Depth-Based 3D Hand Pose Estimation:
From Current Achievements to Future Goals

Shanxin Yuan\textsuperscript{1} Guillermo Garcia-Hernando\textsuperscript{1} Björn Stenger\textsuperscript{2}
Gyeongsik Moon\textsuperscript{4} Ju Yong Chang\textsuperscript{5} Kyoung Mu Lee\textsuperscript{4} Pavlo Molchanov\textsuperscript{6}
Jan Kautz\textsuperscript{6} Sina Honari\textsuperscript{7} Liuhao Ge\textsuperscript{8} Junsong Yuan\textsuperscript{9} Xinghao Chen\textsuperscript{10} Gujin Wang\textsuperscript{10}
Fan Yang\textsuperscript{11} Kai Akiyama\textsuperscript{11} Yang Wu\textsuperscript{11} Qingfu Wan\textsuperscript{12} Meysam Madadi\textsuperscript{13} Sergio Escalera\textsuperscript{3,14}
Shile Li\textsuperscript{15} Dongheui Lee\textsuperscript{15,16} Iason Oikonomidis\textsuperscript{3} Antonis Argyros\textsuperscript{3} Tae-Kyun Kim\textsuperscript{1}

Abstract

In this paper, we strive to answer two questions: What is the current state of 3D hand pose estimation from depth images? And, what are the next challenges that need to be tackled? Following the successful Hands In the Million Challenge (HIM2017), we investigate the top 10 state-of-the-art methods on three tasks: single frame 3D pose estimation, 3D hand tracking, and hand pose estimation during object interaction. We analyze the performance of different CNN structures with regard to hand shape, joint visibility, view point and articulation distributions. Our findings include: (1) isolated 3D hand pose estimation achieves low mean errors (10 mm) in the view point range of [70, 120] degrees, but it is far from being solved for extreme view points; (2) 3D volumetric representations outperform 2D CNNs, better capturing the spatial structure of the depth data; (3) Discriminative methods still generalize poorly to unseen hand shapes; (4) While joint occlusions pose a challenge for most methods, explicit modeling of structure constraints can significantly narrow the gap between errors on visible and occluded joints.

1. Introduction

The field of 3D hand pose estimation has advanced rapidly, both in terms of accuracy [2, 4, 5, 6, 7, 26, 28, 32, 38, 45, 48, 49, 53, 54, 57, 60, 62] and dataset quality [10, 22, 43, 47, 52, 59]. Most successful methods treat the estimation task as a learning problem, using random forests or convolutional neural networks (CNNs). However, a review from 2015 [44] surprisingly concluded that a simple nearest-neighbor neural networks outperforms most existing systems. It concluded that systems do not generalize beyond their training sets [44], highlighting the need for more and better data. Manually labeled datasets such as [31, 41] contain just a few thousand examples, making them unsuitable for large-scale training. Semi-automatic annotation methods, which combine manual annotation with tracking, help scaling the dataset size [43, 46, 52], but in the case of [46] the annotation errors are close to the lowest estimation errors. Synthetic data generation solves the scaling issue, but has not yet closed the realism gap, leading to some kinematically implausible poses [37].

A recent study confirmed that cross-benchmark testing is poor due to different capture set-ups and annotation methods [59]. It showed that training a standard CNN on a million-scale dataset achieves state-of-the-art results. However, the estimation accuracy is not uniform, highlighting the well-known challenges of the task: variations in view point and hand shape, self-occlusion, and occlusion caused by objects being handled.

In this paper we analyze the top methods of the HIM2017 challenge [58]. The benchmark dataset includes data from BigHand2.2M [59] and the First-Person Hand Action dataset (FHAD) [10], allowing the comparison of different algorithms in a variety of settings. The challenge considers three different tasks: single-frame pose estimation, tracking, and hand-object interaction. In the evaluation we consider different network architectures, preprocessing strategies, and data representations. Over the course of the challenge the lowest mean 3D estimation error could be reduced from 20 mm to less than 10 mm. This paper analyzes the errors with regard to seen and unseen subjects, joint visibility, and view point distribution. We conclude by providing insights for designing the next generation of methods.
Related work. Public benchmarks and challenges in other areas such as ImageNet [35] for scene classification and object detection, PASCAL [9] for semantic and object segmentation, and the VOT challenge [19] for object tracking, have been instrumental in driving progress in their respective field. In the area of hand tracking, the review from 2007 by Erol et al. [8] proposed a taxonomy of approaches. Learning-based approaches have been found effective for solving single-frame pose estimation, optionally in combination with hand model fitting for higher precision, e.g., [50]. The review by Supancic et al. [44] compared 13 methods on a new dataset and concluded that deep models are well-suited to pose estimation [44]. It also highlighted the need for large-scale training sets in order to train models that generalize well. In this paper we extend the scope of previous analyses by comparing deep learning methods on a large-scale dataset, carrying out a fine-grained analysis of error sources and different design choices.

2. Evaluation tasks

We evaluate three different tasks on a dataset containing over a million annotated images using standardized evaluation protocols. Benchmark images are sampled from two datasets: BigHand2.2M [59] and First-Person Hand Action dataset (FHAD) [10]. Images from BigHand2.2M cover a large range of hand view points (including third-person and first-person views), articulated poses, and hand shapes. Sequences from the FHAD dataset are used to evaluate pose estimation during hand-object interaction. Both datasets contain $640 \times 480$-pixel depth maps with 21 joint annotations, obtained from magnetic sensors and inverse kinematics. The 2D bounding boxes have an average diagonal length of 162.4 pixels with a standard deviation of 40.7 pixels. The evaluation tasks are 3D single hand pose estimation, i.e., estimating the 3D locations of 21 joints, from (1) individual frames, (2) video sequences, given the pose in the first frame, and (3) frames with object interaction, e.g., with a juice bottle, a salt shaker, or a milk carton. See Figure 1 for an overview. Bounding boxes are provided as input for tasks (1) and (3). The training data is sampled from the BigHand2.2M dataset and only the interaction task uses test data from the FHAD dataset. See Table 1 for dataset sizes and the number of total and unseen subjects for each task.

![Figure 1: Evaluated tasks.](image)

For each scenario the goal is to infer the 3D locations of the 21 hand joints from a depth image. In Single frame pose estimation (left) and the Interaction task (right), each frame is annotated with a bounding box. In the Tracking task (middle), only the first frame of each sequence is fully annotated.

### Table 1: Data set sizes and number of subjects.

<table>
<thead>
<tr>
<th>Number of</th>
<th>Train</th>
<th>Test single</th>
<th>Test track</th>
<th>Test interact</th>
</tr>
</thead>
<tbody>
<tr>
<td>frames</td>
<td>957K</td>
<td>295K</td>
<td>294K</td>
<td>2,965</td>
</tr>
<tr>
<td>subjects (unseen)</td>
<td>5</td>
<td>10 (5)</td>
<td>10 (5)</td>
<td>2 (0)</td>
</tr>
</tbody>
</table>

3. Evaluated methods

We evaluate the top 10 among 17 participating methods [58]. Table 2 lists the methods with some of their key properties. We also indirectly evaluate DeepPrior [29] and REN [15], which are components of rvhand [1], as well as DeepModel [61], which is the backbone of LSL [20]. We group methods based on different design choices.

### 2D CNN vs. 3D CNN. 2D CNNs have been popular for 3D hand pose estimation [1, 3, 14, 15, 20, 21, 23, 29, 57, 61]. Common pre-processing steps include cropping and resizing the hand volume by normalizing the depth values to [-1, 1]. Recently, several methods have used a 3D CNN [12, 24, 56], where the input can be a 3D voxel grid [24, 56], or a projective D-TSDF volume [12]. Ge et al. [13] project the depth image onto three orthogonal planes and train a 2D CNN for each projection, then fusing the results. In [12] they propose a 3D CNN by replacing 2D projections with a 3D volumetric representation (projective D-TSDF volumes [40]). In the HIM2017 challenge [58], they apply a 3D deep learning method [11], where the inputs are 3D points and surface normals. Moon et al. [24] propose a 3D CNN to estimate per-voxel likelihoods for each hand joint. NAIST\_RV [56] proposes a 3D CNN with a hierarchical branch structure, where the input is a 50$^3$-voxel grid.

Detection-based vs. Regression-based. Detection-based methods [23, 24] produce a probability density map for each joint. The method of RCN-3D [23] is an RCN+ network [17], based on Recombinator Networks (RCN) [18] with 17 layers and 64 output feature maps for all layers except the last one, which outputs a probability density map for each of the 21 joints. V2V-PoseNet [24] uses a 3D CNN to estimate per-voxel likelihood of each joint, and a CNN to estimate the center of mass from the cropped depth map. For training, 3D likelihood volumes are generated by placing normal distributions at the locations of hand joints. Regression-based methods [1, 3, 11, 14, 20, 21, 29, 56]
directly map the depth image to the joint locations or the joint angles of a hand model [39, 61]. \textit{rvhand} [1] combines ResNet [16], Region Ensemble Network (\textit{REN}) [15], and DeepPrior [29] to directly estimate the joint locations. \textit{LSL} [20] uses one network to estimate a global scale factor and a second network [61] to estimate all joint angles, which are fed into a forward kinematic layer to estimate the hand joints.

Hierarchical models divide the pose estimation problem into sub-tasks [1, 3, 15, 21, 56]. The evaluated methods divide the hand joints either by finger [21, 56], or by joint type [1, 3, 15]. \textit{mmadadi} [21] designs a hierarchically structured CNN, dividing the convolution+ReLU+pooling blocks into six branches (one per finger with palm and one for palm orientation), each of which is then followed by a fully connected layer. The final layers of all branches are concatenated into one layer to predict all joints. \textit{NAIST, rv} [56] chooses a similar hierarchical structure of a 3D CNN, but uses five branches, each to predict one finger and the palm. \textit{THU, VLCab} [3], \textit{rvhand} [1], and \textit{REN} [15] apply constraints per finger and joint-type (across fingers) in their multiple regions extraction step, each region containing a subset of joints. All regions are concatenated in the last fully connected layers to estimate the hand pose.

Structured methods embed physical hand motion constraints into the model [11, 20, 21, 29, 55, 61]. Structural constraints are included in the CNN model [20, 27, 29] or in the loss function [21, 55]. DeepPrior [29] learns a prior model and integrates it into the network by introducing a bottleneck in the last CNN layer. \textit{LSL} [20] uses prior knowledge in DeepModel [61] by embedding a kinematic model layer into the CNN and using a fixed hand model. \textit{mmadadi} [21] includes the structure constraints in the loss function, which incorporates physical constraints about natural hand motion and deformation. \textit{strawberryfg} [55] applies a structure-aware regression approach, Compositional Pose Regression [42], and replaces the original ResNet-50 with ResNet-152. It uses phalanges instead of joints for representing pose, and defines a loss function that encodes long-range interaction between the phalanges.

Multi-stage methods propagate results from each stage to enhance the training of the subsequent stages [3, 23]. \textit{THU, VLCab} [3] uses \textit{REN} [15] to predict an initial hand pose. In the following stages, feature maps are computed with the guidance of the hand pose estimate in the previous stage. \textit{RCN, 3D} [23] has five stages: (1) 2D landmark estimation using an RCN+ network [17], (2) estimation of corresponding depth values by multiplying probability density maps with the input depth image, (3) inverse perspective projection of the depth map to 3D, (4) error compensation for occlusions and depth errors (a 3-layer network of residual blocks), and, (5) error compensation for noise (another 3-layer network of residual blocks). Residual networks. ResNet [16] is adopted by several methods [1, 3, 15, 23, 24, 42]. V2V-PoseNet [24] uses residual blocks as main building blocks. \textit{strawberryfg} [55] implements the Compositional Pose Regression method [42] by using ResNet-152 as basic network. \textit{RCN, 3D} [23] uses two small residual blocks in its fourth and fifth stage.

### Table 2: Methods evaluated in the hand pose estimation challenge.

<table>
<thead>
<tr>
<th>Method</th>
<th>Model</th>
<th>Input</th>
<th>Ang. range (s, d, t)</th>
<th>3D</th>
<th>De</th>
<th>Hi</th>
<th>St</th>
<th>M</th>
<th>R</th>
</tr>
</thead>
<tbody>
<tr>
<td>V2V-PoseNet [24]</td>
<td>3D CNN, per-voxel likelihood of each joint</td>
<td>88×88×88 voxels</td>
<td>[0.8, 1.2] [-40, 40] [-8, 8]</td>
<td>✓</td>
<td>✓</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>RCN-3D [23]</td>
<td>RCN+ network [17] with 17 convolutional layers</td>
<td>80×80</td>
<td>[0.7, 1.1] [0, 360] [-8, 8]</td>
<td>x</td>
<td>✓</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>✓</td>
</tr>
<tr>
<td>oasis [11]</td>
<td>Hierarchical PointNet with three set abstraction levels and three full-connected layers</td>
<td>1024 3D points</td>
<td>random scaling*</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>NAIST, rv [56]</td>
<td>3D CNN with 5 branches, one for each finger</td>
<td>50×50×50 3D grid</td>
<td>[0.9,1.1] [-90, 90] [-15,15]</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>Vanora [34]</td>
<td>shallow CNN trained end-to-end</td>
<td>resized 2D</td>
<td>random scaling*</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>strawberryfg [55]</td>
<td>ResNet-152 + [22]</td>
<td>224×224 None</td>
<td>x</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>x</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>LSL [20]</td>
<td>ScaleNet to estimate hand scale + DeepModel [61]</td>
<td>128×128</td>
<td>[0.85,1.15] [0,360] [-20,20]</td>
<td>x</td>
<td>✓</td>
<td>x</td>
<td>✓</td>
<td>x</td>
<td>✓</td>
</tr>
</tbody>
</table>

* in both methods, hand segmentation is performed considering different hand arm lengths. **3D**, **De**, **Hi**, **St**, **M**, and **R** denote 3D CNN, Detection-based method, Hierarchical model, Structure model, Multistage model, and Residual net, respectively. The aim of this evaluation is to identify success cases and failure modes. We use both standard error metrics [30, 37, 51] and new proposed metrics to provide further insights. We consider joint visibility, seen vs. unseen subjects, hand view point distribution, articulation distribution, and per-joint accuracy.
4.1. Single frame pose estimation

Over the 6-week period of the challenge the lowest mean error could be reduced from 19.7 mm to 10.0 mm by exploring new model types and improving data augmentation, optimization and initialization. For hand shapes seen during training, the mean error was reduced from 14.6 mm to 7.0 mm, and for unseen hand shapes from 24.0 mm to 12.2 mm. Considering typical finger widths of 10-20 mm, these methods are becoming applicable to scenarios like pointing or motion capture, but may still lack sufficient accuracy for fine manipulation that is critical in some UI interactions.

We evaluate ten state-of-the-art methods (Table 2) directly and three methods indirectly, which were used as components of others, DeepPrior [29], REN [15], and DeepModel [61]. Figure 2 shows the results in terms of two metrics: (top) the proportion of frames in which all joint errors are below a threshold [51] and (bottom) the total proportion of joints below an error threshold [37].

Figure 3 (top-left) shows the success rates based on per-frame average joint errors [30] for a varying threshold. The top performer, V2V-PoseNet, estimates 70% of frames with a mean error of less than 10 mm, and 20% of frames with mean errors under 5 mm. All evaluated methods achieve a success rate greater than 80% with an average error of less than 20 mm.

As has been noted by [15, 27], data augmentation is beneficial, especially for small datasets. However, note that even though the top performing methods employ data augmentation, it is still difficult to generalize to hands from unseen subjects, see Table 3, with an error gap of around 6 mm between seen and unseen subjects. Some methods generalize better than others, in particular RCN-3D is the top performer on unseen subjects, even though it is not the best on seen subjects.

4.1.1 Annotation error

The annotation error takes into account inaccuracies due to small differences of 6D sensor placement for different subjects during the annotation, and uncertainty in the wrist joint location. To quantify this error we selected poses for which all methods achieved a maximum error [51] of less than 10 mm. We denote these as Easy Poses. The pose estimation task for these can be considered solved, as shown in Figure 3. The outputs of the top five methods are visu-
In all four cases, detection-based methods outperform regression-based ones, see the bottom-right plot of Figure 4. In the challenge, the top two methods are detection-based methods, see Table 2. Note that a similar trend can be seen in the field of full human pose estimation, where only one method in a recent challenge was regression-based [25].

**Hierarchical methods:** Hierarchical constraints can help in the case of occlusion. The hierarchical model in rvhand [1] shows similar performance on visible and occluded joints. rvhand [1] has better performance on occluded joints when the error threshold is smaller than 15 mm, see the bottom-right plot of Figure 4. The underlying REN module [15], which includes finger and joint-type constraints seems to be critical. Methods using only per-finger constraints, e.g., mmadadi [21] and NAISTRV [56], generalize less well to occluded joints, see the bottom-left plot of Figure 4.

**Structural methods:** We compare four structured methods LSL [20], mmadadi [21], rvhand [1], and strawberryf [55], see the bottom-middle plot of Figure 4. strawberryf [55] and mmadadi [21] have higher success rates when the error threshold is below 15 mm, while LSL [20] and rvhand [1] perform better for thresholds larger than 25 mm. Embedding structural constraints in the loss func-
tion has been more successful than including them within the CNN layers. strawberryfg [55] performs the best, using constraints on phalanges rather than on joints.

**Single- vs. multi-stage methods:** Cascaded methods work better than single-stage methods, see the bottom-right plot of Figure 4. Compared to other methods, rvhand [1] and THU_VCLab [3] both embed structural constraints, employing REN as their basic structure. THU_VCLab [3] takes a cascaded approach to iteratively update results from previous stages, outperforming rvhand [1].

### 4.1.3 Analysis by number of occluded joints

Most frames contain joint occlusions, see Figure 5 (top). We assume that a visible joint lies within a small range of the 3D point cloud. We therefore detect joint occlusion by thresholding the distance between the joint’s depth annotation value and its re-projected depth value. As shown in Figure 5 (bottom), the average error decreases nearly monotonously for increasing numbers of visible joints.

### 4.1.4 Analysis based on view point

The view point is defined as the angle between the palm and camera directions. The test data covers a wide range of view points for the Single frame pose estimation task, see Figure 6 (top). View points in the [70, 120] range have a low mean error of below 10 mm. View points in the [0, 10] range have a significantly larger error due to the amount of self occlusion. View points in the [10, 30] range have an average error of 15 – 20 mm. View point ranges of [30,70] and [120, 180] show errors of 10 – 15 mm. Third-person and egocentric views are typically defined by the hand facing toward or away from the camera, respectively. However, as shown in Figure 6, there is no clear separation by view point, suggesting a uniform treatment of both cases is sensible. Note that RCN-3D [23] outperforms others with a margin of 2-3 mm on extreme view points in the range of [150,180] degrees due to their depth prediction stage.

### 4.1.5 Analysis based on articulation

We evaluate the effect of hand articulation on estimation accuracy, measured as the average of 15 finger flexion angles, see Figure 7. To reduce the influence from other factors such as view point, we select frames with view point angles within the range of [70, 120]. We evaluate the top five performers, see Figure 7 (bottom). For articulation angles smaller than 30 degrees, the mean error is 7 mm, when the average articulation angle increases to the range of [35, 70], errors increase to 9-10 mm. When the articulation angle is larger than 70 degrees, close to a fist pose, the mean error increases to over 12 mm.

### 4.1.6 Analysis by joint type

As before we group joints according to their visibility and the presence of the subject in the training set. We report the top five performers, see Figure 10. For the easiest case (visible joints of seen subjects), all 21 joints have a similar average error of 6 – 10 mm. For seen subjects, along the kinematic hand structure from the wrist to finger tips, occluded joints have increasingly larger errors, reaching 14 mm in the
finger tips. Visible joints of unseen subjects have larger errors (10–13 mm) than that of seen subjects. Occluded joints of unseen subjects have the largest errors, with a relatively smaller error for the palm, and larger errors for finger tips (24 – 27 mm). We draw two conclusions: (1) all the top performers have difficulty in generalizing to hands from unseen subjects, (2) occlusions have more effect on finger tips than other joints. An interesting observation is that middle and ring fingers tend to have smaller errors in MCP and PIP joints than other fingers. One reason may be that the motion of these fingers is more restricted. The thumb’s MCP joint has a larger error than for other fingers, because it tends to have more discrepancy among different subjects.

Figure 8: Error curves for hand tracking. Left: number of frames with IoU below threshold. Middle: success rate for the detection part of each method in two cases. Right: success rate for three methods in four cases.

Figure 9: Error curves for hand-object interaction. Left: success rate for each method using average error per-frame. Middle: success rate for visible and occluded joints. Right: average error for each joint.

Figure 10: Average error of the top five methods for each joint in the Single frame pose estimation task. Finger tips have larger errors than other joints. For non-tip joints, joints on ring finger and middle finger have lower average errors than other fingers. ‘T’, ‘I’, ‘M’, ‘R’, ‘P’ denotes ‘Thumb’, ‘Index’, ‘Middle’, ‘Ring’, and ‘Pinkly’ finger, respectively.

<table>
<thead>
<tr>
<th>Method</th>
<th>Model</th>
<th>AVG</th>
</tr>
</thead>
<tbody>
<tr>
<td>RCN-3D_track</td>
<td>scanning window + post-processing + pose estimator</td>
<td>10.5</td>
</tr>
<tr>
<td>NAIST_RV_track</td>
<td>hand detector [34] + hand verifier + pose estimator</td>
<td>12.6</td>
</tr>
<tr>
<td>THU_VCLab_track</td>
<td>Estimation [3] with the aid of tracking + re-initialization [33]</td>
<td>13.7</td>
</tr>
</tbody>
</table>

Table 4: Methods evaluated on 3D hand pose tracking. The last column is the average error in mm for all frames.

4.2. Hand pose tracking

In this task we evaluate three state-of-the-art methods, see Table 4 and Figure 8. Discriminative methods [3, 23, 56] break tracking into two sub-tasks: detection and hand pose estimation, sometimes merging the sub-tasks [3]. Based on the detection methods, 3D hand pose estimation can be grouped into pure tracking [23], tracking-by-detection [56], and a combination of tracking and re-initialization [3], see Table 4.

Pure tracking: RCN-3D_track estimate the bounding box location by scanning windows based on the result in the previous frame, including a motion estimate. Hand pose within the bounding box is estimated using RCN-3D [23].

Tracking-by-detection: NAIST_RV_track is a tracking-by-detection method with three components: hand detector, hand verifier, and pose estimator. The hand detector is built on U-net [34] to predict a binary hand-mask, which, after
Current state-of-the-art methods have difficulty generalizing to the hand-object interaction scenario. However, NAIST_RV_obj [56] and rvhand_obj [1] show similar performance for visible joints and occluded joints, indicating that CNN-based segmentation can better preserve structure than image processing operations, see the middle plot of Figure 9.

5. Discussion and conclusions

The analysis of the top 10 among 17 participating methods from the HIM2017 challenge [58] provides insights into the current state of 3D hand pose estimation.

(1) 3D volumetric representations used with a 3D CNN show high performance, possibly by better capturing the spatial structure of the input depth data.

(2) Detection-based methods tend to outperform regression-based methods, however, regression-based methods can achieve good performance using explicit spatial constraints. Making use of richer spatial models, e.g., bone structure [42], helps further. Regression-based methods perform better in extreme view point cases [23], where severe occlusion occurs.

(3) While joint occlusions pose a challenge for most methods, explicit modeling of structure constraints and spatial relation between joints can significantly narrow the gap between errors on visible and occluded joints [1].

(4) Discriminative methods still generalize poorly to unseen hand shapes. Data augmentation and scale estimation methods model only global shape changes, but not local variations. Integrating hand models with better generative capability may be a promising direction.

(5) Isolated 3D hand pose estimation achieves low mean errors (10 mm) in the view point range of [70, 120] degrees. However, errors remain large for extreme view points, e.g., view point range of [0,10], where the hand is facing away from the camera. Multi-stage methods [23] tend to perform better in these cases.

(6) In hand tracking, current discriminative methods divide the problem into two sub-tasks: detection and pose estimation, without using the hand shape provided in the first frame. Hybrid methods may work better by using the provided hand shape.

(7) Current methods perform well on single hand pose estimation when trained on a million-scale dataset, but have difficulty in generalizing to hand-object interaction. Two directions seem promising, (a) designing better hand segmentation methods, and (b) training the model with large datasets containing hand-object interaction.

Acknowledgement: This work was partially supported by Huawei Technologies.
References