3D Hand-Object Pose Estimation from Depth with Convolutional Neural Networks

DOI:
10.1109/FG.2017.58

Document Version
Accepted author manuscript

3D Hand-Object Pose Estimation from Depth with Convolutional Neural Networks

Duncan Goudie and Aphrodite Galata
Advanced Interfaces Group, School of Computer Science, University of Manchester,
Manchester, United Kingdom

***Abstract*— Estimating the 3D pose of a hand interacting with an object is a challenging task, harder than hand-only pose estimation as the object can cause heavy occlusion on the hand. We present a two stage discriminative approach using convolutional neural networks (CNN). The first stage classifies and segments the object pixels from a depth image containing the hand and object. This processed image is used to aid the second stage in estimating hand-object pose as it contains information regarding the object location and object occlusion. To the best of our knowledge, this is the first attempt at discriminative one shot hand-object pose estimation. We show that this approach outperforms the current state-of-the-art and that the inclusion of a segmentation stage to learned discriminative single stage systems improves their performance.**

I. INTRODUCTION

The human hand is a remarkably complex body part to analyse, yet almost the entire global human population finds them vital for communication purposes and for completing everyday tasks. We use them to manipulate objects and to work with tools. This work investigates the problem of estimating the pose of the joints of the hand and the object it is interacting with, ie hand-object pose estimation. Hand-object pose estimation has applications ranging from robotics (teaching a robot hand how to use a tool) to human computer interaction; with a recent surge in popularity for virtual reality devices, the capability to use hands and handheld props as controllers would greatly enhance the user experience.

The related field of hand-only pose estimation has in recent years made use of depth sensors, which provide 2.5D information, to eliminate many of the field's problems including variance in lighting conditions and texture and in background separation. However, despite the advantages that come with depth information, the hand is prone to self occlusion from fingers. Introducing an object for the hand to interact with makes the occlusion problem worse; entire fingers can potentially become occluded from a large object. Given such occlusion, a viable hand-object pose estimation method would need to be capable of estimating 3D joint positions *behind the object*. Previous solutions to the hand-object pose estimation problem that have this capability include [19], [20], [1], [24], which produce good results, but are reliant on the previous frame's pose estimate being close to the current frame's true pose; catastrophic error can occur if the difference in both frame's poses are too large. Many of these solutions use a multi-camera setup rather than a monocular sensor to avoid the problem of objects completely obstructing the view of a joint.

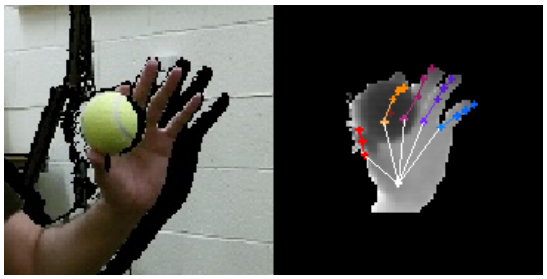


Fig. 1: We present a new two stage discriminative system for estimating the joint and object locations of the hand interacting with an object from the depth image. In this image, the index finger is almost completely occluded by the ball.

In this paper, we present a two stage system which can perform one shot hand-object pose estimation without the need for prior pose information from previous frames. Our system requires only the depth image from a single depth camera and is capable of inferring 3D finger joint positions that have been occluded by the object (see figure 1 for an example result). The first stage segments the object from the hand to provide prior information regarding object positioning and object occlusion to the second stage, which performs the 3D joint position prediction. Our experiments show that our proposed system outperforms the state-of-the-art and also that the inclusion of a segmentation stage can improve the performance of suitably adapted hand-only single stage systems such as DeepPrior [16] to the task of hand-object pose estimation. Our contributions can be summarised as follows:

- To the best of our knowledge, this is the first attempt at discriminative one shot hand-object pose estimation.
- We use an accurate learned hand-object segmentor to aid and improve the performance of the hand-object joint localiser.
- We present two new fully labelled datasets for hand-object pixel segmentation and for hand-object pose estimation, which we will make publicly available. They contain a hand manipulating a spheroidal object.

This work was supported by the Engineering and Physical Sciences Research Council EP/I028099/1.

978-1-5090-4023-0/17/$31.00

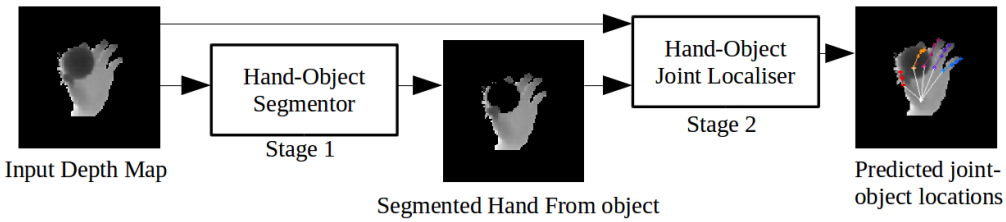


Fig. 2: System overview showing our 2 stage system. The first stage segments and sets the object pixels to zero. A 2 channel image containing both the input depth map and the processed depth map from stage one is sent into the second stage to perform the 3D joint position prediction.

II. RELATED WORK

The literature on the closely related topic of hand-only pose estimation from depth images can be broadly categorised into two categories: generative and discriminative. Generative approaches make reference to a hand model whereas discriminative approaches perform pose estimation from just the given image alone. Supancic et al. [31] produced a comprehensive literature review on hand-only pose estimation from depth. However, the related work on hand-object pose estimation can so far only fall into the generative category.

Generative hand-only pose estimation schemes often make use of an articulated hand model, usually either consisting of simple geometric shapes [18], [23], quadrics [29], a mesh [25], [2], [1] or with gaussians [28]. Oikonomidis et al. [18] made a significant advance with the application of Particle Swarm Optimisation [8] to minimising a cost function with an inherent hand pose hypothesis. This approach is easily extended to also estimate the pose of hands and objects [19] and extra hands [20]. We use a variation on this approach to find the groundtruth joint locations for our new hand-object pose estimation dataset (see section IV-A for more details). Other cost functions and optimisation schemes have been proposed within the generative approach to hand-only pose estimation [23], [28], [14], [25], [10]. Generative approaches tend to offer wide flexibility in the range of poses they can estimate, but due to the high dimensional nature of this problem, the search space has to be narrowed to make this approach feasible. A common approach to restricting the search space is to initialise the system with the previous frame's pose estimate, creating a dependency on the result of the previous frame. Systems which produce renderings of the hand model as part of their cost function require large computational resources to satisfy realtime pose estimation.

Convolutional neural networks (CNN) are increasingly being used in discriminative hand-only pose estimation [16], [15], [5], [27], [34], [3]. Supancic et al. [31] ran extensive tests on a large number of discriminative depth based hand-only pose estimation methods and CNNs were found to outperform competing methods such as random decision forests (RDF). Tompson et al. [34] used CNNs to create heat map predictions for the joint locations. This method was reliant on using the surface depth values for estimating the joint positions in 3D and as such would not be suitable for inference behind the object. Oberweger et al. [16] proposed a set of CNN architectures for use in hand-only pose estimation, including DeepPrior. DeepPrior makes use of a bottleneck within the densely connected layers to force the CNN to learn a lower dimensional representation of the hand pose. CNNs have also been applied to the closely related problem of human body pose estimation [36], [35], [33]. RDFs have been used as an alternative to body pose estimation [26], [4] and for hand-only pose estimation [37], [9].

The topic of this paper is on hand-object pose estimation and a large majority of the work in this field falls within the generative category of pose esimation, where a multi-camera setup was used to help with the object occlusion problem [19], [1] or a 2.5D monocular depth sensor [6], [12], [21], [22]. These methods are quite effective, but suffer from the disadvantages that come from a generative approach. Romero et al. [24] describe a discriminative approach to hand-object pose estimation for monocular RGB images, using (approximate) nearest neighbours and hashing to reduce the number of lookups within their pose database. However, their approach is not one-shot as they use the previous frame's estimate to improve pose lookup performance.

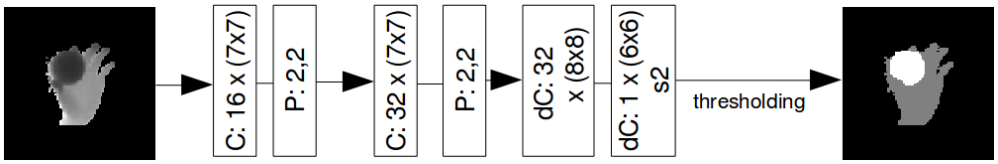
Hybrid approaches, the combination of discriminative and generative systems, attempt to utilise the advantages that come from both systems. A discriminative system is often used to initialise the search space for the generative system [25], [10]. A learned method has been used to improve the performance of a generative scheme by trying to localise the position of salient points (finger nails) [1]. A hybrid approach, using CNNs, has been suggested by Oberweger et al. [17]. They use CNNs to synthesise hand models where a reference is then made to, along with an original discriminative pose estimate, by an updater CNN. Our one-shot discriminative approach could potentially be applied to a future hybrid system.

III. HAND-OBJECT POSE ESTIMATION

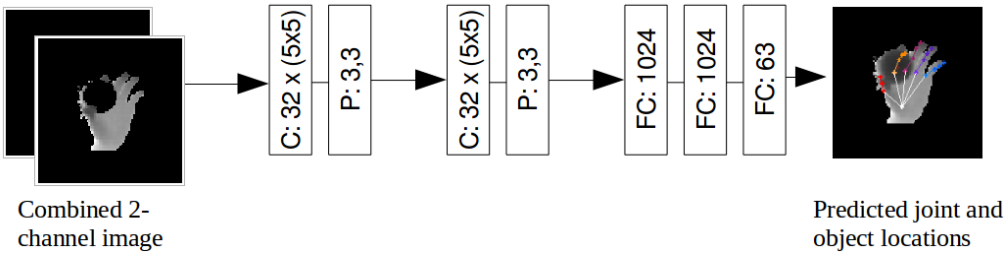
In this section we describe our contribution to hand-object pose estimation.

A. Problem Formulation

Our goal is to estimate the hand-object joint locations $\boldsymbol{J} = \{\boldsymbol{j}_i\}_{i=1}^{J}$ where $\boldsymbol{j}_i$ is a vector, $[x_i, y_i, z_i]$, representing the 3D position of a single joint or object. $J = 21$ in our



(a) Stage 1: Hand-Object Segmentor



(b) Stage 2: Hand-Object Joint Localiser

Fig. 3: Architectures for both stages in our system.

formulation, where we allocate 3D position vectors for: 4 joints in each finger (including tip), upper most 3 joints (including tip) of the thumb, wrist center and object center; from now on, we will refer to each of these as a joint. Systems such as [16] can localise hand joints from a single depth image, I, of a hand isolated from the background. A CNN regressor can be trained to learn a regression function $\hat{\boldsymbol{J}} = f(I)$ to predict the joint locations.

We propose a two stage system (see figure 2). Stage one is a segmentor which classifies pixels as belonging either to a hand or object class. The object classified pixels are set to zero, leaving only the depth values for the hand classified pixels from I to produce I^*. Stage two performs hand-object joint prediction given a two channel image from I and I^*,

$$\hat{\boldsymbol{J}} = f(I, I^*) \tag{1}$$

We describe these stages in further detail in the following sections.

B. Convolutional Neural Networks

CNNs have been shown to produce state-of-the-art results in many computer vision problems such as image classification [11]. The typical CNN architecture is composed of multiple stacked convolutional layers, with max-pooling in between them to both reduce the computational load and increase robustness. The last few layers are often (dense) fully connected neurons which perform decision making given the activations of the previous convolutional layers. Training is done using batches of ground truth labelled images with stochastic gradient descent attempting to minimise a designated loss term.

CNNs have been shown by [31] to produce the current best performing discriminative systems for hand-only pose estimation, outperforming alternative methods such as RDFs and nearest neighbours. It is for this reason that we use CNNs to learn our regressor functions for stage one and two. For all stages in our system we use ReLU activation functions and we use euclidean distance as our loss term in training. Stages one and two take as input a depth image, I or $\{I, I^*\}$, respectively, of size 96×96. The background pixels have been set to 0 and the depth values normalised to be within 0 and 1, where the values closer to 0 denote the pixel being closer to the camera. In training the CNNs we augment our dataset by rotating it between $-90°$ and $+90°$ and with translations of up to 15 pixels in either direction. The artificial rotations allow our system to cope with in-plane rotation of the hand and the translations improve robustness as it forces the system to learn to cope with differently centred images.

C. Stage One: Hand-Object Segmentation

The objective of this stage is to segment the object pixels from a given depth map, I, of the hand-object. For the related area of segmenting the hand from the background in hand-only pose estimation, RDFs have been shown to be reliable [34]. However, we found that CNNs outperform RDFs when applied to the problem of hand-object segmentation (see section IV-B.3 for more details), especially when configured as a fully convolutional network (FCN) [13]. FCNs replace the fully connected layers with deconvolutional layers, essentially making the entire network a learned nonlinear filter.

Our network architecture for stage one is described in

figure 3a. We use convolution filters of size 7×7. We found that keeping the filtered images to a high resolution led to the best segmentation performance, so for that reason we kept max-pooling to a minimum by setting them to have a size 2 and a stride of 2. Removing the max-pooling layer did not improve results. After the convolutional layers, the deconvolutional layers had to be designed in a way that would take the multiple filtered images they receive and to output a single channel image; hence why we used a single deconvolutional filter in the final layer. The deconvolutional filter sizes had to be chosen carefully so that the output image of this network matched the size of the input image, I. We achieved this using a first set of deconvolutional filters of size 8×8 and a second deconvolutional filter of size 6×6 with a stride of 2. After systematic testing of different values, the values we use for this system were the values we found to produce the best results.

We use our hand-object segmentation dataset for training this stage. Each image is of size 96×96 and are accompanied with per-pixel class labels with values of: 0.0, 0.5 and 1.0 to represent the background, hand and object class pixel respectively. The output image from this network contain pixels of continuous values between 0 and 1 and so to perform a classification decision on each pixel we use thresholding. We found that using a threshold of 0.05 and 0.75 to separate the background and hand classes, and the hand and object classes, respectively to work best, to produce a label map. Finally, to create I^*, we correlate the predicted object classified pixels from the label map with I and set the corresponding depth values to zero, leaving just the hand depth pixels.

D. Stage Two: Hand-Object Localisation

Stage two performs the regression function in equation 1 to predict the 3D hand-object joint locations. We tried many CNN architectures, configurations and parameters for this stage and found that the simple 2 convolutional layer CNN which took as input a combined 2-channel image, $\{I, I^*\}$, to perform best. The architecture parameters used in this system are described in figure 3b.

In training this stage, the ground truth labels, which consist of $\boldsymbol{J}$ with $J = 21$ for each image, have been flattened to form a single vector of length 63. The x_i and y_i positions have been projected to u_i and v_i, respectively to present position in terms of pixel location within the image with values normalised to be between 0 and 1. z_i has been converted to $\tilde{z}_i$ to represent the z component as a ratio between the smallest and largest depth values present within the image. The images within the training set of the hand-object pose estimation dataset were processed by stage one, already trained on the hand-object segmentation dataset, to create the processed training images, I^*, for this stage. In testing, the CNN makes a prediction in the form of $[u_i, v_i, \tilde{z}_i]$. We apply these transforms in reverse to get $\hat{\boldsymbol{J}}$.

IV. EXPERIMENTS AND RESULTS

A. Dataset

In this paper we present two new datasets[1], one for each of the two stages, which were created using the Microsoft Xbox One Kinect depth camera and we make these two datasets publicly available. These cameras produce a depth map and an RGB registered image pair of size 512×424. The hand-object segmentation dataset consists of 5635 image pairs for training and 1042 for testing. The hand-object pose estimation dataset consists of 3986 image pairs for training and 745 for testing. For both datasets, the background pixels have been set to zero. All training and testing datasets consists of a hand interacting with a tennis ball, with the exception of the hand-object pose estimation test dataset which has a mixture of objects: tennis ball, orange and lemon.

In creating the joint label dataset, we use a method similar to the one employed by [34], which uses a generative hand-only pose estimation method [18]: using a hand model and Particle Swarm Optimisation (PSO) [8] to estimate the ground truth labels. We use two cameras, one facing the front of the hand and the other facing the back to ensure all hand joints and object positions were always present in at least one camera. To improve accuracy and to encourage faster convergence, we manually label the joints to get surface positions and use this generative method to refine the joint positions and to ensure they fit within a hand model's constraints. We use the fitness function from [18] and include an extra term, the euclidean distance between manual labelling and hand model labelling. For full details on this generative method we refer the reader to [18] and [19].

B. Empirical Evaluation

1) Evaluation Metrics: For evaluating the overall system, we use two well established evaluation metrics [31], [16], [32] and [30]: the per-joint euclidean error averaged over all frames and the proportion of frames that has a mean and max euclidean error of less than a threshold, ϵ_{mean} and ϵ_{max} respectively. Evaluating systems agaisnt the max euclidean error metric is regarded within the community as a very challenging metric. Under the latter metric, the better performing system would have a greater area under

[1]Available at http://www.cs.man.ac.uk/%7egoudied/

the curve plot. For evaluating the segmentation system, we work out the proportion of frames that has the total number of misclassified pixels below a threshold, ϵ_{pixels}.

2) Experimental Details: We used Caffe [7], an efficient CUDA parallelised solver for CNNs using back-propagation. We used RMSprop, a stochastic gradient descent method with an adaptive learning rate, to train our networks. We set the RMS decay to 0.02 and the weight decay to 0.0005. We use a batch size of 128. We set the learning rate to 0.001 and 0.005 for stages one and two respectively and decreased it by factor 0.7071 and 0.316 respectively twice over a total of 70 and 50 epochs respectively.

3) Hand-Object Segmentor: As the goal of this stage is to zero out the object classified pixels from the image, we only performed pixel classification as an object class vs non-object class problem. We compare this solution with two alternatives: replacing the deconvolutional layers with dense fully-connected (FC) layers and with RDFs. We found the best FC layer arrangement was 128-64-9216 neurons, where each neuron in the last FC layer corresponds to a pixel in the original 96 $\times$ 96 depth map and used (3,3) pooling layers instead of (2,2). We used a similar approach to [26] with setting up our RDFs; we trained 10 trees to a maximum depth of 20 and evaluated 5000 candidate tests at each node.

Figure 4 shows the quantitative evaluation of our hand-object pixel segmentor. It is clear that our proposed system with deconvolutional layers massively outperforms the use of FC layers and the RDFs. The number of learnable parameters within the FC model is of several magnitudes greater than that of the deconvolutional model and as such makes the model harder to generalise. Both systems qualitatively performed quite well (see figure 6) when the tennis ball had minimal occlusion. When there was heavy occlusion with several fingers covering portions of the object, the proposed system was able to classify the object pixels with good semantic accuracy (from a human perspective) whereas the dense FC system struggled.

We evaluated our proposed system, using deconvolutional layers, with different raw pixel threshold values for hand-object pixel classification (ie a raw pixel value above T would classify that pixel as belonging to the object class). A T value of 0.75 was found to create the best overall performance, performing better than $T = 0.80$ for all test images and with an average classification error of 64.0 pixels against 73.5 pixels taken over all frames. Comparing against the system with FC layers, this error comes to around 123.7 pixels or 209.6 pixels for RDFs.

4) Hand-Object Localiser: We evaluate our proposed system and compare against DeepPrior [16], a state-of-the-art hand-only pose estimation method that can be adapted

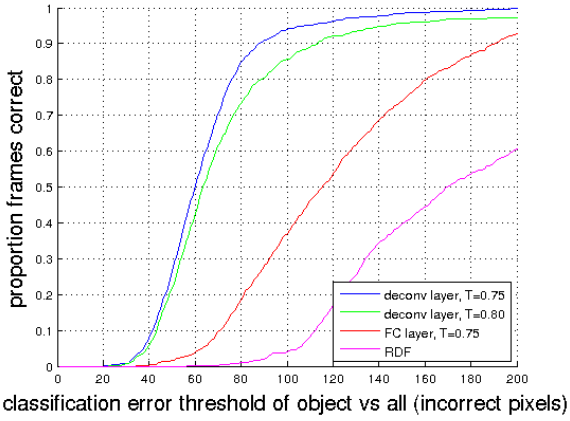


Fig. 4: Evaluation of stage one, the hand-object segmentor.

to our problem. We implement DeepPrior ourselves on our hand-object pose estimation dataset using the same hyper parameters and details discussed in section IV-B.2. We use the same CNN architectural parameters for DeepPrior as described in [16], including the use of 30 FC neurons in the FC "Prior" layer.

To show that the inclusion of a hand-object segmentation stage improves results, we also add and remove this stage from these two systems as appropriate. All charts in figure 5 show that the inclusion of a segmentation stage improves performance, with both mean-error (top left) and max-error (top right) plots for the with segmentation system (solid line) containing a greater area under the curve as opposed to those without (dashed line). The bar chart (bottom left) shows the average max-error for selected joints (wrist, finger and thumb tips, ball and mean over all of them); we only show the finger tips as they had the largest error out of any joint within a finger joint chain. This chart supports our hypothesis that the inclusion of a segmentation stage improves performance for all joints. The ring and little fingertip joints appear to perform less well than the index and middle finger; the testset includes poses where the ball completely occludes the ring and little fingers, making these appendaces difficult to estimate and it might also be a characteristic on what the training dataset is capable of.

We performed segmentation noise tests on the localiser stage by adding synthetic noise to the tennis ball images within the hand-object pose estimation test dataset. We excluded orange and lemon images as to test noise we required comparisons with the exact objects seen in the training dataset. To add noise, we first subtracted random circles around the outline of the groundtruth segmentation and then

performed erosion on the remaining segmentation. Although hard to approximate true noise from a CNN segmentor, visually this produced acceptable noisy segmentations. We then treated these noisy images as if they were I^*. Throughout these tests we trained the localiser on groundtruth segmented I^*. In the bottom right chart of figure 5, we found that our system was sensitive to around 10-15% synthetic noise, ie we found that with any more noise it would be better to use a single stage localiser instead. This is a reasonable allowance for our system as we have shown that with stage one processed I^* (depicted by bold red line in bot. right Fig. 5), the localiser performs almost as well as if I^* were groundtruth test images (bold blue line).

The average max joint error for our proposed system was found to be 23.15mm. This rises to 25.01mm for if the segmentor stage was removed, showing a 7.4% improvement with segmentor included. Similarly, we find with using this metric, there is a 4.0% improvement when the segmentor is included for the DeepPrior system. Our proposed full system with both stages performs marginally better than DeepPrior with segmentor. Overall, our complete system performs better than DeepPrior, with a 7.2% improvement. We performed t-tests on these systems and found that the use of a segmentor did provide a statistically significant improvement. It took on average 0.95ms to feedforward 1 image through our stage two CNN and 1.14ms for our stage one CNN (Intel i7 and Nvidia GeForce 780Ti).

Qualitative results can be seen in figure 7 where we compare our system directly with DeepPrior. Both systems appear to cope well against the occlusion caused by the ball. For the finger joints that are visible, we can see that our system qualitatively appears to perform better than DeepPrior with a smaller max-joint error.

V. Conclusions

In this paper we present a two stage system for hand-object pose estimation, using the first stage to segment the object pixels to provide prior information regarding the object's positioning to the second stage 3D joint localiser. We show that the use of a hand-object segmentation stage improves performance over learned single stage systems and that our system outperforms the state-of-the-art. To the best of our knowledge, this is the first attempt at discriminative one shot hand-object pose estimation. As future work, we plan to use our method within a hybrid hand-object pose estmation system.

References

[1] L. Ballan, A. Taneja, J. Gall, L. V. Gool, and M. Pollefeys. Motion capture of hands in action using discrimative salient points. In *European Conference on Computer Vision*, 2012.

[2] M. Bray, E. Koller-Meier, and L. V. Gool. Smart particle filtering for 3d hand tracking. In *IEEE International Conference on Automatic Face and Gesture Recognition*, 2004.

[3] C. Choi, A. Sinha, J. Hee Choi, S. Jang, and K. Ramani. A collaborative filtering approach to real-time hand pose estimation. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[4] M. Dantone, J. Gall, C. Leistner, and L. Van Gool. Human pose estimation using body parts dependent joint regressors. In *Computer Vision and Pattern Recognition (CVPR), 2013 IEEE Conference on*, pages 3041–3048. IEEE, 2013.

[5] L. Ge, H. Liang, J. Yuan, and D. Thalmann. Robust 3d hand pose estimation in single depth images: From single-view cnn to multi-view cnns. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[6] H. Hamer, K. Schindler, E. Koller-Meier, and L. Van Gool. Tracking a hand manipulating an object. In *2009 IEEE 12th International Conference on Computer Vision*, pages 1475–1482. IEEE, 2009.

[7] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

[8] J. Kennedy and R. Eberhart. Particle swarm optimisation. In *International Conference on Neural Networks*, 1995.

[9] C. Keskin, F. Kıraç, Y. E. Kara, and L. Akarun. Real time hand pose estimation using depth sensors. In *Consumer Depth Cameras for Computer Vision*, pages 119–137. Springer, 2013.

[10] P. Krejov, A. Gilbert, and R. Bowden. Combining discriminative and model based approaches for hand pose estimation. In *Automatic Face and Gesture Recognition (FG), 2015 11th IEEE International Conference and Workshops on*, volume 1, pages 1–7. IEEE, 2015.

[11] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[12] N. Kyriazis and A. A. Argyros. Scalable 3d tracking of multiple interacting objects. In *Computer Vision and Pattern Recognition*, 2014.

[13] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

[14] S. Melax, L. Keselman, and S. Orsten. Dynamics based 3d skeletal hand tracking. In *Proceedings of Graphics Interface 2013*, pages 63–70. Canadian Information Processing Society, 2013.

[15] N. Neverova, C. Wolf, G. W. Taylor, and F. Nebout. Hand segmentation with structured convolutional learning. In *Computer Vision–ACCV 2014*, pages 687–702. Springer, 2014.

[16] M. Oberweger, P. Wohlhart, and V. Lepetit. Hands deep in deep learning for hand pose estimation. In *Computer Vision Winter Workshop*, 2015.

[17] M. Oberweger, P. Wohlhart, and V. Lepetit. Training a feedback loop for hand pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3316–3324, 2015.

[18] I. Oikonomidis, N. Kyriazis, and A. A. Argyros. Efficient model-based 3d tracking of hand articulations using kinect. In *British Machine Vision Conference*, 2011.

[19] I. Oikonomidis, N. Kyriazis, and A. A. Argyros. Full dof tracking of a hand interacting with an object by modeling occlusions and physical constraints. In *International Conference on Computer Vision*, 2011.

[20] I. Oikonomidis, N. Kyriazis, and A. A. Argyros. Tracking the articulated motion of two strongly interacting hands. In *Computer Vision and Pattern Recognition*, 2012.

[21] P. Panteleris, N. Kyriazis, and A. A. Argyros. 3d tracking of human hands in interaction with unknown objects. In Mark W. Jones Xianghua Xie and Gary K. L. Tam, editors, *Proceedings of the British Machine Vision Conference (BMVC)*, pages 123.1–123.12. BMVA Press, September 2015.

[22] T. Pham, A. Kheddar, A. Qammaz, and A. A. Argyros. Towards force sensing from vision: Observing hand-object interactions to infer

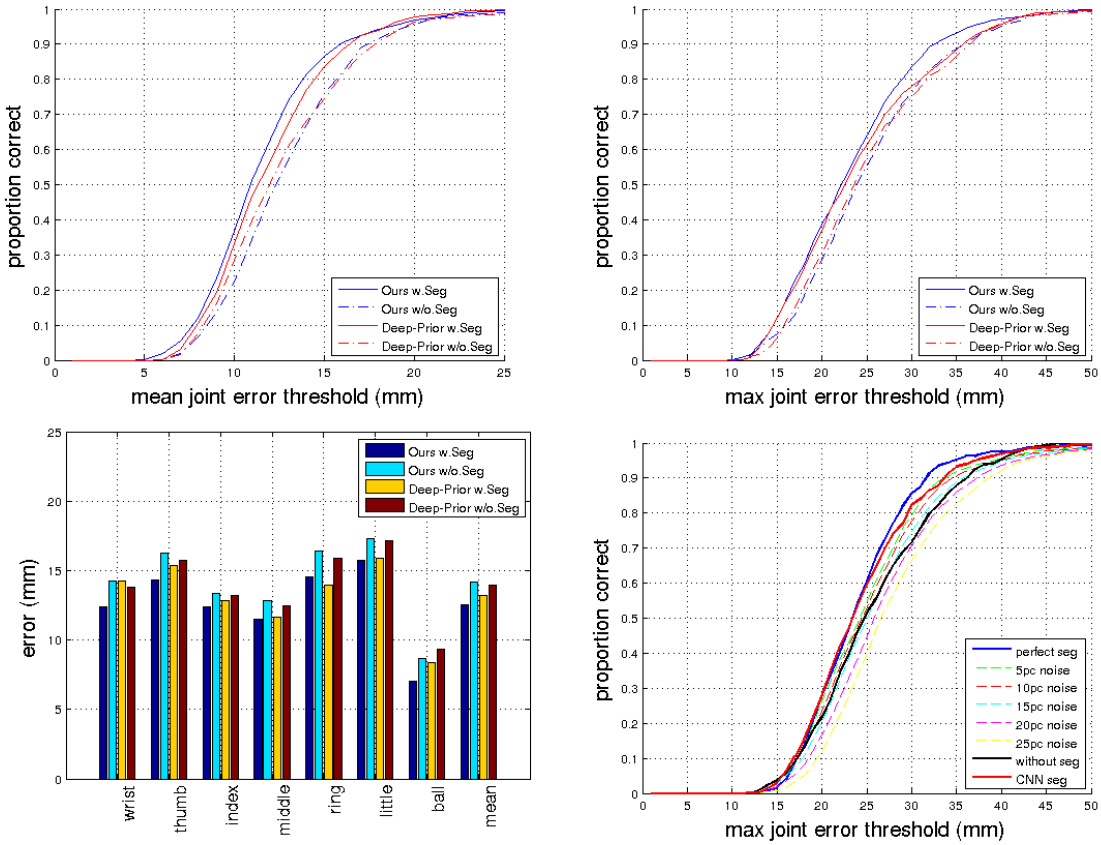


Fig. 5: Hand-object pose estimation results. Top left: mean joint error. Top right: max joint error. Bottom left: Average error for each appendage. Bottom right: noise test results. (Best viewed in colour)

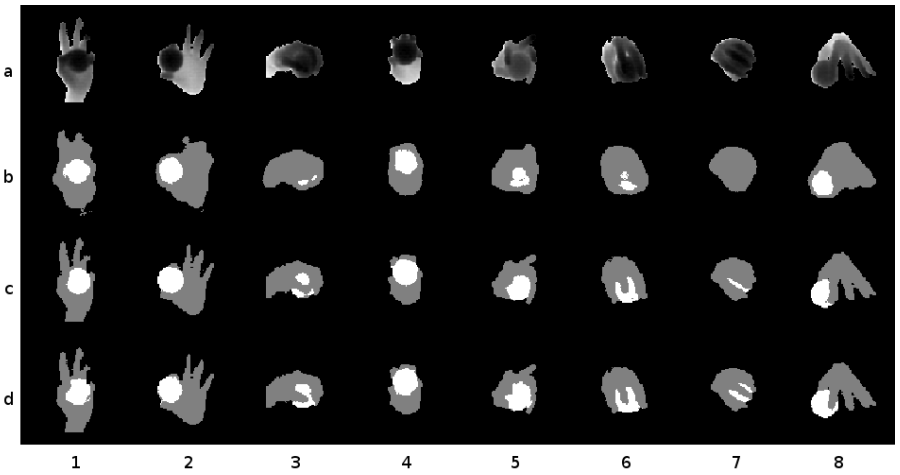


Fig. 6: Qualitative results of our segmentor: (a) the original input depthmap, (b) fully-connected network results, (c) fully convolutional network results and (d) ground truth segmentation images. The white regions represent the segmented object.

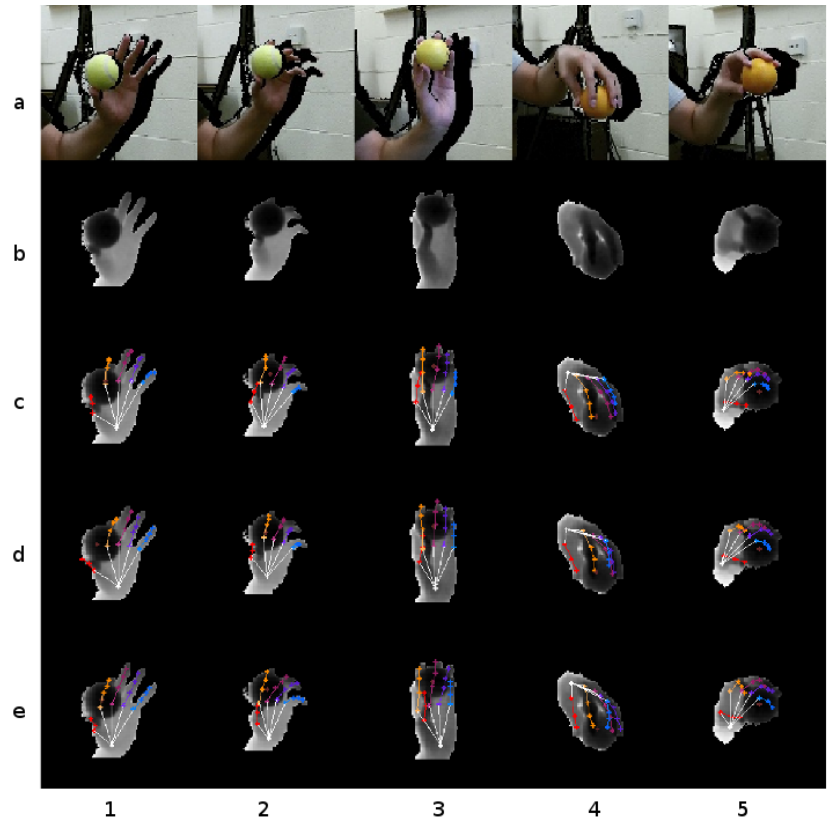


Fig. 7: Example qualitative results showing (a) the RGB image, (b) the depth image, (c) DeepPrior, (d) Ours and (e) ground truth. (Best viewed in colour)

manipulation forces. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2810–2819, 2015.

[23] C. Qian, X. Sun, Y. Wei, X. Tang, and J. Sun. Realtime and robust hand tracking from depth. In *Computer Vision and Pattern Recognition*, 2014.

[24] J. Romero, H. Kjellström, C. H. Ek, and D. Kragic. Non-parametric hand pose estimation with object context. *Image and Vision Computing*, 31(8):555–564, 2013.

[25] T. Sharp, C. Keskin, D. Robertson, J. Taylor, J. Shotton, D. K. C. R. I. Leichter, ..., and S. Izadi. Accurate, robust, and flexible real-time hand tracking. In *Proc. CHI*, volume 8, 2015.

[26] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake. Real-time human pose recognition in parts from single depth images. In *Computer Vision and Pattern Recognition*, 2011.

[27] A. Sinha, C. Choi, and K. Ramani. Deephand: Robust hand pose estimation by completing a matrix imputed with deep features. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[28] S. Sridhar, A. Oulasvirta, and C. Theobalt. Interactive markerless articulated hand motion tracking using rgb and depth data. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 2456–2463. IEEE, 2013.

[29] B. Stenger, P. R. S. Mendoca, and R. Cipolla. Model-based 3d tracking of an articulated hand. In *Computer Vision and Pattern Recognition*, 2001.

[30] X. Sun, Y. Wei, S. Liang, X. Tang, and Sun. J. Cascaded hand pose regression. In *Computer Vision and Pattern Recognition*. IEEE, 2015.

[31] J. S. Supancic, G. Rogez, Y. Yang, J. Shotton, and D. Ramanan. Depth-based hand pose estimation: methods, data, and challenges. *Computer Vision (ICCV), IEEE International Conference on*, 2015.

[32] D. Tang, H. J. Chang, A. Tejani, and T. K. Kim. Latent regression forest: Structured estimation of 3d articulated hand posture. In *Computer Vision and Pattern Recognition*, 2014.

[33] J. Tompson, R. Goroshin, A. Jain, Y. LeCun, and C. Bregler. Efficient object localization using convolutional networks. *IEEE Conference on Computer Vision and Pattern Recognition*, 2015.

[34] J. Tompson, M. Stein, Y. Lecun, and K. Perlin. Real-time continuous pose recovery of human hands using convolutional networks. In *Transactions on Graphics*, 2014.

[35] J. J. Tompson, A. Jain, Y. LeCun, and C. Bregler. Joint training of a convolutional network and a graphical model for human pose estimation. In *Advances in Neural Information Processing Systems*, pages 1799–1807, 2014.

[36] A. Toshev and C. Szegedy. Deeppose: Human pose estimation via deep neural networks. In *Computer Vision and Pattern Recognition*, 2014.

[37] C. Xu and L. Cheng. Efficient hand pose estimation from a single depth image. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 3456–3462. IEEE, 2013.