Synthetic View Generation for Absolute Pose Regression and Image Synthesis: Supplementary material

Pulak Purkait[1]
pulak.cv@gmail.com
Cheng Zhao[2]
irobotcheng@gmail.com
Christopher Zach[1]
christopher.m.zach@gmail.com

[1] Toshiba Research Europe Ltd.
Cambridge, UK

[2] University of Birmingham
Birmingham, UK

Contents

1 The network architecture of proposed SPP-Net

As shown in Figure 1, 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. 1(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

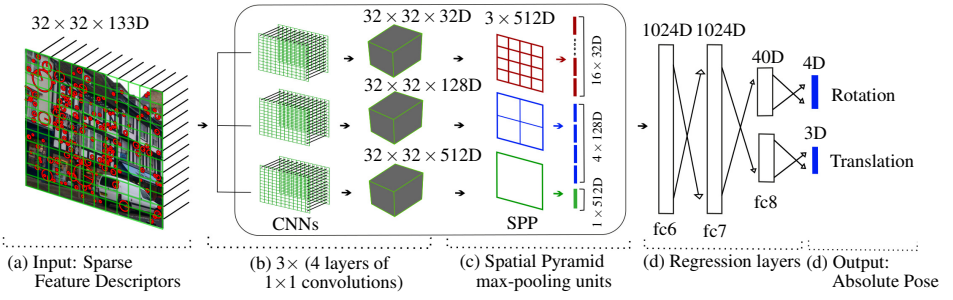


Figure 1: Proposed SPP-Net for absolute pose regression takes sparse feature points as input and predicts the absolute pose.

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. 1) 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.

2 Validation of Different Steps

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 [3], utilizing a tensorflow implementation available online [1], 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 3. 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:

- 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.

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

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) [3]	8.9M	1.6B
Baseline (ResNet50) [4, 5]	26.5M	3.8B
PoseNet LSTM [7]	9.0M	1.6B

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

- 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 3, 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.

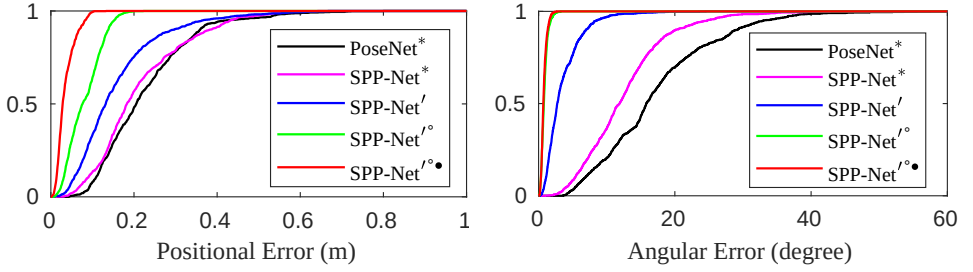
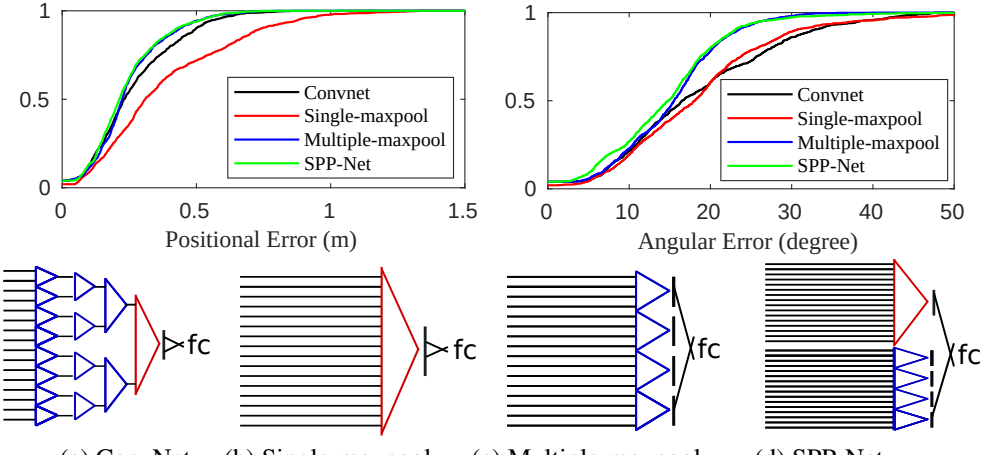


Figure 2: Left-Right: demonstrate our localization accuracy for both position and orientation as a cumulative histogram of errors for the entire testing set. Where the baselines—Net*: trained with the training data only, Net$'$: trained with the clean synthetic data, Net$'^{\circ}$: trained with the synthetic data under realistic noise, Net$'^{\circ\bullet}$: trained with the synthetic data under realistic noise and outliers.



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

Figure 3: 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.

	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 Kitchen	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 3: Evaluation of SPP-Net with varying number of parameters on seven Scenes datasets.

3 More Visualizations

A video (`chess.mov`[2]) is uploaded that visualizes the "Chess" sequence with overlaid features. The relevance of features is determined and visualized as in Fig. 6 in the main text. A relatively small and also temporally coherent set of salient features is chosen by SPP-Net for pose estimation.

4 Pose Regression 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 3 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 over-fitting).

In Table 4, we display the results on Cambridge Landmark Datasets [3] 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.

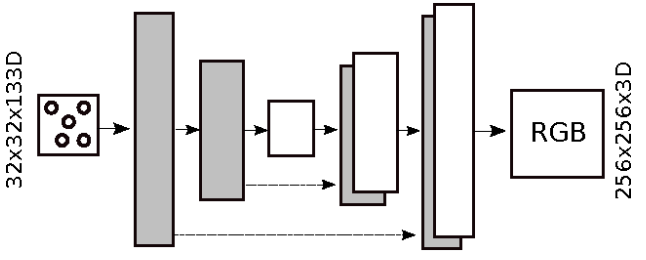
5 Architectures of the RGB image synthesis technique

The proposed architecture is displayed in Fig. 4. The generator has an U-Net architecture consists of a number of skip connections. Note that our input is a sparse descriptor of size $32 \times 32 \times 133D$ and the output is a RGB image of size $256 \times 256 \times 3$. Thus the skip connections are performed with feature descriptors of sizes 16×16 8×8 and 4×4 only. The

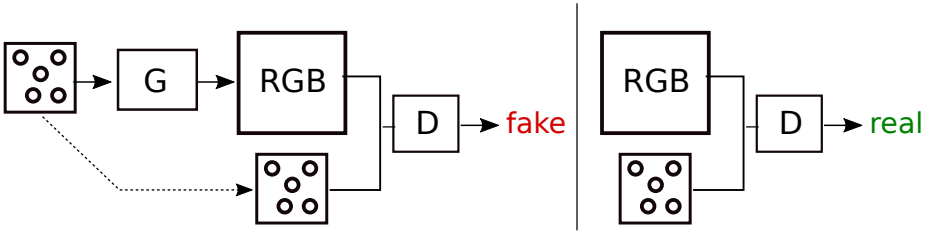
[2] https://youtu.be/Fuv18OMaTnk

	SPP-Net (0.25×)	SPP-Net	SPP-Net (4×)
Great Court	7.58m, 5.91°	5.42m, 2.84°	5.48m, 2.77°
King's College	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 Church	2.17m, 5.61°	1.44m, 3.31°	1.42m, 3.28°
Street	33.9m, 31.2°	24.5m, 23.8°	17.5m, 20.2°

Table 4: Evaluation of SPP-Net with varying number of parameters on Cambridge Landmark datasets [4].



(a) Generator (G) network used for ℓ_2 [4].



(b) Training a conditional GAN to map sparse feature descriptors to RGB image.

Figure 4: Proposed architectures for RGB image synthesis.

discriminaor network takes RGB image and sparse descripors both as input followed by separate convolution layers. The stream pairs are concatenated just before the last layer. The networks are trained simultaniously from scratch.

6 More Results on RGB image synthesis

More results on RGB image synthesis are displayed in Fig. 5 and Fig. 6. We observe that our GAN based RGB image generation produces consistent results. Note that we have displayed the consecutive frames—which are not some cherry picked examples.

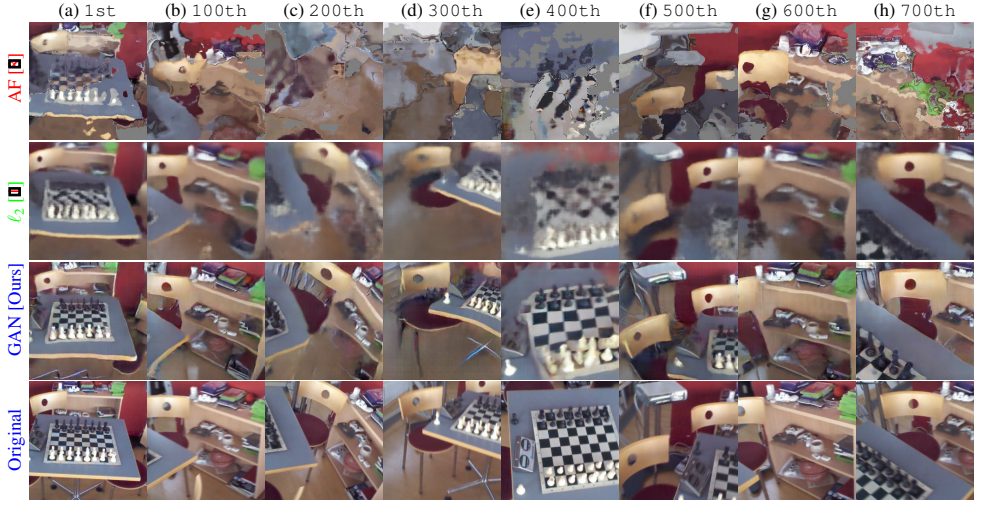


Figure 5: RGB images synthesized by different methods at the test poses of the chess image sequence of 7-Scenes Dataset [6]. The indices of the images of the test sequence are mentioned in the top of the figure.

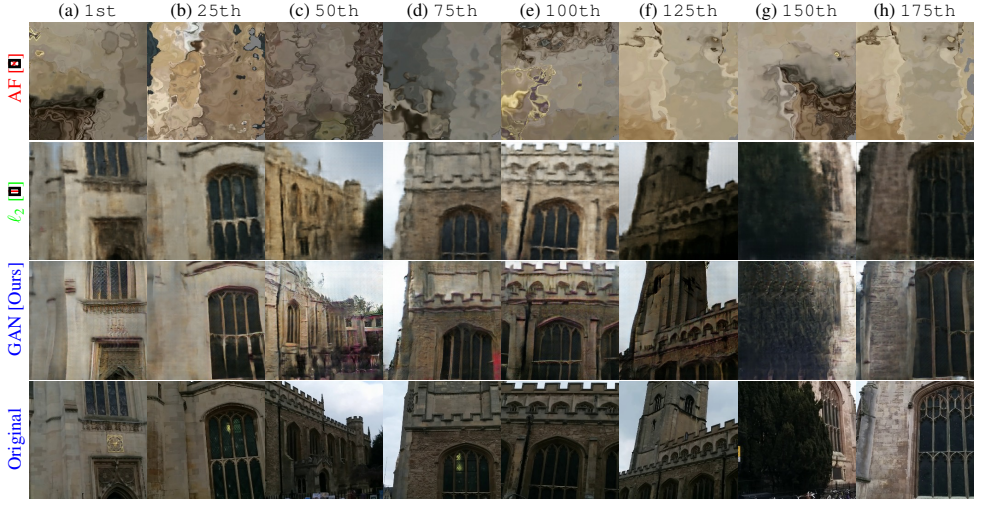


Figure 6: RGB images synthesized by different methods at the test poses of the "StMary's Church" image sequence of Cambridge Dataset [3]. The indices of the images of the test sequence are mentioned in the top of the figure.

References

[1] Alexey Dosovitskiy and Thomas Brox. Inverting visual representations with convolutional networks. In *Proc. CVPR*, pages 4829–4837, 2016.

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *Proc. ECCV*, pages 346–361. Springer, 2014.

[3] Alex Kendall, Matthew Grimes, and Roberto Cipolla. Posenet: A convolutional network for real-time 6-dof camera relocalization. In *Proc. ICCV*, pages 2938–2946, 2015.

[4] Zakaria Laskar, Iaroslav Melekhov, Surya Kalia, and Juho Kannala. Camera relocalization by computing pairwise relative poses using convolutional neural network. *Proc. ICCV Workshops*, 2017.

[5] Iaroslav Melekhov, Juha Ylioinas, Juho Kannala, and Esa Rahtu. Image-based localization using hourglass networks. *Proc. ICCV Workshops*, 2017.

[6] 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 *Proc. CVPR*, pages 2930–2937, 2013.

[7] Tobias Weyand, Ilya Kostrikov, and James Philbin. Planet-photo geolocation with convolutional neural networks. In *Proc. ECCV*, pages 37–55. Springer, 2016.

[8] Tinghui Zhou, Shubham Tulsiani, Weilun Sun, Jitendra Malik, and Alexei A Efros. View synthesis by appearance flow. In *Proc. ECCV*, pages 286–301. Springer, 2016.