A Framework for Articulated Hand Pose Estimation and Evaluation

Gernot Riegler(✉), David Ferstl(✉), Matthias Rüther(✉), and Horst Bischof(✉)

Institute for Computer Graphics and Vision, Graz University of Technology, Inffeldgasse 16, 8010 Graz, Austria
{riegler,ferstl,ruether,bischof}@icg.tugraz.at.com

Abstract. We present in this paper a framework for articulated hand pose estimation and evaluation. Within this framework we implemented recently published methods for hand segmentation and inference of hand postures. We further propose a new approach for the segmentation and extend existing convolutional network based inference methods. Additionally, we created a new dataset that consists of a synthetically generated training set and accurately annotated test sequences captured with two different consumer depth cameras. The evaluation shows that we can improve with our methods the state-of-the-art. To foster further research, we will make all sources and the complete dataset used in this work publicly available.

Keywords: Pose estimation · Random forest · Convolutional networks · Segmentation · Synthetic data

1 Introduction

The availability of affordable depth sensors in the consumer market, especially beginning with the first Microsoft Kinect, provided new ways for human computer interaction. This hardware combined with data-driven methods [10,22,23] enabled the estimation of human body pose in real-time which is now used for example in video games. The latest depth sensors, especially those based on the time-of-flight principle, have a smaller package size than ever. They can be integrated into notebooks, tablets, or bundled with a virtual reality headset, which allows new and interesting possibilities for natural interactions by hand gestures and movements.

Although the pose estimation of an articulated hand shares similarities with body pose estimation, it remains a challenging problem. First of all, a human hand has many degrees of freedom. This implies also the possibilities for a high degree of self occlusions. For instance, only a few joints are visible in the depth map of a fist gesture. Further, depending on the viewpoint, one hand pose can be mapped to significantly different depth maps. An additional difficulty is inherited from the sensors themselves. The output of a consumer depth sensor is typically of low resolution and contains noise of different distributions.

R.R. Paulsen and K.S. Pedersen (Eds.): SCIA 2015, LNCS 9127, pp. 41–52, 2015.
DOI: 10.1007/978-3-319-19665-7_4

Current methods for articulated hand pose estimation can be roughly divided into two areas. Model based tracking methods such as [1,11,16,20] try to fit a known hand-model to the depth data and track it over time. Whereas discriminative approaches like [14,24–26] try to learn a mapping from the visual input of the depth sensor to a pose vector. However, a comparison between different methods remained difficult in the past, because of the lack of available datasets and source codes. Only recently, Tang *et al.* [24] and Tompson *et al.* [26] independently published datasets of varying quality.

In this work we provide a common framework for hand pose estimation. This includes interchangeable methods for hand segmentation, preprocessing, model training and inference. We re-implemented several published methods and introduce also extensions, especially for methods based on convolutional networks. Further, we propose a mean-shift method for hand segmentation that gives a very good trade-off between speed and accuracy. To overcome the limitations of existing datasets, we created a new one that contains more than $4 \cdot 10^5$ annotated training images, rendered from a realistic human model. To evaluate the different methods, we additionally annotated test sequences captured with two different depth sensors. To foster further research of articulated hand pose estimation, we will make the source code of the complete framework and the datasets publicly available at our project page: http://rvlab.icg.tugraz.at/pose.

2 Related Work

Pose estimation and the related gesture recognition of a human hand has due to its importance for human computer interaction a long history in the computer vision community. Mitra and Acharya [17] give a good overview of gesture recognition. Erol *et al.* [7] focus specifically on hand pose estimation, where the authors divide the approaches based on the data input, color images and depth maps, and between model based tracking methods and discriminative methods.

With the availability of affordable depth sensors in the consumer market, the interest in pose estimation from depth maps has increased. In the seminal work of Shotton *et al.* [22], the authors utilize a random decision forest for body pose estimation from depth data. They predict for each pixel a class probability, if it belongs to the background, or to one of the 38 defined body parts. In the same spirit, Girshick *et al.* [10] and Sun *et al.* [23] refine the random forest framework with joint regression methods and incorporating dependency relationships between output variables, respectively.

Earlier methods to estimate the articulated hand pose rely on model based tracking. Oikonomidis *et al.* [20] fit a hand model with 26 degrees of freedom to the data with particle swarm optimization. Similar, De La Gorce *et al.* [11] use a detailed hand mesh with simulated texture and lightning to fit and track a model to a monocular image stream. To handle occlusions, Balan *et al.* [1] detect finger tips as salient points and assign them to the hand jointly with the pose estimation. Further, they include edge and optical flow clues in their objective function. Melax *et al.* [16] propose a tracker based simulation of a hand

model to fit the depth data in terms of a 3D error function. They spawn multiple simulations and constrain the hand model to improve the overall accuracy.

Following the success in body pose estimation, discriminative methods have recently become popular for articulated hand pose estimation. Keskin *et al.* [14] propose to cluster the training data based on their shape with a random decision forest and train for each cluster an expert similar to [22]. In [25], Tang *et al.* train a semi-supervised random forest to combine on synthetic data in combination with sparsely labeled real depth data. The structure of the hand in a random forest framework is exploited by Tang *et al.* [24]. The authors treat the joint localization as a structured coarse to fine search by conditioning the trees on the previous estimations and a latent tree model.

In contrast to random forest based methods, approaches that rely on convolutional networks have become popular since the enormous success in object classification [15]. Hand segmentation with a convolutional network is described by Neverova *et al.* [18]. They combine synthetic and real depth data to conduct a pixel-wise prediction of 20 different hand parts. In the work of Tompson *et al.* [26] the authors train a convolutional network to predict joint locations in a coarse 2D heat-map. The depth of the joint is obtained by the depth value at the 2D joint location. An additional inverse kinematic model is utilized to refine the results.

All of the above mentioned methods use their own datasets for training and evaluation. Unfortunately, most of the authors did not publish their datasets, making a fair comparison complicated. Only very recently Tang *et al.* [24] and Tompson *et al.* [26] independently released their images along with annotations. However, these datasets have their drawbacks. The annotations provided by Tang *et al.* are located on the bone centers, rather than on the joint locations of a human hand skeleton. This prohibits refinement and regularization techniques that rely on this skeletal information. Further, the annotations were obtained by the tracking method of [16] and only little effort was put into refining the estimates. The dataset of Tompson *et al.* has a higher annotation accuracy, but lacks in variability of the training data. Only a sequence of a single user was recorded for training. The test set consists of two persons, whereby one is the same as in the training sequence.

Our synthetic dataset that is used for training provides a high annotation accuracy in terms of joint locations. To ensure a high variability in the dataset we use several size and shape variations of the animated hand. Because the training set is synthetically generated, the test sequences are guaranteed to be completely independent to the test sequences. Further, we provide test sequences captured with two different depth sensors to evaluate all methods within our framework.

3 Framework

In this section we describe the general framework we utilize for articulated hand pose estimation. We summarize the existing methods implemented in our framework and propose a new method for hand segmentation and extensions to infer the joint locations. An overview of the framework is visualized in Figure 1.

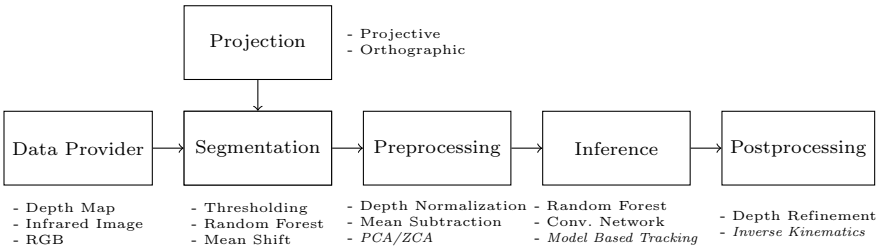


Fig. 1. Framework Overview: The main building blocks of our hand pose estimation pipeline. The data provider loads depth maps and optional infrared images from a specified dataset, or from a camera stream. The projection is an abstraction of the camera parameters and maps the pixels of a depth map from the 2D image space to 3D world coordinates and vice versa. In the next step, the hand gets segmented from the background. The segmentation results are then preprocessed including depth normalization and mean subtraction. Finally, the joint locations are predicted in an inference step and postprocessing is conducted to refine the results.

Data Provider. The data provider abstracts the access to the various sources of data. This allows the evaluation of a single method on different datasets. Further, the data provider can encapsulate a camera stream to qualitatively test a method on new data. The type of data itself depends on the depth sensor. For a typical time-of-flight sensor the data consists of a depth map and an infrared image. However, it could also include a color image, or any combination of it.

Projection. The focus of this work is on hand pose estimation from depth data. This implies that we can project the points from the 2D image space to a metric 3D space and vice versa given the camera projection model. For the remainder of this work we will stick to the following convention: We denote a coordinate in the 2D image space with associated depth d as $\mathbf{u} = (u, v, d)^T$. Given a projection ϕ the 2D point is related to a 3D world coordinate $\mathbf{x} = (x, y, z)$ by $\phi(\mathbf{u}) = \mathbf{x}$. Conversely, given a 3D point we get the corresponding image coordinate with a depth value by $\phi'(\mathbf{x}) = \mathbf{u}$. The projection model ϕ highly depends on the camera. It can be a projective model, or an orthographic one. Further, the camera intrinsics need to be known, for example from a camera calibration procedure. For more details we refer to [12].

Segmentation. Most of the existing methods need a hand detection or segmentation prior to the inference. The result of this step is either one, or possibly more detection results represented as bounding boxes, or as pixel-wise segmentation masks. Currently, we have implemented two existing approaches and we additionally propose a new mean-shift based approach.

The most simple hand segmentation is given by depth thresholding. Based on the common assumption that the hand is the nearest object to the sensor, one defines all pixels as belonging to a hand where the depth d is in the interval $[d_{\text{nearest}}, d_{\text{nearest}} + t_t]$, where t_t is a user defined threshold. One can easily see, that

the parameter t_t is very critical and that the segmentation, depending on the arm posture, will include many pixels of other body parts and the background.

The second existing approach implemented in our framework is derived from [22] and used for example in [26] for hand segmentation. We train a random forest that assigns each pixel a probability if it belongs to a hand or the background. Following the work of [22], we sample split functions in the training phase of the form

$$D\left(u + d^{-1}\Delta u, v + d^{-1}\Delta v\right) - D(u,v) \geq t_d \tag{1}$$

where (u, v) is the current pixel location with associated depth d, $(\Delta u, \Delta v)$ is a random offset vector sampled from a log-space, t_d is a sampled threshold and D is the depth map. We further apply a median filter and a morphological closing to remove outliers from the segmentation result.

The two segmentation methods described above work in the 2D image space. We propose a new simple technique that is fast and directly works with the points in 3D. Given the set of 3D points $\{\mathbf{x}_i\}_{i=1}^{N}$ and a starting point $\mathbf{x}^{(0)}$, for example the nearest point to the sensor, we apply a mean shift segmentation [5] to find the hand center by iterating

$$\mathbf{x}^{(t+1)} = \sum_i k\left(\left\|\frac{\mathbf{x}_i - \mathbf{x}^{(t)}}{h}\right\|\right)\mathbf{x}_i \left(\sum_i k\left(\left\|\frac{\mathbf{x}_i - \mathbf{x}^{(t)}}{h}\right\|\right)\right)^{-1} \tag{2}$$

until convergence or a maximum number of iterations. With an appropriate kernel profile $k(\cdot)$, the bandwidth parameter h can be related to the circumference of a common hand. A point x_i is considered as belonging to a hand, if $k(\cdot)$ is greater than ϵ.

Preprocessing. Convolutional networks, need an input of a predefined size. Therefore, we resize the segmented hand to an uniform patch size $p_w \times p_h$. However, this is also beneficial for other methods, because it decreases the variability between depth sensors that have different resolutions. We improve this robustness further, by enforcing a zero mean and a unit standard deviation of the depth values in each patch.

Convolutional network based methods have a better training convergence by decorrelating the data. For this purpose, we subtract the mean of all training samples from the data. Optionally, we could perform a Zero Component Analysis [4].

Inference. Given the segmented and preprocessed patch, we want to infer the hand posture that is defined by the set of joint locations. At the moment, we have implemented a random forest method and different approaches based on convolutional networks. All methods have in common that they determine the location in the 2D image space (u, v) with an associated depth value d, $\mathbf{u} = (u, v, d)$.

The first method is a simple regression forest similar to [9]. Given the segmented and preprocessed hand patches P_i as input, the samples are separated by split functions of the form

$$P_i(u_1, v_1) - P_i(u_2, v_2) \geq t_p \tag{3}$$

where (u_i, v_i) get sampled from the interval $[0, p_w) \times [0, p_h)$. In each leaf node we store the mean joint locations of the training samples that reach this node. At test time, the estimated joint locations are then given by the mean of the independent tree predictions.

The second type of methods that we have implemented are based on convolutional networks. One existing approach to compute predictions for (u, v) is to regress the locations of each joint in a discrete heat-map of size $h_w \times h_h$ as proposed in [26]. This has the drawback that the depth of each joint has to be determined in an additional postprocessing step. Hence, we extend this approach by adding for each joint an additional 1D discrete heat-map of size h_d to also infer the depth. A full 3D heat-map for each joint would not be feasible, because of the huge amount of parameters that have to be computed.

Instead of regressing a heat-map for the joint locations, we propose to regress the values directly. This has the benefit that we do not have to discretize the space of valid locations and further it reduces the amount of parameters in the convolutional network. In this way we can directly regress the depth values and along with the 2D joint locations.

An addition to this model relies on the assumption that the set of joint locations lies on a lower dimensional sub-space. Therefore, we can regress the values of the lower dimensional sub-space and reducing the number of parameters we have to estimate. One way to compute a mapping to a lower dimensional space is by applying a Principal Component Analysis (PCA) on the set of joint locations given by the training data [19]. This maps the joint space to a linear sub-space. However, we can implement a non-linear mapping directly in the convolutional network with an auto-encoder [13] and learn the sub-space jointly with the network itself. This is achieved by introducing a fully-connected layer of N neurons between fully-connected layers of M neurons, given $N \ll M$ [13].

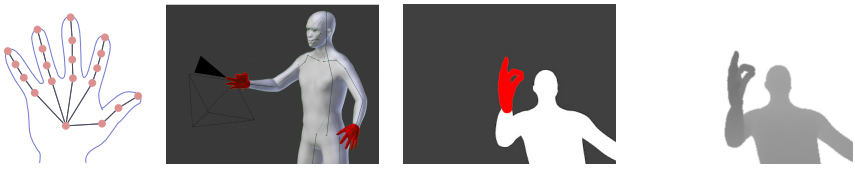
Postprocessing. The postprocessing step serves two purposes. First, for methods that do not directly infer the depth of a joint, we try to estimate it from the depth map itself by choosing the depth value at the estimated 2D joint location. The second purpose is the refinement of the estimation result. For the heat-map estimates we obtain sub-pixel accuracy by fitting a 2D Gaussian distribution [26] to the heat-map. For the future we also plan to implement different kinds of inverse kinematic models, as for example proposed in [25], or [26].

4 Dataset

This section describes the proposed dataset that consists of a synthetically generated training set and three test sequences captured by two different consumer

depth sensors. With this approach we can cover a wide range of hand shapes and also many hand postures in the training set, ensuring a high variability. Further, we minimize with this approach the cumbersome manual annotation of the data and therefore increase the accuracy of the annotations. The test sets are recorded with two different depth sensors. This has the benefit that we can evaluate the generalization-abilities of the different methods.

An important decision for the annotation is the placement of the points that define the hand skeleton. We defined those points in a way that they closely resemble the anatomy of a human hand. Therefore, we place them directly at the real joints of the hand and additionally on the finger tips. A 2D sketch of this is shown in Figure 2(a). A major advantage of this placement is that the distance between the single points stay almost constant in 3D.



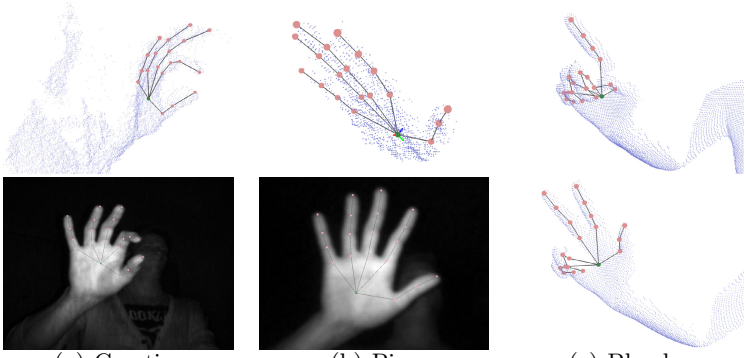
(a) Skeleton (b) Blender Model (c) Rendered Texture (d) Rendered Depth

Fig. 2. Hand Skeleton and Blender Data: (a) A sketch of the hand skeleton that we use for the annotation. It closely resembles the anatomy of a real human hand so that the distances in 3D stay almost constant. (b) A textured MakeHuman mesh in Blender with the complete skeleton. The black pyramid visualizes the camera. (c) The rendered textures can be used to train segmentation algorithms. (d) The corresponding rendered depth map of the Blender scene.

The whole synthetic training data is rendered in Blender [3]. We utilize therefore a high quality mesh of a complete human body that is created with MakeHuman [2]. For the model visualized in Blender see Figure 2(b). By using a complete human body model over a hand model alone, it is possible to generate more realistic images. We also texture the model by coloring the hand in red. In this way, we also create training data for segmentation algorithms. The output of a Blender rendering is the segmentation mask of the hand, a depth map that is perfectly pixel-aligned to the segmentation mask, and the joint annotations. For an example see Figures 2(c) and 2(d). After the rendering we add a Gaussian noise with a zero mean and a standard deviation of $\sigma = 3$ to the depth maps to simulate some of the sensor noise.

To achieve a high variability in the training set, we alter the model in two ways. First, we change the size of the hand and the fingers to simulate the diversity of different people. Second, we model a wide range of hand gestures and arm postures. This set is further increased by interpolating between random pairs of the modeled poses. In total, we rendered over 4×10^5 depth maps with associated hand segmentation masks and annotations with this setup.

For the test set we recorded three sequences with 400 frames each. We employ as depth sensors the Creative GestureCam [6] and the PMD Pico [21]. Both



(a) Creative (b) Pico (c) Blender

Fig. 3. Annotations: (a) and (b) depict sample annotations in the 3D point cloud and the 2D infrared image. In comparison, (b) shows the annotation and the point cloud of the synthetic generated data.

sensors use the time-of-flight principle to compute dense depth measurements. Additionally to the depth maps, the sensors provide infrared images. We annotated the data with the aid of those infrared images and the 3D point clouds associated with the depth maps. Samples of the annotations from the test sequences in comparison to the synthetically generated data is shown in Figure 3.

5 Evaluation

In this section, we evaluate the above described methods on our proposed dataset. We compare the different approaches to segment the hand from the background in Section 5.1. The evaluation of the pose estimation is afterwards described in Section 5.2.

5.1 Hand Segmentation

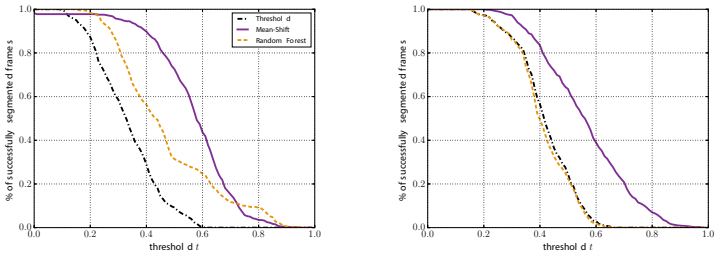
To evaluate the performance of the hand segmentation we use the intersection-over-union metric as it is commonly used in object localization [8]:

$$r = \text{area}\left(B_{\text{es}} \cap B_{\text{gt}}\right) \cdot \text{area}\left(B_{\text{es}} \cup B_{\text{gt}}\right)^{-1} . \tag{4}$$

B_{es} is the estimated hand localization and defined as the rectangle that encloses the hand segmentation. Similarly, we define B_{gt} as the rectangle with minimum area that encloses the 2D ground-truth joint locations. The value r is in the interval of $[0, 1]$ and higher values correspondent to better hand segmentations.

We evaluate the random forest based hand segmentation as explained in [26] and our proposed mean-shift based approach. The parameters for the random forest are chosen as in [26], with four trees, a maximal tree depth of 25 and 10.000 sampled split-functions per node. The random forest is trained on a subset

of 20% of the training images. For the mean-shift based approach we use a exponential kernel profile $k(x) = \exp(-x)$ and set the bandwidth parameter to $h = 13.5$. As baseline, we also include a simple depth thresholding. In this case, the segmentation contains all pixels with depth value d in the interval $[d_{\text{nearest}}, d_{\text{nearest}} + t_t]$ and $t_t = 300\text{mm}$.



(a) Creative (b) Pico

Fig. 4. Segmentation Results: The plots depict the percentage of frames where the intersection-over-union r is greater than a threshold t. The first test sequence was recorded with the Creative GestureCam (a) and the second one with a PMD Pico (b).

We evaluated those three methods on a test sequence captured by a Creative GestureCam and on one captured by a PMD Pico. In Figure 4 we plot the percentage of frames with a intersection-over-union value r greater than an increasing threshold t. We can see that the proposed mean-shift based approach delivers the best segmentations in terms of fitting the ground-truth 2D joint locations the tightest. Our method also generalizes better over different camera models. The random forest method and the thresholding include many pixels of the arm and other body parts. However, the mean-shift approach fails in some cases to find the hand, in contrast to the random forest and the thresholding method. This is especially the case, if the hand is not the nearest object to the depth sensor. Another advantage of the mean-shift segmentation is the speed as it runs a magnitude faster than the random forest based method.

5.2 Pose Estimation

In this section we evaluate the above described methods for inference on two of our three test sequences[1]. We use for all experiments the proposed mean-shift approach to segment the hand from the background, with the previously stated parameters. However, instead of initializing the algorithm with the pixel with the smallest depth value, we set $\mathbf{x}^{(0)}$ to the ground truth location of the palm to ensure a successful hand segmentation. After scaling the segmented hand to a patch size of 96×96, we normalize the depth within each patch to zero mean and a unit standard deviation. Further, we subtract the mean over the training set from each patch.

[1] Results on additional test sequences can be found on our project page.

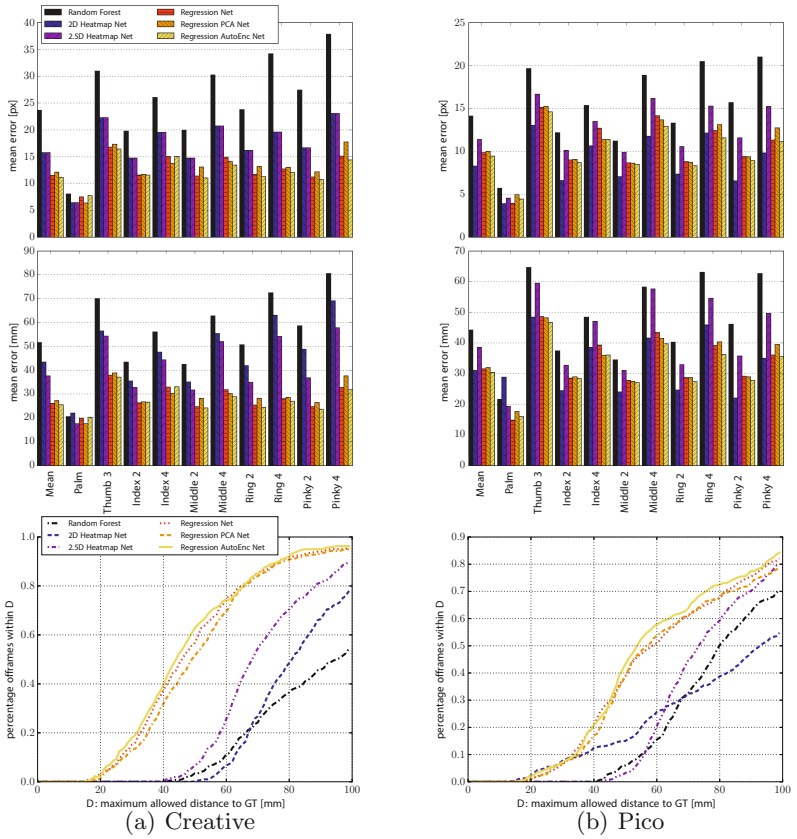


Fig. 5. Quantitative Evaluation: The graphs show the inference result in 2D and 3D for three test sequences. (a) was captured with a Creative GestureCam, whereas (b) was captured with a PMD Pico. In the first row, we plot the mean estimation results in 2D. The second row shows the same in 3D. Finally, we show in the last row the percentage of frames where all joints are within a distance of D.

The first method is the regression forest, where we trained 8 trees with a maximal depth of 25. For each node we sample 10^4 split functions and optimize them with a random sub-set of $5 \cdot 10^3$ training patches. In the leaf node we store the mean of the regression vectors that reaches this node and at inference, we compute the mean over all tree results to get the final estimate. For the convolutional network based methods we utilize the same multi-resolution architecture as in [26]. Only the fully connected layers differ for these networks. As activation functions we use a ReLU in the hidden layers, and a linear one in the output layer. To improve generalization, we apply dropout with a probability of 0.5 in the fully-connected layers. The first network is similar to the one in [26], except that we have 20 output heat-maps of size 18×18 instead of 14. In the second network we add for each joint a 1D heat-map of size 25 to infer also a depth

value from the network. We replace the heat-maps with standard regression in the third network architecture and have two fully-connected layers with 4096 neurons each, so we get an output structure of $4096 \times 4096 \times 60$ neurons. Finally, we evaluate the network methods that are based on the sub-space assumption described above. In the first variant, we approximate a PCA with an auto-encode of the form $4096 \times 4096 \times 20 \times 60$, where we utilize also linear units in the layer before the last. Finally, we use a non-linear auto-encoder in the output layers of the form $4096 \times 4096 \times 20 \times 4096 \times 60$. This projects the pose space to a non-linear sub-space as described in [13].

The results are depicted in Figure 5. We show in the first row the 2D mean error in pixels. Similarly, the second row compares the mean error of the 3D estimates in *mm*. In the last row, we visualize the percentage of frames where all 3D joint estimates are within an increasing distance D to the ground-truth.

We can observe that the convolutional network methods outperform the regression forest. Our proposed extension to the heat-map to infer also the depth is beneficial. In 2D the accuracy stays identical, however in 3D we get an improvement of the results. Interesting is the fact that the regression based networks perform even better. While the PCA based network achieves nearly no gain of performance, we can see that the non-linear auto-encoder further boosts the results, validating the sub-space assumption.

6 Conclusion

We presented in this a paper a framework for articulated hand pose estimation that includes state-of-the art segmentation and inference methods. Further, we proposed a new method for segmenting the hand from the background and improving existing inference approaches. To overcome common problems with current benchmarks for hand pose estimation, we created a new dataset that consists of a synthetically generated training set that covers a large space of possible postures. Additionally, we accurately annotated three test sequences obtained by two consumer depth sensors. In the evaluations we showed the benefits of our proposed methods. To foster further research in the community we make all sources and the dataset publicly available. In the future, we plan to add test sequences from more depth sensors, as for example the Microsoft Kinect One^{TM} and implementing further segmentation and inference methods within our framework.

Acknowledgments. This work was supported by the Austrian Research Promotion Agency (FFG) under the *FIT-IT Bridge* program, project #838513 (TOFUSION).

References

1. Ballan, L., Taneja, A., Gall, J., Van Gool, L., Pollefeys, M.: Motion capture of hands in action using discriminative salient points. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part VI. LNCS, vol. 7577, pp. 640–653. Springer, Heidelberg (2012)

2. Bastioni, M., Re, S., Misra, S.: Ideas and methods for modeling 3D human figures: the principal algorithms used by makehuman and their implementation in a new approach to parametric modeling. In: COMPUTE (2008)
3. Blender Online Community: Blender - A 3D Modelling and Rendering Package. Blender Foundation (2015). http://www.blender.org
4. Coates, A., Ng, A.Y.: Selecting receptive fields in deep networks. In: NIPS (2011)
5. Comaniciu, D., Meer, P.: Mean Shift: A Robust Approach Toward Feature Space Analysis. TPAMI **24**(5), 603–619 (2002)
6. Creative Technology Inc., Singapore.: Creative Interactive Gesture Camera
7. Erol, A., Bebis, G., Nicolescu, M., Boyle, R.D., Twombly, X.: Vision-Based Hand Pose Estimation: A Review. CVIU **108**(1–2), 52–73 (2007)
8. Everingham, M., Eslami, S., Van Gool, L., Williams, C., Winn, J., Zisserman, A.: The Pascal Visual Object Classes Challenge: A Retrospective. IJCV, 1–39 (2014)
9. Gall, J., Yao, A., Razavi, N., Gool, L.J.V., Lempitsky, V.: Hough Forests for Object Detection, Tracking, and Action Recognition. TPAMI **33**(11), 2188–2202 (2011)
10. Girshick, R.B., Shotton, J., Kohli, P., Criminisi, A., Fitzgibbon, A.W.: Efficient regression of general-activity human poses from depth images. In: ICCV (2011)
11. de Gorce, M.L., Fleet, D.J., Paragios, N.: Model-Based 3D Hand Pose Estimation from Monocular Video. TPAMI **33**(9), 1793–1805 (2011)
12. Hartley, R., Zisserman, A.: Multiple View Geometry in Computer Vision. Cambridge University Press. 2 edn. (2003)
13. Hinton, G.E., Salakhutdinov, R.R.: Reducing the Dimensionality of Data with Neural Networks. Science **313**(5786), 504–507 (2006)
14. Keskin, C., Kıraç, F., Kara, Y.E., Akarun, L.: Hand pose estimation and hand shape classification using multi-layered randomized decision forests. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part VI. LNCS, vol. 7577, pp. 852–863. Springer, Heidelberg (2012)
15. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: NIPS (2012)
16. Melax, S., Keselman, L., Orsten, S.: Dynamics based 3d skeletal hand tracking. In: I3D (2013)
17. Mitra, S., Acharya, T.: Gesture Recognition: A Survey. SMC **37**(3), 311–324 (2007)
18. Neverova, N., Wolf, C., Taylor, G.W., Nebout, F.: Hand segmentation with structured convolutional learning. In: Cremers, D., Reid, I., Saito, H., Yang, M.-H. (eds.) ACCV 2014. LNCS, vol. 9005, pp. 687–702. Springer, Heidelberg (2015)
19. Oberweger, M., Wohlhart, P., Lepetit, V.: Hands deep in deep learning for hand pose estimation. In: CVWW (2015)
20. Oikonomidis, I., Kyriazis, N., Argyros, A.: Full DoF tracking of a hand interacting with an object by modeling occlusions and physical constraints. In: ICCV (2011)
21. PMD Technologies. Germany.: Camboard Pico
22. Shotton, J., Sharp, T., Kipman, A., Fitzgibbon, A., Finocchio, M., Blake, A., Cook, M., Moore, R.: Real-time human pose recognition in parts from single depth images. In: CVPR (2011)
23. Sun, M., Kohli, P., Shotton, J.: Conditional regression forests for human pose estimation. In: CVPR (2012)
24. Tang, D., Chang, H.J., Tejani, A., Kim, T.K.: Latent regression forest: structured estimation of 3D articulated hand posture. In: CVPR (2014)
25. Tang, D., Yu, T.H., Kim, T.K.: Real-time articulated hand pose estimation using semi-supervised transductive regression forests. In: ICCV (2013)
26. Tompson, J., Stein, M., Lecun, Y., Perlin, K.: Real-Time Continuous Pose Recovery of Human Hands Using Convolutional Networks. TOG **33**(5), 169 (2014)