SPP-Net: Deep Absolute Pose Regression with Synthetic Views

Pulak Purkait, Cheng Zhao, Christopher Zach
Toshiba Research Europe, Cambridge, UK
pulak.isi@gmail.com

Abstract

Image based localization is one of the important problems in computer vision due to its wide applicability in robotics, augmented reality, and autonomous systems. There is a rich set of methods described in the literature how to geometrically register a 2D image w.r.t. a 3D model. Recently, methods based on deep (and convolutional) feedforward networks (CNNs) became popular for pose regression. However, these CNN-based methods are still less accurate than geometry based methods despite being fast and memory efficient. In this work we design a deep neural network architecture based on sparse feature descriptors to estimate the absolute pose of an image. Our choice of using sparse feature descriptors has two major advantages: first, our network is significantly smaller than the CNNs proposed in the literature for this task—thereby making our approach more efficient and scalable. Second—and more importantly—, usage of sparse features allows to augment the training data with synthetic viewpoints, which leads to substantial improvements in the generalization performance to unseen poses. Thus, our proposed method aims to combine the best of the two worlds—feature-based localization and CNN-based pose regression–to achieve state-of-the-art performance in the absolute pose estimation. A detailed analysis of the proposed architecture and a rigorous evaluation on the existing datasets are provided to support our method.

1. Introduction

Image localization is the task of accurately estimating the location and orientation of an image with respect to a global map and has been studied extensively in robotics and computer vision. In this work we consider the more specific setting of estimating the perspective pose of an image with respect to a given 3D model, in particular 3D point clouds. Traditionally, this problem has been tackled either by direct 2D-3D matching (e.g. [16, 27]) or by inserting an image retrieval stage to narrow down the search space (e.g. [40, 28, 8]). PoseNet [13] and related approaches [38, 12, 36] demonstrated that deep learning methods—which have shown excellent performance in numerous classification and regression problems—are also able to estimate camera poses directly from input images.

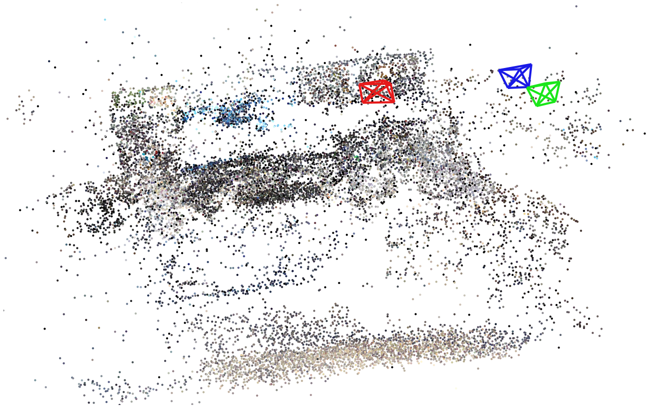


Figure 1: An example of 6DOF pose estimation results on heads sequence of the Seven Scenes dataset [6] where PoseNet [13] fails to predict accurate pose (marked by red, positional error = 0.31m and angular error = 27.4°) whereas the proposed SPP-Net predicts a pose (marked by blue, positional error = 0.06m and angular error = 2.18°) close to the ground truth (marked by green).

Despite the good performance of PoseNet and subsequent architectures for pose regression from images [13, 12, 36], we believe that PoseNet-like methods are fundamentally limited in the following ways:

1. Forward regression architectures such as deep neural networks have no built-in reasoning about geometry and most likely do not extract an "understanding" of the underlying geometric concepts during the training phase. Consequently, we postulate (and empirically validate) that PoseNet-like approaches suffer from poor extrapolation ability to unseen poses that are significantly different from the ones in the training set. In many application settings the distributions of training poses and test poses can differ substantially: training images (and poses) might be chosen such that structure-from-motion computation to

obtain a 3D model is made easier, whereas test poses may be arbitrarily distributed within the maneuverable space. Hence, pose regression typically faces a severe domain adaptation problem in general.

2. A lot of computation (and trainable parameters) in PoseNet-like architectures goes into the feature extraction stage, which uses rather heavy-weight CNNs such as VGG-Net [31] or GoogleNet [32]. In light of empirical evidence that gold-standard feature descriptors (such as SIFT [18]) in general have better accuracy in pose estimation than CNN-based approaches, we conjecture that heavy-weight dense feature extraction via CNNs is not necessary for this task. Hence, the networks used in our approach are significantly smaller and faster to train than existing CNN-based solutions for pose regression. Our choice to rely only on sparse features will be also very beneficial to address the above-mentioned domain adaptation problem.

The goal in this work is to close the gap between explicit correspondence-based methods and deep learning methods for pose regression. The general advantages of using deep learning for pose regression are the benefits of end-to-end training, the reduced memory requirements (≈ 100 MB for the network instead of several GB for a typical 3D point cloud database), and real-time performance (less than 50 ms for pose estimation per image).

Our contributions can be summarized as follows:

- We address the domain adaptation problem in pose estimation by augmenting the training set with synthetically generated training images and poses. In the spirit of [8] these synthetic poses may cover regions in pose space not available in the training data.
- Our choice of sparse features as input to the DNN implies, that we do not have to generate realistic RGB images for synthetic poses, but only have to predict realistic sets of features and associated descriptors.
- We refine the method in [8] to generate synthetic images leveraging the 3D map and feature correspondences.
- We propose a DNN architecture based on an ensemble of spatial pyramid max-pooling units [7] for pose regression. This network can be trained (from scratch) on those real+synthetic datasets without pretraining and is significantly smaller than PoseNet-like networks reported in the literature.
- We evaluate our proposed DNN on standard datasets for pose estimation, and we put a particular focus on the generalization performance to unseen test poses. We demonstrate that our method reduces the performance gap between the training based and direct feature based methods and produces state-of-the-art results among training based methods on benchmark datasets.

Thus, in this work we show that relatively light-weight pose regression network can achieve state-of-the-art results in CNN-based pose regression, and we demonstrate that adding synthesized data to the training set substantially improves its generalization ability to novel poses.

2. Related Work

Localization from 3D structure Localization and estimation of the camera pose with respect to a 3D environment, that is known in advance, is a well-studied problem in the field. A number of existing works (e.g. [16, 26, 17, 27]) draws its inspiration from the related problem of detecting loop closures in visual SLAM [20, 4, 2] by explicitly searching for 2D-3D correspondences (and subsequently determining the pose by a robust perspective N-point algorithm). If the underlying 3D point cloud is very large, then adding an intermediate, bag-of-features based step to quickly identify relevant subsets of the point cloud can be highly beneficial to reduce the computational costs [40, 28, 8]. One problem with methods using a 3D point cloud is a somewhat limited scalability: works such as [17] use relatively large point clouds containing many millions of 3D points, but ultimately cover only a tiny fraction of the world.

Localization via image retrieval One way to make location recognition more scalable is by casting it as an image retrieval task. In image retrieval it is relatively well understood how to index large image collections for efficient search, which can be based on bag-of-features (e.g. [21, 23, 11]) or more general global image descriptors (such as [35, 37, 10, 22]). Without an underlying 3D model, location (or place / landmark) recognition via image retrieval can only provide an approximate estimate for the pose and will not determine all necessary 6 d.o.f. accurately.

Learning based algorithms To our knowledge the first attempt to use machine learning techniques specifically for pose estimation is [30], where a random forest regressor is used to (densely) predict initial 2D-3D correspondences from image appearance. The resulting pose is obtained by subsequent robust estimation. PoseNet [13] is probably the first approach aiming for end-to-end, CNN-based pose regression from an input image. The network (based on a pre-trained GoogleNet) has to extract higher-level and invariant properties of an image in order to accurately predict the pose parameters. [36] enhance in the regression layers by utilizing LSTMs, and [12] improve upon the original PoseNet by leveraging different loss functions, and [3, 15, 19] are recent developments in direct CNN-based pose estimation.

Since none of the above deep learning architecture for pose regression has an intrinsic "understanding" of the geometric concepts behind pose estimation, the generalization performance to new poses that are significantly different to the training poses is expected to be poor. Note that "geometric" methods for pose estimation will suffer to some ex-

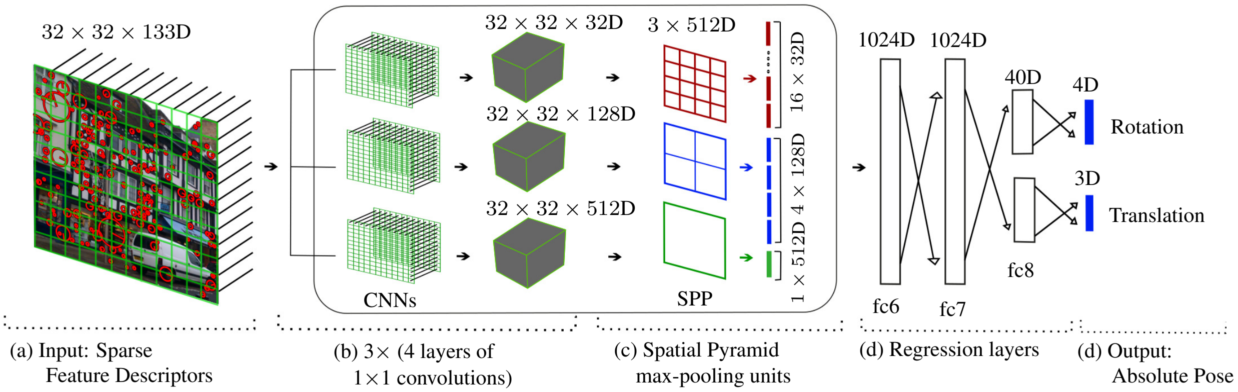


Figure 2: Proposed SPP-Net for absolute pose regression takes sparse feature points as input and predicts the absolute pose.

tent from the same problem (e.g. due to limited view-point invariance of feature descriptors), but will in general much better extrapolate to novel poses. This work aims improve the generalization ability of CNN-based pose regression by using an architecture that is capable to learn from synthesized training examples.

3. Spatial pyramid pose net

In the following we describe our proposed DNN architecture, which we term "spatial pyramid pose net" or SPP-Net. SPP-Net takes a set of sparse feature descriptors as input and estimates 6 d.o.f. camera pose. The input descriptors undergo a number of 1×1 convolutions/ReLU layers, followed by an ensemble of multiple parallel max-pooling layers, which are succeeded by three fully connected pose regression layers. The proposed SPP-Net is a light-weight, fast, and it performs analogously with the original PoseNet [13]. Moreover, the proposed architecture has an additional advantage that it can be further trained on augmented images generated from the reconstructed 3D map. Being trained on such synthetic poses, it produces state-of-the-art results on benchmark datasets. In the following subsections, we describe the proposed network in detail and in section 4, we present our approach for simulating augmented views.

Feature processing Given a set of input keypoints $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots \mathbf{x}_N\}$, the task is to estimate the pose of the camera (R, T) with respect to the global coordinate frame. The i-th keypoint $\mathbf{x}_i$ is described by its pixel coordinates (p_i, q_i), scale s_i, orientation θ_i, and a feature descriptor $\mathbf{f}_i$ of dimensionality D. The number of of sparse features extracted from input images varies, and there are two complementary approaches to facilitate the use of a CNN on sparse features: the method used in [5] embeds the sparse features into a dense grid (and has to handle feature collisions and empty locations), and PointNet [24] (and similarly [33]) processes each input feature independently and uses max-pooling to symmetrize the network output. Our approach combined elements of both by using spatial binning, indepdent feature processing and global max-pooling: we arrange the set of keypoints on a 2D regular grid based on the pixel locations (p_i, q_i) and split the input image of size $W \times H$ into $d_1 \times d_2$ cells (where each cell is of size $W/d_1 \times H/d_2$ pixels). If a cell occupies multiple features, we select a feature randomly and obtain a $(D + 5)$-dimensional vector $\big(\mathbf{f}_i, p_i \bmod d_1, q_i \bmod d_2, \sin\theta_i, \cos\theta_i, \log(1 + s_i)\big)$ corresponding to a keypoint $\mathbf{x}_i$. This results a spatially organized array of at most $d_1 \times d_2$ feature descriptors (with $D + 5$ dimensions), which is the input to the network. Empty cells are represented by a zero feature vector. In all of our experiments, we used $d_1 = d_2 = 32$ for the (outdoor) datasets of large images and $d_1 = d_2 = 16$ for the (indoor) datasets of small images.

The rationale behind our spatial binning approach is to reduce the amount of processing and to balance the spatial distribution of features across the image.

Network Architecture As shown in Figure 2, the proposed network consists of an array of CNN subnets, an ensemble layer of max-pooling units at different scales and two fully connected layers followed by the output pose regression layer. At each scale, a CNN feature descriptors is fed to the ensemble layer of multiple maxpooling units [Fig. 2(b)]. A CNN consists of 4 convolution layers of size 1×1 of dimensionally D'_s which are followed by relu activation and batch normalization. Thus, the set of $d_1 \times d_2$, $(D + 5)$-dimensional input descriptors is fed into the CNNs at multiple scales, each of which produces feature map of size $d_1 \times d_2 \times D'_s$. Note that the number of feature descriptors is unaltered during the convolution layers. Experimentally we have found that the chosen 1×1 convolutions with stride 1×1 performs better than larger convolutions. In all of our experiments, we utilize SIFT descriptors of size $D = 128$ and the dimension of the CNN feature map D'_s at level s is chosen to be $D'_s = 512/2^{2s}$.

Inspired by spatial pyramid pooling [7], in SPP-Net we concatenate the outputs of the individual max-pooling layers before reaching the final fully connected regression layers. We use parallel max-pooling layers at several resolutions: at the lowest level of the ensemble layer has D'_0 global max-pooling units (each taking $d_1 \times d_2$ inputs), and at the sth level it has $2^{2s} \times D'_s$ max-pooling units (with a receptive field of size $d_1/(2^s) \times d_2/(2^s)$). The response of all the max-pooling units are then concatenated to get a fixed length feature vector of size $\sum_s 2^{2s} \times 512/2^{2s} = 512 \times (s+1)$. In all of our experiments, we have chosen a fixed level $s = 2$ of max-pooling unites. Thus, the number of output feature channel of the ensemble layer is $D' = 1536$. The feature channels are then fed into two subsequent fully connected layers (fc6 and fc7 of Fig. 2) of size 1024. We also incorporate dropout strategy for the fully connected layers with probability 0.5. The fully connected layers are then split into two separate parts, each of dimension 40 to estimate 3-dimensional translation and 4-dimensional quaternion separately.

The number of parameters and the operations used in different layers are demonstrated in Table 1. A comparison among different architectures can also be found in Table 2.

type / depth	patch size / stride	output	#params	# FLOPs
conv0/1	$1 \times 1/1$	$32 \times 32 \times 128$	17K	17M
conv0/2	$1 \times 1/1$	$32 \times 32 \times 256$	32.7K	32.7M
conv0/3	$1 \times 1/1$	$32 \times 32 \times 256$	65.5K	65.5M
conv0/4	$1 \times 1/1$	$32 \times 32 \times 512$	131K	131M
conv1/1	$1 \times 1/1$	$32 \times 32 \times 128$	17K	17M
conv1/2	$1 \times 1/1$	$32 \times 32 \times 128$	16.4K	16.4M
conv1/3	$1 \times 1/1$	$32 \times 32 \times 128$	16.4K	16.4M
conv1/4	$1 \times 1/1$	$32 \times 32 \times 128$	16.4K	16.4M
conv2/1	$1 \times 1/1$	$32 \times 32 \times 128$	17K	17M
conv2/2	$1 \times 1/1$	$32 \times 32 \times 64$	8.3K	8.3M
conv2/3	$1 \times 1/1$	$32 \times 32 \times 64$	4.1K	4.1M
conv2/4	$1 \times 1/1$	$32 \times 32 \times 32$	2K	2M
max-pool0/5	$32 \times 32/32$	$1 \times 1 \times 512$	–	–
max-pool1/5	$16 \times 16/16$	$2 \times 2 \times 128$	–	–
max-pool2/5	$8 \times 8/8$	$4 \times 4 \times 32$	–	–
fully-conv/6	–	1×1024	1.51M	1.51M
fully-conv/7	–	1×1024	1.04M	1.04M
fully-conv/8	–	1×40	82K	82K
fully-conv/8	–	1×40	82K	82K
pose T/9	–	1×3	0.1K	0.1K
pose R/9	–	1×4	0.1K	0.1K
			$\approx$ 3M	346.3M

Table 1: A detailed descriptions of the number of parameters and floating point operations (FLOPs) utilized at different layers in the proposed SPP-Net.

Method	#params	#FLOPs
SPP-Net (Proposed)	3M	0.35B
Original PoseNet (GoogleNet) [13]	8.9M	1.6B
Baseline (ResNet50) [15, 19]	26.5M	3.8B
PoseNet LSTM [38]	9.0M	1.6B

Table 2: Comparison on the number of parameters and floating point operations (FLOPs).

Loss function We follow [12] in the choice of the loss,

$$\mathcal{L}_\sigma(q, T) \propto \sigma_q^{-2} \left\| q^\dagger - q/\|q\| \right\| + \sigma_T^{-2} \|T^\dagger - T\| + \log \sigma_q^2 + \log \sigma_T^2 \tag{1}$$

where $q^\dagger$ and $T^\dagger$ are the ground truth orientation and position of the image, respectively. We employ unit quaternions to represent 3D rotations. Note the reprojection error could be a geometrically more meaninful loss function, especially since SPP-Net takes sparse features as input. As also pointed out in [12], we found it difficult to train a network directly using the reprojection loss, hence we rely on (1) instead.

4. Mining new views

In this section, we discuss our proposed method for mining synthetic poses and generating synthetic features "images". Before generating synthetic views, we remove all the points in the point-cloud which are either seen in only the test images or observed in fewer than two training images. Also, the observed image indices and the respective feature descriptors for the remaining points corresponding to the test images are removed. Thus, in our preprocessed 3D point cloud there is no information about the test image set.

Each point $\mathbf{X}_i$ in the reconstructed 3D point cloud $\mathcal{D}$ contains the 3D location (X_i, Y_i, Z_i), the indices of the images $\mathcal{I}_{\mathbf{X}_i}$ where the point $\mathbf{X}_i$ was observed and the indices of the keypoints in the observed image. Moreover, the positions and the orientations of the images $\mathcal{I}_{\mathbf{X}_i}$ are also available. This enables us to synthesize more realistic unobserved views from the 3D point cloud $\mathcal{D}$. Inspired by the idea of view synthesis in the context of absolute pose estimation [8, 25, 34] we utilize a similar strategy as described below.

4.1. Pose set augmentation

For outdoor datasets, a horizontal plane is robustly fitted to the training camera positions. The synthetic poses are then generated by perturbing each training pose along the detected horizontal plane. Translations and orientations are chosen uniformly within the range of $[-2.5\text{m}, 2.5\text{m}]$ and $[-30^\circ, 30^\circ]$ respectively. The axis of the angular shift is chosen as the normal to the detected horizontal plane. For indoor datasets the random shifts are given along all the directions uniformly in the interval $[-0.25\text{m}, 0.25\text{m}]$ and random orientations $[-30^\circ, 30^\circ]$ along arbitrary axes. The traversal is performed 50 times for each of the training pose. The intrinsic camera parameters (focal length, radial distortions) of the synthetic views were chosen to be the same as the training images. Note that in none of the datasets any prior knowledge of the test poses are exploited during the pose augmentation.

The set of synthetic poses is made more distinctive by removing all the repetitive poses, *e.g.*, poses within 0.1m location and 1° orientation of an existing training pose. Furthermore, synthetic poses inside the point-cloud or extremely closed to it (more than 25 points inside the frustum and within radius 1m from the camera center) are not useful and realistic, thus have been removed consequently. Note that no optimal placement and orientation strategies are utilized unlike in [8]. However, we believe that the performance of the network could generalize well if the augmented poses cover the area of the concerned view-points of interest. The current pose mining produces best results on the benchmarking datasets.

4.2. Generate synthetic views

For a given camera pose $\mathcal{P}$, we project the points $\mathbf{X}$ from the point-cloud $\mathcal{D}$ in the front of the camera and within the viewing frustum. Not all the points might be relevant here—we keep only those points which ensure *detectability* and *repeatability* of the descriptors under perspective distortions. Based on that the following selection criterion are made:

1. the relative scale $s_\mathbf{X}$ of the projected point of $\mathbf{X}$ must be greater than 1.25 and less than 120.0,
2. at least one of the original viewing direction must be oriented within 20° of the current view.

Note that the scale $s_\mathbf{X}$ is computed as the relative scale w.r.t. the observed image under consideration.

Once a 3D point is chosen, the feature descriptor $\mathbf{f}_i$ is copied corresponding to the nearest observed image. The pixel coordinate (p_i, q_i) of the feature point is computed under the perspective projection of the 3D point $\mathbf{X}$. The rotation of the feature descriptor is copied from the chosen observed image. At this point we further discard poses, in which the projected point cloud are not sufficiently well spatially distributed in the image: at least four of 4×4 bins arranged over the image have to be non-empty for a sampled pose to be processed further.

In order to make the synthesized view robust to noise and outliers, we add the following sources of noise:

- Additive Gaussian noise with diagonal co-variance Σ_x is added with the feature descriptors $\mathbf{f}_i$. The co-variance matrix Σ_x is determined based on the descriptors in the training data. Further, the projected pixel locations (p_i, q_i) are corrupted by Gaussian noise with variance of 1 pixel.
- Outliers comprising of 25% of the total number of projected points are added from randomly chosen pixel locations. The feature descriptors, scale and rotations of the outliers are copied from randomly chosen 3D points.
- Outlier keypoints from the training images, that are not utilized for SfM reconstruction, are also projected to the synthetic poses. In this case, we fit an homography through all the inlier points of the synthesized pose and the training pose, and then use the same homography to project the outliers (25%) to the target synthetic image. This step is omitted if the number of common inliers is less than 50.

The above procedure is summarized in algorithm 1

Algorithm 1: Synthetic pose generation from 3D map

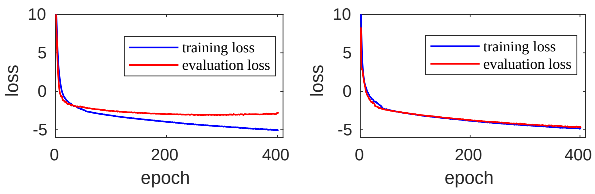
```
Input  : 3D map $\{\mathcal{D}\}$, Training feature descriptors $\{\mathcal{I}\}$, and poses $\{\mathcal{P}_\mathcal{I}\}$
Output: Synthetic pose and feature descriptors $\{\mathcal{P}, \mathcal{F}\}$
$\{\mathcal{P}\} := \emptyset$; $\{\mathcal{P}, \mathcal{F}\} := \emptyset$; $i := 0$
forall $\mathcal{P} \in \{\mathcal{P}_\mathcal{I}\}$ do
    while $i < 50$; $i$++ do $\{\mathcal{P}\}$.append := augment pose ($\mathcal{P}$);
$\{\mathcal{P}\}$ := prune repetitive pose $\{\mathcal{P}\}$ ;                    /* [see sec 4.1] */
forall $\mathcal{P} \in \{\mathcal{P}\}$ do
    $\mathcal{F} = \emptyset$; project $\mathbf{X}$ into the camera plane $\forall \mathbf{X} \in \{\mathcal{D}\}$;
    forall $\mathbf{X} \in$ viewing frustum of $(\mathcal{P})$ do
        forall observed images $\mathcal{I}_\mathbf{X}$ of the 3D point $\mathbf{X}$ do
            if $s_\mathbf{X} \notin [1.25, 120]$ then continue;
            if $\angle(\mathcal{P}, \mathcal{I}_\mathbf{X}) > 20^\circ$ then continue;
            $\mathcal{F}$.append := projected pixel location of $\mathbf{X}$ + scale $s_\mathbf{X}$ and
              orientation $\theta$ + descriptors at $\mathbf{X}$ in $\mathcal{I}_\mathbf{X}$ ;   /* [see sec 4.2] */
    if $\mathcal{F}$ passes the sustainability check then
        $\mathcal{H}$ := homography between the inliers of $\mathcal{I}$ and $\mathcal{F}$;
        $\mathcal{F}$.append := outlier keypoints of $\mathcal{I}$ projected by $\mathcal{H}$;
        $\mathcal{F}$.perturbation (noise & outliers) ;                  /* [see sec 4.2] */
        $\{\mathcal{P}, \mathcal{F}\}$.append := $(\mathcal{P}, \mathcal{F})$;
return $\{\mathcal{P}, \mathcal{F}\}$ ;
```

5. Experiments

The proposed SPP-Net is trained on a number of widely used datasets for absolute pose estimation using a Tensorflow [1] implementation. The loss (1) is minimized using ADAM [14] with a batch size of 300. The weight decay is set to 10^{-5}. The network is trained for 400 epochs with an initial learning rate 0.001 which is gradually decreased by a factor of 10 after every 100 epochs. All the experiments are evaluated on a desktop equipped with a NVIDIA Titan X GPU, where evaluation of the SPP-Net requires about 2ms of run-time. The network takes $2-4$hrs to train for a typical dataset and only 36.8 MB to store the weights.

5.1. Validation of the proposed pose augmentation

To validate the efficiency of the augmented poses, we conduct an experiment with one of the difficult sequences (heads) of Microsoft's 7-Scenes Dataset [30]. From the 3D map of training images, we synthetically generate views



(a) training data only (b) additional synthetic data

Figure 3: Training and evaluation loss on testing dataset. When the model is trained with the training data only the evaluation loss quickly become stagnant.

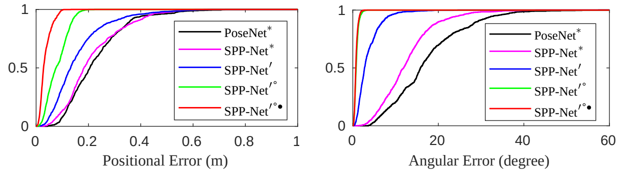


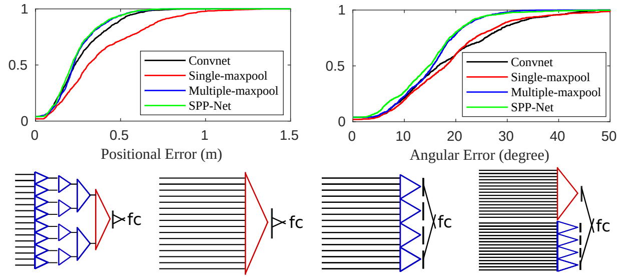
Figure 4: The above figures demonstrate our localization accuracy for both position and orientation as a cumulative histogram of errors for the entire testing set. Where the baselines—Net*: trained with the training data only, Net$'$: trained with the clean synthetic data, Net$'^{\circ}$: trained with the synthetic data under realistic noise, Net$'^{\circ\bullet}$: trained with the synthetic data under realistic noise and outliers.

corresponding to *test poses*. The proposed SPP-Net is then trained on the training images + the synthetically generated images (sparse feature descriptors) and evaluated on the feature descriptors extracted from the original test images. The generated synthetic test images do not exploit any test image content but the 6 d.o.f. poses[1]. If the networks is provided only with the training images, it does not generalize well to the test images. However, after adding synthetic test poses to the training data, the evaluation loss decreases in conjunction with the training loss. In Figure 3 we illustrate the training and evaluation loss with and without additional synthetic training data. These results justify the utilization of our synthetic pose augmentation method.

Using SPP-Net with synthesized poses clearly improves the network's ability to generalize beyond real training poses. Moreover, SPP-Net can be trained for a target set of poses of interest, which e.g. might not be represented in the original training set. This is the main motivation for choosing using sparse features for absolute pose regression.

We perform another experiment to validate different steps of the proposed augmentation, where we generate three different sets of synthetic poses with increasing realistic adjustment on each step of the synthetic image generation process. The first set of synthetic poses contains no noise or outliers, the second set is generated with added noise, and the third set is generated with added noise and outliers as described above. Note that all the networks are evaluated on the original sparse test feature descriptors. We also evaluate PoseNet [13], utilizing a tensorflow implementation available online [2], trained on the original training images for 800 epochs. The proposed SPP-Net, trained only on the training images, performs analogously to PoseNet. However, with the added synthetic poses the performance improves immensely with the realistic adjustments as shown in Figure 4. Note that since PoseNet uses full image, it cannot easily benefit from augmentation.

An additional experiment is conducted to validate the architecture of SPP-Net. In this experiment, the SPP-Net is evaluated with the following architecture settings:



(a) ConvNet (b) Single-maxpool (c) Multiple-maxpool (d) SPP-Net

Figure 5: Top row: the results with different architecture settings–ConvNet is a conventional feed forward network acting on the sorted sparse descriptors. Single-maxpool and Multiple-maxpool are when only a single maxpooling unit at level-0 and multiple maxpooling at level-2 is used. We observe better performance when we combine those in SPP-Net. Bottom row: 1D representation of different architectures where the convolutions and maxpooling unites are represented by horizontal lines and triangles respectively. The global max-pooling is colored by red and other maxpooling unites are colored by blue.

- ConvNet: conventional feed forward network with convolution layers and max-pooling layers are stacked one after another (same number of layers and parameters as SPP-Net) acting on the sorted 2D array of keypoints.
- Single maxpooling: a single maxpooling layer at level 0,
- Multiple maxpooling: one maxpooling layer at level 2,
- SPP-Net: concatenate responses at three different levels.

In Figure 5, we display the results with the different choices of the architectures where we observe best performance with SPP-Net. Note that no synthetic data used in this case.

5.2. Visualizing leveraged image features

It is instructive to visualize which keypoints extracted in the image are eventually most relevant to predict the pose pa-

[1] Note that except the current experiment, the synthetic feature descriptors do not incorporate any information of the 6 d.o.f. test poses.

[2] github.com/kentsommer/keras-posenet

Scene	Area or Volume	Active Search (SIFT) [27]	Original PoseNet [13]	PoseNet LSTM [38]	PoseNet Geo. Cost [12]	SPP-Net	SPP-Net (with Synthetic data)
Great Court	$8000m^2$	–	–	–	6.83m, 3.47°	13.2m, 8.02°	5.42m, 2.84°
King's College	$5600m^2$	0.42m, 0.55°	1.66m, 4.86°	0.99m, 3.65°	0.88m, 1.04°	1.91m, 2.36°	0.74m, 0.96°
Old Hospital	$2000m^2$	0.44m, 1.01°	2.62m, 4.90°	1.51m, 4.29°	3.20m, 3.29°	2.51m, 3.74°	2.18m, 3.92°
Shop Facade	$875m^2$	0.12m, 0.40°	1.41m, 7.18°	1.18m, 7.44°	0.88m, 3.78°	1.31m, 7.82°	0.59m, 2.53°
StMary's Church	$4800m^2$	0.19m, 0.54°	2.45m, 7.96°	1.52m, 6.68°	1.57m, 3.32°	3.21m, 6.97°	1.83m, 3.35°
Street	$50000m^2$	0.85m, 0.83°	–	–	20.3m, 25.5°	54.9m, 37.2°	24.5m, 23.8°
Chess	$6m^3$	0.04m, 1.96°	0.32m, 6.60°	0.24m, 5.77°	0.13m, 4.48°	0.22m, 7.61°	0.12m, 4.42°
Fire	$2.5m^3$	0.03m, 1.53°	0.47m, 14.0°	0.34m, 11.9°	0.27m, 11.3°	0.37m, 14.1°	0.22m, 8.84°
Heads	$1m^3$	0.02m, 1.45°	0.30m, 12.2°	0.21m, 13.7°	0.17m, 13.0°	0.22m, 14.6°	0.11m, 8.33°
Office	$7.5m^3$	0.09m, 3.61°	0.48m, 7.24°	0.30m, 8.08°	0.19m, 5.55°	0.32m, 10.0°	0.16m, 4.99°
Pumpkin	$5m^3$	0.08m, 3.10°	0.49m, 8.12°	0.33m, 7.00°	0.26m, 4.75°	0.47m, 10.2°	0.21m, 4.89°
Red Kitchen	$18m^3$	0.07m, 3.37°	0.58m, 7.54°	0.24m, 5.52°	0.23m, 5.35°	0.34m, 11.3°	0.21m, 4.76°
Stairs	$7.5m^3$	0.03m, 2.22°	0.48m, 13.1°	0.40m, 13.7°	0.35m, 12.4°	0.40m, 13.2°	0.22m, 7.17°

Table 3: Median localization results for the Cambridge Landmarks [13] and seven Scenes datasets [6]. We compare the performance of various training-based algorithms (some baseline entries are copied from [38]). Active search [27] is a sota traditional geometry based baseline. We demonstrate a notable improvement over the original PoseNet [13] and re-projection error proposed in this paper, narrowing the margin to the geometry based techniques.

rameters. We define the contribution of a feature to pose prediction as the number of max-pooling units where the given feature is the winning branch in the max-pooling step. The higher the contribution, the more prominent is this feature represented in the following pose regression layers. Due to the sampling step to choose a single feature from multiple ones falling into the same cell (recall Sec. 3), there is an intrinsic randomness in the network output and in which features are relevant. Hence, in the following we consider average contribution computed from 100 runs.

In Fig 6 we display the most contributing feature points for two complementary scenes. For outdoor environments many features relevant for pose prediction cluster near the skyline induced by building, and for indoor scenarios one generally observes a mix between distinctive small-scale features and background features at a larger keypoint scale. Further, in Fig. 7, we display a pair of images where more than 50% of bins (cells) are empty yet SPP-Net successfully estimates the pose. This indicates that SPP-Net shows robustness to unevenly distributed image features.

A video (`chess.mov`[3]) is uploaded that visualizes the "Chess" sequence with overlaid features. The relevance of features is determined and visualized as in Fig. 6. A relatively small and also temporally coherent set of salient features is chosen by SPP-Net for pose estimation.

5.3. Benchmarking localization accuracy

Baseline Methods We compare the proposed SPP-Net against the following baselines:

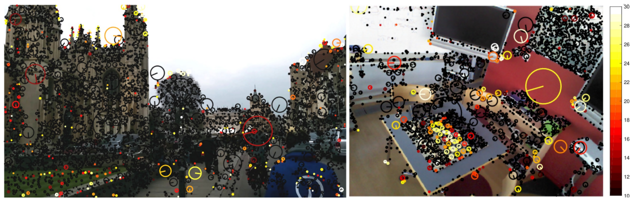
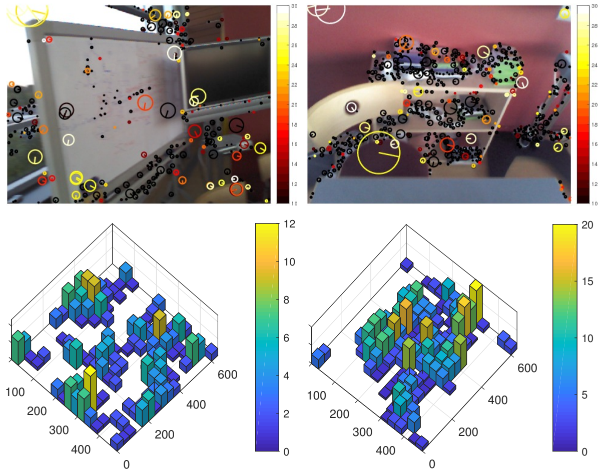


Figure 6: The keypoints for images from the "Kings College" and "Chess" sequences are displayed. Highly relevant feature points (with average contributions > 10) are colored (using the "hot" colormap) according to their contributions to the ensemble layer (see text). In general, feature descriptors at larger scales seem to be more relevant. Further, in the King's college sequence, keypoints near the building outlines are relatively consistently important for pose prediction. The indoor "Chess" sequence exhibits a mix of features on the unique chess pieces and on the background.

- Active Search [27]: This is a direct feature-based approach where the feature descriptors are matched across the 3D point-cloud and the absolute pose is estimated robustly utilizing the P3P algorithm.
- Original PoseNet [13]: The first convnet-based method where the last soft-max classification layer of GoogleNet is replaced by the fully connected regression layers.
- PoseNet LSTM [38] : Similar as above, but multiple LSTM units were utilized to the convnet features followed by a regression layers.
- PoseNet Geometric Cost [12] (PoseNet2): The network

[3] https://youtu.be/Fuv18OMaTnk



(a) Angular Error = 12.71°, Positional Error = 0.36m

(b) Angular Error = 6.52°, Positional Error = 0.21m

Figure 7: A pair of typical test images of the "chess" sequences are displayed along with the histograms where 56.2% and 51.8% cells of the 16 × 16 grids are empty. Although, our method predicts the pose successfully, it is bit worse than the median prediction mentioned in table 3.

is trained with the same loss function as ours and fine-tuned with the re-projection cost.

- Proposed SPP-Net trained without augmented posses is also included as baseline.

Cambridge Landmarks Datasets The datasets [13] provides a labeled set of image sequences of different outdoor scenes where the ground-truth poses were obtained by utilizing VisualSFM [39]. The datasets also provide the SfM "reconstruction"($.nvm$) files containing the 3D point-cloud and the 2D-3D assignments required by our pose augmentation. Creation of synthetic images takes approximately two hours for a typical dataset (per sequence). The SPP-Net is trained on the augmented (training and synthesized) dataset. The results are displayed in Table 3. The SPP-Net produces comparable results with the original PoseNet and comparable with PoseNet2 [12] once trained on an augmented dataset, however, SPP-Net is more lightweight, much faster and not required to be pre-trained on a larger datasets. Note that the proposed network is of limited size, increasing the size of the network (and essentially size of the augmented poses) will further improve the performance.

Seven Scene Datasets The Microsoft 7-Scenes Dataset [30] consists of texture-less RGB-D images of seven different indoor scenes. The scenes were captured with a Kinect device and reference poses were obtained using KincetFusion [9]. The dataset was originally constructed for RGB-D based localization as the scene is extremely challenging for direct approach with the texture-less feature descriptors.

The 3D map and the feature descriptors are not provided with the datasets required by the proposed augmented pose generation technique. Thus, we reconstructed the 3D point cloud from scratch using toolboxes such as VisualSFM [39] and COLMAP [29]. We register the SfM camera poses to the KincetFusion reference poses by a similarity transformation, and the same transformation is used to register the 3D points w.r.t. the reference poses. The points obtained by SfM reconstruction are refined using the reference poses. Note that as the reference poses are rather noisy, hence the similarity transformation does not relate SfM poses and reference poses well in all scenes. In particular, we observed good results on "Stairs", "Heads" and "Fire" sequences as the similarity transformation is a good fit for these scenes as shown in Table 3. Overall we have obtained very competitive results in this dataset.

	SPP-Net (0.25×)	SPP-Net	SPP-Net (4×)
Chess	0.15m, 4.89°	0.12m, 4.42°	0.10m, 3.36°
Fire	0.28m, 12.4°	0.22m, 8.84°	0.21m, 8.35°
Heads	0.14m, 10.7°	0.11m, 8.33°	0.11m, 8.06°
Office	0.19m, 6.15°	0.16m, 4.99°	0.13m, 4.07°
Pumpkin	0.34m, 8.47°	0.21m, 4.89°	0.20m, 5.35°
Red Kit.	0.26m, 5.16°	0.21m, 4.76°	0.22m, 5.29°
Stairs	0.25m, 7.38°	0.22m, 7.17°	0.20m, 7.25°

Table 4: Evaluation of SPP-Net with varying number of parameters on seven Scenes datasets.

5.4. Varying network size

This experiments aims to determine the sensitivity of the SPP-Net architecture to the number of network parameters. We consider two modifications for the network size:

- half the number of feature channels used in convolutional and fully connected layers of SPP-Net,
- conversely, double the number of all feature channels and channels in the fully connected layers.

As a result we have about one fourth and 4× number of parameters, respectively, compared to our standard SPP-Net. The above networks are trained on the augmented poses of the seven Scenes datasets. The results are displayed in Table 4 and indicate, that the performance of the smaller network is degrading relatively gracefully, whereas the larger network offers insignificant gains (and it seems to show some signs of overfitting).

In Table 4, we display the results on Cambridge Landmark Datasets [13] where we observe similar performance as above. It improves the performance with the size of the network for most of the sequence, except the sequence "Shop Facade". Again, we believe that in this case the larger network starts to overfit on this smaller dataset.

	Ours SPP-Net (0.25×)	Ours SPP-Net	Ours SPP-Net (4×)
Great Court	7.58m, 5.91°	5.42m, 2.84°	5.48m, 2.77°
King's Coll.	1.41m, 2.02°	0.74m, 0.96°	0.83m, 1.01°
Old Hosp.	2.06m, 3.91°	2.18m, 3.92°	1.83m, 3.25°
Shop Facade	0.87m, 3.36°	0.59m, 2.53°	0.64m, 3.05°
StMary's Ch.	2.26m, 6.46°	1.83m, 3.35°	1.62m, 3.42°
Street	33.9m, 31.2°	24.5m, 23.8°	17.5m, 20.2°

Table 5: Evaluation of SPP-Net with varying number of parameters on Cambridge Landmark datasets [13].

6. Conclusion

In this work we presented a deep learning architecture for pose prediction able to learn from real and synthesized views. Thus, pose regression can be trained for any region in the space of all poses using a virtually unlimited amount of (synthetic) training data. Our proposed method to create synthetic data aims to be sufficiently realistic by using an underlying 3D point cloud and an outlier and noise generation model. We performed a number of numerical experiments to validate our architecture and the proposed augmentation procedure, and we achieve state-of-the-art results on benchmark datasets for pose regression.

References

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. Corrado, A. Davis, J. Dean, M. Devin, et al. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. 2015. 5

[2] A. Angeli, D. Filliat, S. Doncieux, and J.-A. Meyer. Fast and incremental method for loop-closure detection using bags of visual words. *IEEE Transactions on Robotics*, 24(5):1027–1037, 2008. 2

[3] L. Clement and J. Kelly. How to train a cat: Learning canonical appearance transformations for robust direct localization under illumination change. *arXiv preprint arXiv:1709.03009*, 2017. 2

[4] M. Cummins and P. Newman. Fab-map: Probabilistic localization and mapping in the space of appearance. *The International Journal of Robotics Research*, 27(6):647–665, 2008. 2

[5] A. Dosovitskiy and T. Brox. Inverting visual representations with convolutional networks. In *Proc. CVPR*, pages 4829–4837, 2016. 3

[6] B. Glocker, S. Izadi, J. Shotton, and A. Criminisi. Real-time rgb-d camera relocalization. In *Mixed and Augmented Reality (ISMAR), 2013 IEEE International Symposium on*, pages 173–179. IEEE, 2013. 1, 7

[7] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *Proc. ECCV*, pages 346–361. Springer, 2014. 2, 4

[8] A. Irschara, C. Zach, J.-M. Frahm, and H. Bischof. From structure-from-motion point clouds to fast location recognition. In *Proc. CVPR*, pages 2599–2606. IEEE, 2009. 1, 2, 4, 5

[9] S. Izadi, D. Kim, O. Hilliges, D. Molyneaux, R. Newcombe, P. Kohli, J. Shotton, S. Hodges, D. Freeman, A. Davison, et al. Kinectfusion: real-time 3d reconstruction and interaction using a moving depth camera. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, pages 559–568. ACM, 2011. 8

[10] H. Jegou, M. Douze, and C. Schmid. Hamming embedding and weak geometric consistency for large scale image search. *Proc. ECCV*, pages 304–317, 2008. 2

[11] H. Jégou, M. Douze, and C. Schmid. Improving bag-of-features for large scale image search. *International journal of computer vision*, 87(3):316–336, 2010. 2

[12] A. Kendall and R. Cipolla. Geometric loss functions for camera pose regression with deep learning. In *Proc. CVPR*, 2017. 1, 2, 4, 7, 8

[13] A. Kendall, M. Grimes, and R. Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *Proc. ICCV*, pages 2938–2946, 2015. 1, 2, 3, 4, 6, 7, 8, 9

[14] D. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015. 5

[15] 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. 2, 4

[16] Y. Li, N. Snavely, and D. P. Huttenlocher. Location recognition using prioritized feature matching. In *Proc. ECCV*, pages 791–804. Springer, 2010. 1, 2

[17] Y. Li, N. Snavely, D. P. Huttenlocher, and P. Fua. Worldwide pose estimation using 3d point clouds. In *Large-Scale Visual Geo-Localization*, pages 147–163. Springer, 2016. 2

[18] D. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, 2004. 2

[19] I. Melekhov, J. Ylioinas, J. Kannala, and E. Rahtu. Image-based localization using hourglass networks. *arXiv preprint arXiv:1703.07971*, 2017. 2, 4

[20] P. Newman and K. Ho. Slam-loop closing with visually salient features. In *Proc. ICRA*, pages 635–642. IEEE, 2005. 2

[21] D. Nister and H. Stewenius. Scalable recognition with a vocabulary tree. In *Proc. CVPR*, volume 2, pages 2161–2168. Ieee, 2006. 2

[22] F. Perronnin, Y. Liu, J. Sánchez, and H. Poirier. Large-scale image retrieval with compressed fisher vectors. In *Proc. CVPR*, pages 3384–3391. IEEE, 2010. 2

[23] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *Proc. CVPR*, pages 1–8. IEEE, 2007. 2

[24] C. R. Qi, H. Su, K. Mo, and L. J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation. *arXiv preprint arXiv:1612.00593*, 2016. 3

[25] T. Sattler, M. Havlena, K. Schindler, and M. Pollefeys. Large-scale location recognition and the geometric burstiness problem. In *Proc. CVPR*, pages 1582–1590, 2016. 4

[26] T. Sattler, B. Leibe, and L. Kobbelt. Fast image-based localization using direct 2d-to-3d matching. In *Proc. ICCV*, pages 667–674. IEEE, 2011. 2

[27] T. Sattler, B. Leibe, and L. Kobbelt. Efficient & effective prioritized matching for large-scale image-based localization. *IEEE Trans. on Pattern Anal. and Mach. Intell.*, 39(9):1744–1756, 2017. 1, 2, 7

[28] G. Schindler, M. Brown, and R. Szelisk. City-scale location recognition. In *Proc. CVPR*, 2007. 1, 2

[29] J. L. Schonberger, F. Radenovic, O. Chum, and J.-M. Frahm. From single image query to detailed 3d reconstruction. In *Proc. CVPR*, pages 5126–5134, 2015. 8

[30] J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi, and A. Fitzgibbon. Scene coordinate regression forests for camera relocalization in rgb-d images. In *Proc. CVPR*, pages 2930–2937, 2013. 2, 5, 8

[31] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *Proc. ICLR*, 2015. 2

[32] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proc. CVPR*, pages 1–9, 2015. 2

[33] G. Tolias, R. Sicre, and H. Jégou. Particular object retrieval with integral max-pooling of cnn activations. *arXiv preprint arXiv:1511.05879*, 2015. 3

[34] A. Torii, R. Arandjelovic, J. Sivic, M. Okutomi, and T. Pajdla. 24/7 place recognition by view synthesis. In *Proc. CVPR*, pages 1808–1817, 2015. 4

[35] A. Torralba, R. Fergus, and Y. Weiss. Small codes and large image databases for recognition. In *Proc. CVPR*, pages 1–8. IEEE, 2008. 2

[36] F. Walch, C. Hazirbas, L. Leal-Taixé, T. Sattler, S. Hilsenbeck, and D. Cremers. Image-based localization with spatial lstms. In *Proc. ICCV*, 2017. 1, 2

[37] Y. Weiss, A. Torralba, and R. Fergus. Spectral hashing. In *Proc. NIPS*, pages 1753–1760, 2009. 2

[38] T. Weyand, I. Kostrikov, and J. Philbin. Planet-photo geolocation with convolutional neural networks. In *Proc. ECCV*, pages 37–55. Springer, 2016. 1, 4, 7

[39] C. Wu et al. Visualsfm: A visual structure from motion system. In . 8

[40] W. Zhang and J. Kosecka. Image based localization in urban environments. In *3D Data Processing, Visualization, and Transmission, Third International Symposium on*, pages 33–40. IEEE, 2006. 1, 2