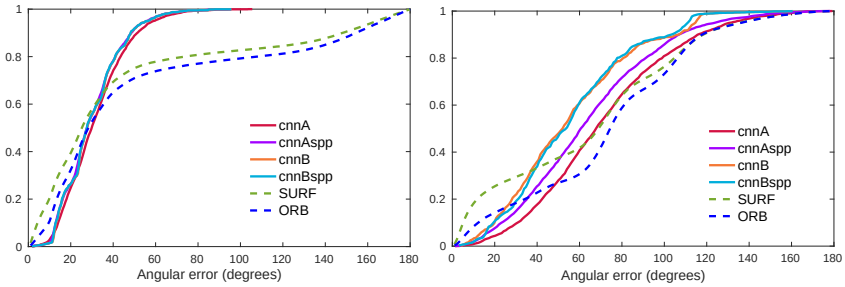


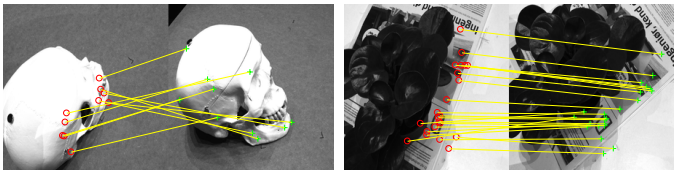
Fig. 5: Accuracy of our Siamese network architectures for estimating the relative camera orientation (left) and translation (right). Presented is the normalized cumulative histograms of errors for all scenes of the DTU dataset.

dataset. In detail, we define the field of view of each camera as a cone, i.e. a frustum, and calculate the intersection for pairs of corresponding frustums by using the publicly available OpenMVG [21] library. As a result, the number of overlapping pairs of views is 512 for scenes of type-I and 753 for type-II.

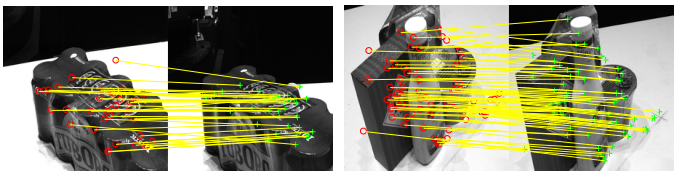
4.2 Comparing CNN models

To compare the discussed Siamese network architectures with and without SPP, i.e. cnnA/cnnB and cnnAspp/cnnBspp (see Sec. 3), we created training image pairs by rescaling the input images so that the smaller dimension was fixed to 323 pixels and keeping the original aspect ratio. Depending on the model, we then used either random 227×227 or 323×323 pixel crops for training (i.e. the larger size for architectures applying SPP), and a center crop (227×227) at the test time. To train our networks we used stochastic gradient descent (SGD) and the Adam solver [22]. Learning rate $\left(10^{-4}\right)$, weight decay $\left(10^{-5}\right)$ as well as the size of mini-batches (128) were the same for all models during the training phase. We used the publicly available machine learning framework Torch [23] on two NVIDIA Titan X cards applying the data parallelism trick to speed up training. It took around 60 hours to finish 15 training epochs with the 581k image pairs of the training dataset described in Section 3.4.

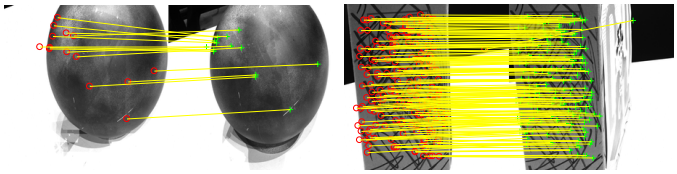
Fig. 5 shows a set of normalized cumulative histograms of relative orientation and translation errors for each of the discussed models evaluated on all scenes of the DTU dataset. According to the results it can be seen that having a bigger output feature map size before the final FC layers is clearly beneficial. This can be seen especially in the case of the reported error on relative translations (Fig. 5 right) where the cnnB and cnnBspp outperform cnnA and cnnAspp. In addition, utilizing SPP yields a further improvement. We dubbed the top-performing model (cnnBspp) to `cnn-spp` and used it for further experiments reported in the following section.



(a) Failure cases for which the traditional SURF approach was not able to detect enough matching points (inliers) properly or they were not distributed well in the image pair. As a result, the method has poor performance relative to the proposed method. ROE: 92.35° (*52.66°*); 57.86° (*29.27°*). RTE: 113.62° (*34.49°*); 118.9° (*56.42°*)



(b) Both approaches produce competitive results. ROE: 9.05° (*12.36°*); 19.93° (*12.9°*). RTE: 13.51° (*15.10°*); 53.74° (*58.09°*)



(c) Point-based descriptor finds sufficient amount of well-distributed features and outperforms our approach in relative translation estimation. ROE: 10.53° (*12.52°*); 9.24° (*11.55°*). RTE: 3.23° (*41.15°*); 15.42° (*57.62°*)

Fig. 6: Visual illustration of the performance of a baseline (SURF) on example image pairs. For all the cases (6a, 6b, 6c) we also report the error measures (ROE and RTE) of the baseline and our best model (in *parentheses*) on the given pairs.

4.3 Comparison to traditional methods

We compare our best model with a baseline consisting of two feature based methods, namely SURF and ORB. For both of these methods, we used the OpenCV implementation with the default parameters. The pose was recovered from the essential matrix estimated using the five-point method and RANSAC.

The results presented in Fig. 5 confirm that our proposed model `cnn-spp` performs better compared with the baseline feature based methods. For fair comparison, we resize all images of the DTU dataset to 227×227 size, transform internal camera parameters accordingly, and evaluate CNNs and feature-based approaches on this data. Our results show that transfer learning from external datasets can be utilized effectively to train a general relative pose regressor. Further, it should be noted that essential matrix estimation requires knowledge of the internal camera parameters, whereas our CNN based approach does not use that information.

We illustrate example results in Fig. 6. The yellow lines show matching points of SURF features across the images in a pair. The visualization shows that our method is robust and in some cases produces more accurate relative pose estimations than conventional point-based methods (Fig. 6a).

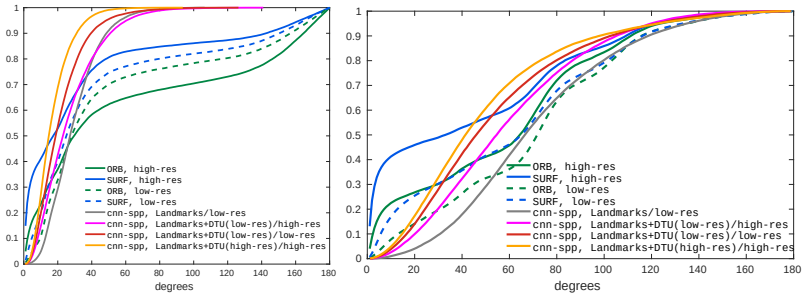
In Sec. 3.4 we described the data used to train our models in the previous experiments. However, the visual characteristics and distribution of relative poses in this data are quite different from the data used in evaluation, namely the DTU dataset. Therefore, given that many studies [13,16] show that it is essential to use training data which is as close as possible to the data in the target domain, it is interesting to see how the performance is affected if a subset of DTU scenes is used for training. Thus, we divided the DTU scenes into two sets, and used one of them to fine-tune cnn-spp model pre-trained on Landmarks dataset (Sec. 3.4). The other part of the DTU images was then used as the final testing set. Furthermore, according to the properties of SPP layer we conduct experiments using different input image sizes for the proposed model. Particularly, low resolution images (227×227) and high resolution (1600×1200) images were evaluated. The size of high resolution images corresponds to the original images in the test set of the DTU dataset.

The final results, illustrated in Fig. 7, show that the fine-tuned model produces more accurate relative camera orientation estimations than the one trained just on Landmarks dataset (red and grey curves respectively). Further, utilizing high resolution images both during training and evaluation leads to the best performance among the CNN-based methods (yellow curve). On average the proposed method falls slightly behind the feature based approaches in estimating the relative translation. However, based on our inspection it can be said that in certain cases (see for example Fig. 7b) where the objects are mostly textureless the proposed CNN-based camera pose regressor significantly outperforms traditional feature based approaches (Fig. 7c).

5 Discussion and Conclusion

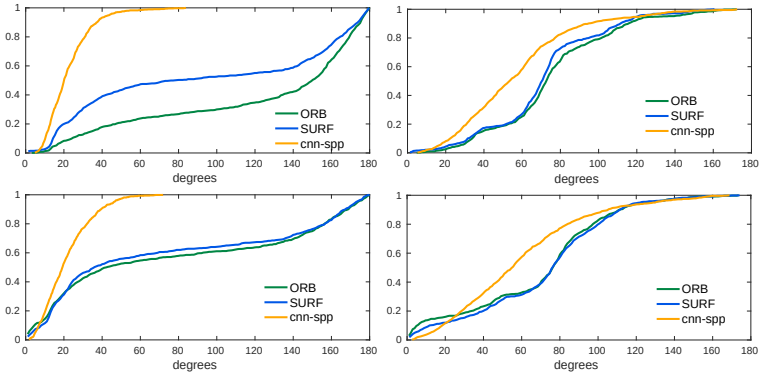
We presented an end-to-end trainable convolutional neural network based approach for estimating the relative pose between two cameras. We evaluated several different network architectures which all were based on the idea of combining two identical network structures with weight sharing, i.e. the Siamese network. We showed that the problem is solvable by this structure and that it is useful to have a larger feature map fed as input to the stack of final fully-connected layers. We further showed that applying spatial pyramid pooling is the key to even more accurate relative pose estimations as it opens the door for larger images that according to our results is one way to improve the accuracy. Overall, our proposal demonstrates very promising results, but there is also some room for further improvement.

For future directions, one interesting option is to construct a model based on two steps. During the first step, a CNN would produce coarse estimations and then, on the second step, would further refine the final estimate using the preliminary estimations. This kind of a model is reminiscent of the one presented in [12]. We leave constructing such a model for future work.



(a) Cumulative histogram of errors for relative camera orientation (left) and relative translation (right) for the test scenes of the DTU dataset. Experiments were evaluated on two different pre-defined image sizes: 227 × 227 (low-res) and 1600 × 1200 (high-res). The notation for CNN approaches is following: model, training data and the training image size (high-res or low-res), and the size of test images.

(b) Some scenes from the test set of the DTU dataset representing textureless objects with light reflections.



(c) Our CNN-based method performs clearly better than conventional local feature based approaches in estimating relative camera orientation (left column) and translation (right column) for the hard cases visualized in Fig. 7b. Particularly, neither SURF nor ORB are able to localize sufficient amount of inliers for such scenes, and, hence, their performance is quite poor.

Fig. 7: A comparison of traditional point-based methods and the proposed CNN-based approach for estimating relative camera pose. The first row (Fig. 7a) shows that in general cnn-spp predicts relative orientation more accurately than SURF or ORB descriptor, but in some cases falls behind in estimating relative translation. However, for cases (Fig. 7b) where point-based methods are not able to detect enough features, cnn-spp performs significantly better (Fig. 7c). Furthermore, utilizing high resolution DTU images during training can further improve the results (Fig. 7a).

References

1. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vision (2004)
2. Tola, E., Lepetit, V., Fua, P.: DAISY: An efficient dense descriptor applied to wide baseline stereo. IEEE Trans. on PAMI **32** (2010)
3. Bay, H., Tuytelaars, T., Gool, L.V.: SURF: Speeded up robust features. In: In ECCV. (2006)
4. Engel, J., Schöps, T., Cremers, D.: LSD-SLAM: Large-scale direct monocular SLAM. (2014)
5. Jensen, R., Dahl, A., Vogiatzis, G., Tola, E., Aanæs, H.: Large scale multi-view stereopsis evaluation. In: CVPR. (2014) 406–413
6. He, K., Zhang, X., Ren, S., Sun, J.: Spatial pyramid pooling in deep convolutional networks for visual recognition. In: ECCV. (2014) 346–361
7. Rublee, E., Rabaud, V., Konolige, K., Bradski, G.: ORB: An efficient alternative to sift or surf. In: ICCV. (2011)
8. Calonder, M., Lepetit, V., Strecha, C., Fua, P.: BRIEF: binary robust independent elementary features. In: ECCV. (2010)
9. DeTone, D., Malisiewicz, T., Rabinovich, A.: Deep image homography estimation. CoRR **abs/1606.03798** (2016)
10. Konda, K., Memisevic, R.: Learning visual odometry with a convolutional network. In: VISIGRAPP. (2015)
11. Mohanty, V., Agrawal, S., Datta, S., Ghosh, A., Sharma, V.D., Chakravarty, D.: DeepVO: A deep learning approach for monocular visual odometry. CoRR **abs/1611.06069** (2016)
12. Ummenhofer, B., Zhou, H., Uhrig, J., Mayer, N., Ilg, E., Dosovitskiy, A., Brox, T.: DeMoN: Depth and motion network for learning monocular stereo. CoRR **abs/1612.02401** (2016)
13. Kendall, A., Grimes, M., Cipolla, R.: PoseNet: A convolutional network for real-time 6-DOF camera relocalization. In: ICCV. (2015)
14. Chopra, S., Hadsell, R., LeCun, Y.: Learning a similarity metric discriminatively, with application to face verification. CVPR (2005)
15. Zhou, B., Lapedriza, A., Xiao, J., Torralba, A., Oliva, A.: Learning deep features for scene recognition using places database. NIPS (2014)
16. Krizhevsky, A., Sutskever, I., Hinton, G.E.: Imagenet classification with deep convolutional neural networks. In: Advances in NIPS. (2012)
17. Babenko, A., Lempitsky, V.S.: Aggregating deep convolutional features for image retrieval. ICCV (2015)
18. Razavian, A.S., Sullivan, J., Maki, A., Carlsson, S.: Visual instance retrieval with deep convolutional networks. CoRR **abs/1412.6574** (2014)
19. Azizpour, H., Razavian, A.S., Sullivan, J., Maki, A., Carlsson, S.: From generic to specific deep representations for visual recognition. In: CVPRW. (2015)
20. Wilson, K., Snavely, N.: Robust global translations with 1dsfm. In: ECCV. (2014)
21. Moulon, P., Monasse, P., Marlet, R., Others: Openmvg. an open multiple view geometry library. (`https://github.com/openMVG/openMVG`)
22. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. CoRR **abs/1412.6980** (2014)
23. Collobert, R., Kavukcuoglu, K., Farabet, C.: Torch7: A matlab-like environment for machine learning. In: BigLearn, NIPS Workshop. (2011)