Supplementary Material: Learning to Estimate 3D Hand Pose from Single RGB Images

Christian Zimmermann, Thomas Brox
University of Freiburg
{zimmermann, brox}@cs.uni-freiburg.de

Supplementary

A. HandSegNet architecture and learning schedule

Table 1 contains the architecture used for *HandSegNet*. It was trained for hand segmentation on *R-train* with a batch size of 8 and using ADAM solver [1]. The network was initialized using weights of Wei *et al*. [2] for layers 1 to 16 and then trained for 40000 iterations using a standard softmax cross-entropy loss. The learning rate was $1 \cdot 10^{-5}$ for the first 20000 iterations, $1 \cdot 10^{-6}$ for following 10000 iterations and $1 \cdot 10^{-7}$ until the end. Except for random color hue augmentation of 0.1 no data augmentation was used. From the 320×320 pixel images of the training set a 256×256 crop was taken randomly.

B. PoseNet architecture and learning schedule

Table 2 contains the architecture used for *PoseNet*. In all cases it was trained with a batch size of 8 and using ADAM solver [1]. The initial 16 layers of the network are initialized using weights of Wei *et al*. [2] all others are randomly initialized . The network is trained for 30000 iterations using a L_2 loss. The learning rate is $1 \cdot 10^{-4}$ for the first 10000 iterations, $1 \cdot 10^{-5}$ for following 10000 iterations and $1 \cdot 10^{-6}$ until the end. For ground truth generation of the score maps we use normal distributions with a variance of 25 pixels and the mean being equal to the given keypoint location. We normalize the resulting maps such that each map contains values from 0 to 1, if there is a keypoint visible. For invisible keypoints the map is zero everywhere.

We train *PoseNet* on axis aligned crops that are resized to a resolution of 256×256 pixels by bilinear interpolation. The bounding box is chosen such that all keypoints of a single hand are contained within the crop. We augment the cropping procedure by modifying the calculated bounding box in two ways. First, we add noise to the calculated center of the bounding box, which is sampled from a zero mean normal distribution with variance of 10 pixels. The size of the bounding box is changed accordingly to still contain all hand keypoints. Second we find it helpful to improve generalization performance by adding a bit of noise on the coordinates used to generate the score maps. Therefore, we add a normal distribution of zero mean and variance 1.5 to the ground truth keypoint coordinates, whereas each keypoint is sampled independently. Additionally we apply random contrast augmentation with a scaling factor between 0.5 and 1.0, which is sampled from a uniform distribution.

id	Name	Kernel	Dimensionality
	Input image	-	$256 \times 256 \times 3$
1	Conv. + ReLU	3×3	$256 \times 256 \times 64$
2	Conv. + ReLU	3×3	$256 \times 256 \times 64$
3	Maxpool	4×4	$128 \times 128 \times 64$
4	Conv. + ReLU	3×3	$128 \times 128 \times 128$
5	Conv. + ReLU	3×3	$128 \times 128 \times 128$
6	Maxpool	4×4	$64 \times 64 \times 128$
7	Conv. + ReLU	3×3	$64 \times 64 \times 256$
8	Conv. + ReLU	3×3	$64 \times 64 \times 256$
9	Conv. + ReLU	3×3	$64 \times 64 \times 256$
10	Conv. + ReLU	3×3	$64 \times 64 \times 256$
11	Maxpool	4×4	$32 \times 32 \times 256$
12	Conv. + ReLU	3×3	$32 \times 32 \times 512$
13	Conv. + ReLU	3×3	$32 \times 32 \times 512$
14	Conv. + ReLU	3×3	$32 \times 32 \times 512$
15	Conv. + ReLU	3×3	$32 \times 32 \times 512$
16	Conv. + ReLU	3×3	$32 \times 32 \times 512$
17	Conv.	1×1	$32 \times 32 \times 2$
18	Bilinear Upsampling	-	$256 \times 256 \times 2$
19	Argmax	-	$256 \times 256 \times 1$
	Hand mask	-	$256 \times 256 \times 1$

Table 1: Network architecture of the proposed *HandSegNet* network. Except for input and hand mask output every row of the table gives a data tensor of the network and the operations that produced it.

id	Name	Kernel	Dimensionality
	Input image	-	$256\times256\times3$
1	Conv. + ReLU	3×3	$256\times256\times64$
2	Conv. + ReLU	3×3	$256\times256\times64$
3	Maxpool	4×4	$128\times128\times64$
4	Conv. + ReLU	3×3	$128\times128\times128$
5	Conv. + ReLU	3×3	$128\times128\times128$
6	Maxpool	4×4	$64\times64\times128$
7	Conv. + ReLU	3×3	$64\times64\times256$
8	Conv. + ReLU	3×3	$64\times64\times256$
9	Conv. + ReLU	3×3	$64\times64\times256$
10	Conv. + ReLU	3×3	$64\times64\times256$
11	Maxpool	4×4	$32\times32\times256$
12	Conv. + ReLU	3×3	$32\times32\times512$
13	Conv. + ReLU	3×3	$32\times32\times512$
14	Conv. + ReLU	3×3	$32\times32\times512$
15	Conv. + ReLU	3×3	$32\times32\times512$
16	Conv. + ReLU	3×3	$32\times32\times512$
17	Conv.	1×1	$32\times32\times21$
18	Concat(16, 17)	-	$32\times32\times533$
19	Conv. + ReLU	7×7	$32\times32\times128$
20	Conv. + ReLU	7×7	$32\times32\times128$
21	Conv. + ReLU	7×7	$32\times32\times128$
22	Conv. + ReLU	7×7	$32\times32\times128$
23	Conv. + ReLU	7×7	$32\times32\times128$
24	Conv.	1×1	$32\times32\times21$
25	Concat(16, 17, 24)	-	$32\times32\times554$
26	Conv. + ReLU	7×7	$32\times32\times128$
27	Conv. + ReLU	7×7	$32\times32\times128$
28	Conv. + ReLU	7×7	$32\times32\times128$
29	Conv. + ReLU	7×7	$32\times32\times128$
30	Conv. + ReLU	7×7	$32\times32\times128$
31	Conv.	1×1	$32\times32\times21$

Table 2: Network architecture of the *PoseNet* network. Except for input every row of the table represents a data tensor of the network and the operations that produced it. Outputs of the network are are predicted score maps $\mathbf{c}$ from layers 17, 24 and 31.

C. PosePrior architecture

Table 3 contains the architecture used for each stream of the *PosePrior*. It uses 6 convolutional layers followed by two fully-connected layers. All use ReLU activation function and the fully-connected layers have a dropout probability of 0.2 to randomly drop a neuron. Preceeding to the first FC layer, information about the hand side is concatenated to the flattened feature representation calculated by the convolutional layers. All drops in spatial dimension are due to strided convolutions. The network ceases with a fully-connected layer that estimates P parameters, where $P = 3$ for Viewpoint estimation and $P = 63$ for the coordinate estimation stream.

id	Name	Kernel	Dimensionality
	Input	-	$32\times32\times21$
1	Conv. + ReLU	3×3	$32\times32\times32$
2	Conv. + ReLU	3×3	$16\times16\times32$
3	Conv. + ReLU	3×3	$16\times16\times64$
4	Conv. + ReLU	3×3	$8\times8\times64$
5	Conv. + ReLU	3×3	$8\times8\times128$
6	Conv. + ReLU	3×3	$4\times4\times128$
7	Reshape + Concat	-	130
8	FC + ReLU + Drop(0.2)	-	512
9	FC + ReLU + Drop(0.2)	-	512
10	FC	-	512
	Output	-	P

Table 3: Network architecture of a single stream for the proposed *PosePrior* network. Except for input and output every row of the table gives a data tensor of the network and the operations that produced it. Reduction in the spatial dimension is due to stride in the convolutions. P is the number of estimated parameters and is $P = 3$ for Viewpoint estimation and $P = 63$ for the coordinate estimation stream.

D. GestureNet architecture and learning schedule

We train the *GestureNet* using Adam solver, a batch size of 8 and an initial learning rate of $1 \cdot 10^{-4}$ which drops by one decade at 15000 and 20000 iterations. Training is finished at iteration 30000. The network is trained with a standard softmax cross-entropy loss on randomly cropped 256×256 images.

E. Additional results

Figure 1 shows results of the proposed approach.

id	Name	Dimensionality
	Input $\mathbf{c}^{\text{rel}}$	63
1	FC + ReLU + Dropout(0.2)	512
2	FC + ReLU + Dropout(0.2)	512
3	FC	35

Table 4: Network architecture of the *GestureNet* used for our experiments. All layers were initialized randomly. Probability to drop a neuron in the indicated layers is set to 0.2.

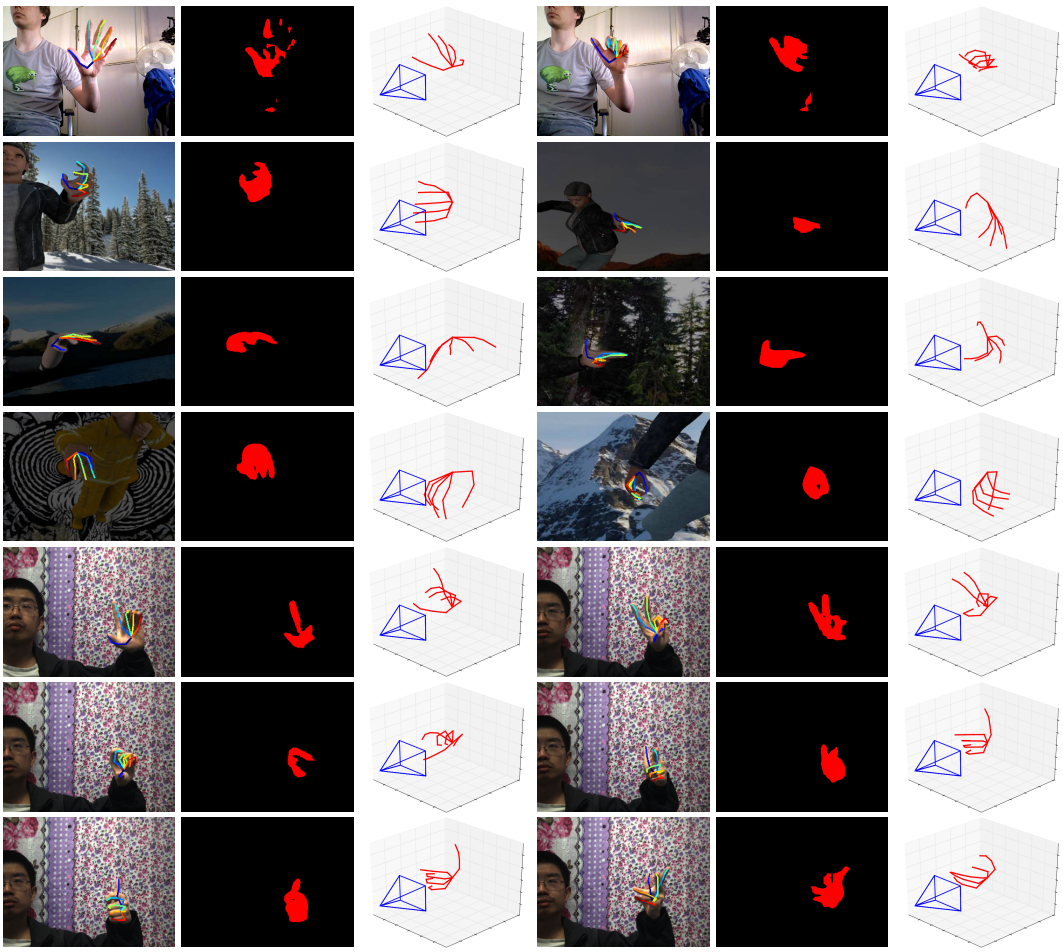

Figure 1: Qualitative examples of our complete system. Input to the network are color image and information if its a left or right hand. The network estimates the hand segmentation mask, localizes keypoints in 2D (shown overlayed with the input image) and outputs the most likely 3D pose. The top row shows samples from a dataset we recorded for qualitative evaluation, the following three rows are from *R-val* and last three rows are from *S-val*.

Figure 2: Two samples from the NYU Hand Pose Dataset by Tompson *et al*. [3]. Due to artefacts in the color images this dataset is not suited to evaluate color based approaches.

F. NYU Hand Pose Dataset

A commonly used benchmark for 3D hand pose estimation is the NYU Hand Pose Dataset by Tompson *et al*. [3]. We can't use it for our work, because it only provides registered color images, which exclusively provide color information for pixels with valid depth data. This results into corrupted images as shown in Figure 2. This makes it infeasible to use for an approach that only utilizes color.

References

[1] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[2] Shih-En Wei, Varun Ramakrishna, Takeo Kanade, and Yaser Sheikh. Convolutional pose machines. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4724–4732, 2016.

[3] J. Tompson, M. Stein, Y. Lecun, and K. Perlin. Real-Time Continuous Pose Recovery of Human Hands Using Convolutional Networks. *ACM Transactions on Graphics*, 33(5):1–10, Sept. 2014.