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

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

[1]Imperial College London [2]Rakuten Institute of Technology [3]University of Crete and FORTH [4]ASRI, Seoul National University [5]Kwangwoon University [6]NVIDIA Research [7]University of Montreal [8]Nanyang Technological University [9]Tsinghua University [10]Nara Institute of Science and Technology [11]Fudan University [12]Computer Vision Center, Universitat Autònoma de Barcelona [13]University of Barcelona [14]Human-centered Assistive Robotics, Technical University of Munich (TUM) [15]Institute of Robotics and Mechatronics, German Aerospace Center (DLR)

Abstract

In this paper, we strive to answer two questions: What is the current state of 3D hand pose estimation? And, what are the next challenges that need to be tackled? Following the successful Hands In the Million Challenge (HIM2017)*, we investigate 11 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 [40, 150] 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 performance [27, 29, 46, 45, 51, 54, 51, 57, 58, 6, 4, 50, 37, 25, 5, 4, 60, 3, 31] and dataset quality [9, 42, 44, 49, 56, 21]. The most successful methods treat the estimation task as a learning problem, using random forests or convolutional neural networks (CNNs). However, a review from 2015 [43] surprisingly concluded that *a simple nearest-neighbor baseline outperforms most existing systems*. It concluded that *most systems do not generalize beyond their training sets* [43], highlighting the need for more (and better) data. Manually labeled datasets such as [30, 40] contain just a few thousand examples, making them unsuitable for large-scale training. Semi-automatic annotation, combining manual annotation with tracking, helps to scale the dataset size [42, 44, 49], but in the case of [44] the annotation error can be of the same order as 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 [36].

A recent study confirmed that cross-benchmark testing is poor due to different capture set-ups and annotation methods [56]. 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. These include variations in view point and hand shape, self-occlusion, and occlusion caused by objects being handled.

In this paper we conduct a series of analyses, looking at how methods perform in these scenarios. The *Hands In the Million (HIM2017)* challenge [55] serves as a test-bed for this analysis. This benchmark dataset includes data from *BigHand2.2M* [56] and the *First-Person Hand Action dataset (FHAD)* [9], 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. We aim to answer the question of where we are, as a field, in terms of accuracy. We consider network architectures, preprocessing strategies and data representations. Over the course of the challenge the best average 3D estimation error could be reduced from 20 to less than $10\,\text{mm}$. This paper analyzes the errors with regard to seen and unseen subjects, joint visibility, and view point distribution.

We conclude with providing a number of insights that can be useful when designing the next generation of methods.

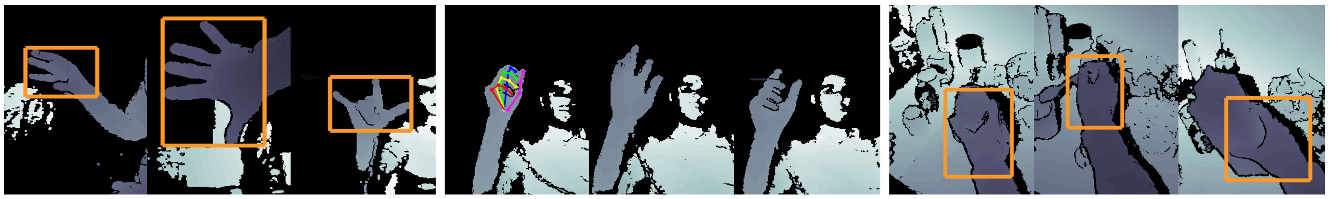
Figure 1. **Evaluated tasks**. For each scenario the goal is to infer the 3D locations of the 21 hand joints from a depth image. **Left: Single frame pose estimation**, each frame is annotated with a bounding box. **Middle: Tracking task**, first frame of each sequence is fully annotated. **Right: Interaction task**, each frame is annotated with a bounding box.

Related work. Public benchmarks and challenges in other areas such as ImageNet [34] for scene classification and object detection, PASCAL [8] for semantic and object segmentation, and the VOT challenge [18] for visual object tracking, have been instrumental in driving progress in their respective area. In the area of hand tracking, the review from 2007 by Erol *et al.* [7] proposed a taxonomy of the different methods. Generally, machine learning approaches have been found effective for solving single-frame pose estimation, possibly in combination with iterative model optimization to achieve a more precise fit, *e.g.*[47]. The review by Supancic *et al.* [43] compared 13 methods on a new dataset and concluded that deep models seem well-suited to the pose estimation task [43]. 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 more than a million annotated images based on a standardized evaluation protocol. The benchmark images were sampled from two datasets: *BigHand2.2M* [56] and *First-Person Hand Action dataset (FHAD)* [9]. Images from *BigHand2.2M* cover a large range of hand view points (including third-and first-person views), articulated poses, and hand shapes. Sequences extracted from *FHAD* are used to evaluate hand pose estimation in the hand-object interaction scenario. Both datasets contain 640 × 480-depth maps with 21 joint annotations obtained from magnetic sensors and inverse kinematics. The evaluation tasks are 3D single hand pose estimation 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, salt shaker, knife, or milk carton. See Figure 1 for an overview. For each scenario, the goal is to accurately estimate the 3D locations of 21 joints. Bounding boxes are provided as input for tasks (1) and (3). The training data is sampled from the *BigHand2.2M* dataset. 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.

Number of	Train	Test single	Test track	Test interact
frames	957K	295K	294K	2965
subjects (unseen)	5	10 (5)	10 (5)	2 (0)

Table 1. **Data set sizes and number of subjects.**

3. Evaluated methods

We evaluate the top 10 methods among 17 participating methods in total [55], and use the *Holi CNN* from [54] as baseline, which shows state-of-the-art performance [56] and is easy to implement. Table 2 lists the methods and some of their key properties. We indirectly evaluate *DeepPrior* [28] and *REN* [14], which are components of *rvhand* [1], as well as *DeepModel* [59], which is the backbone of *LSL* [19]. In the following we group methods based on different design choices.

2D CNN *vs.* 3D CNN. 2D CNNs are commonly used in 3D hand pose estimation [22, 2, 12, 41, 1, 20, 19, 54, 28, 59, 14]. Common pre-processing steps include cropping and resizing the hand volume, normalizing the depth values to [-1, 1]. For example, the input for *slivorezzz* [22] is the cropped depth image resized to 80×80 pixels, while the input for *THU VCLab* [2] is a cropped depth image of size 96×96 (see Table 2, third column). Recently, several methods use a 3D CNN [13, 10, 23, 53], where the input can be a 3D voxel grid [23, 53], or a projective D-TSDF volume [10]. Ge *et al.* [11] projected the depth image onto three orthogonal planes and trained a 2D CNN for each projection, then fusing the results. They later proposed a 3D CNN [10] by replacing 2D projections with a 3D volumetric representation (projective D-TSDF volumes [39]). In the *HIM2017* [55] challenge, they propose an end-to-end trained 3D CNN [13]. *mks0601* [23] proposes a 3D CNN to estimate the per-voxel likelihood for each hand joint. The network has has a encoder and decoder. *NAIST_RV* [53] propose a 3D CNN with a hierarchical branch structure, where the input is a 50×50×50 3D grid.

Detection-based *vs.* Regression-based. Detection-based methods [23, 22, 11, 41, 59] produce a probability density map for each joint. The method by *slivorezzz* [22] is an RCN+ network [16] (based on Recombinator Networks (RCN) [17]) with 17 layers and 64 output feature maps for all layers except the last one, which outputs a probability

Method	**Model**	**Input**	**Aug. range (s,θ,t)**	**3D**	**De**	**Hi**	**St**	**M**	**R**
mks0601 [23]	3D CNN, per-voxel likelihood for each joint	88×88×88 voxels	[0.8, 1.2] [-40,40] [-8,8]	✓	✓	✗	✗	✗	✓
slivorezzz [22]	RCN+ network [16] with 17 convolutional layers	80×80	[0.7, 1.1] [0, 360] [-8,8]	✗	✓	✗	✗	✓	✓
oasis [13]	3D CNN with 8 conv layers and 3 fully connected layers	3D hand volume	random scaling*	✓	✓	✗	✗	✗	✗
THU_VCLab [2]	Pose-REN [2]: *REN* [14] + cascaded + hierarchical.	96×96	[0.9,1.1] [-45,45] [-5,5]	✗	✗	✓	✗	✓	✓
NAIST_RV [53]	3D CNN with 5 branches, one for each finger	50×50×50 3D grid	[0.9,1.1] [-90, 90] [-15,15]	✓	✗	✓	✗	✗	✗
Vanora [12]	shallow CNN trained end-to-end	resized 2D	random scaling*	✗	✗	✗	✗	✗	✗
strawberryfg [52]	ResNet-152 + [41]	224×224	None	✗	✗	✗	✓	✗	✓
rvhand [1]	ResNet [15] + *REN* [14] + Deep Prior [28]	192×192	[0.9,1.1] [-90, 90] [-15,15]	✗	✗	✓	✓	✗	✓
mmadadi [20]	Hierarchical tree-like structured CNN [20]	192×192	[random] [-30, 30] [-10,10]	✗	✗	✓	✓	✗	✗
LSL [19]	ScaleNet to estimate hand scale + *DeepModel* [59]	128×128	[0.85,1.15] [0,360] [-20,20]	✗	✗	✗	✓	✗	✗
baseline	CNN with 3 conv. layers (*Holi CNN* of [54])	96×96 3 scales	None	✗	✗	✗	✗	✗	✗

Table 2. **Methods evaluated in the hand pose estimation challenge.** Methods are ordered by average error on the leader-board. **3D**, **De**, **Hi**, **St**, **M**, and **R** denote **3D CNN**, **Detection-based method**, **Hierarchical model**, **Structured model**, **Multistage model**, and **Residual net**, respectively. * in both methods, hand segmentation is performed considering different hand arm lengths.

density map for each of the 21 joints. *mks0601* [23] use a 3D CNN to estimate per-voxel likelihood for each joint, and a simple CNN to estimate the correct center of mass from the cropped depth map. For training, 3D likelihood volumes were generated by placing normal distributions at the locations of hand joints.

Regression-based methods [2, 53, 28, 12, 20, 1, 19] directly map the depth image to the joint locations or the joint angle parameters of a hand model [38, 59]. *rvhand* [1] combines ResNet [15], Region Ensemble Network (*REN*) [14], and *DeepPrior* [28] to directly estimate the joint locations. *LSL* [19] used one network to estimate a global scale factor and a second network [59] to estimate all joint angles, which were fed to a forward kinematic layer to estimate the hand joints.

Hierarchical models divide the pose estimation problem into sub-tasks [2, 14, 20, 53, 1]. The evaluated methods divide the hand joints either by finger [20, 53], or by joint type [2, 14, 1]. *mmadadi* [20] 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. *NAIST_RV* [53] choose a similar hierarchical structure of a 3D CNN, but use five branches, each to predict one finger with the palm. *THU VCLab* [2], *rvhand* [1] and *REN* [14] implicitly apply constraints for both fingers and joint-type (across fingers) in their multiple regions extraction step, each region contains a certain subset of joints. All regions are concatenated in the last fully connected layers to estimate the full hand pose.

Structured methods embed physical hand motion constraints into the model [19, 20, 28, 52, 59]. Structural constraints are included in the CNN model [28, 26, 19] or in the loss function [20, 52]. *DeepPrior* [28] learns a prior model and integrates it into the network by introducing a *bottleneck* in the last layer of the CNN. *LSL* [19] uses prior knowledge in *DeepModel* [59] by embedding a kinematic model layer into the CNN and estimated model parameters of a pre-defined fixed hand model. *mmadadi* [20] includes the structure constraints in its loss function, which incorporates physical constraints about natural hand motion and deformation. *strawberryfg* [52] applies a structure-aware regression approach, Compositional Pose Regression [41], and replaces the original ResNet-50 with ResNet-152. It uses phalanges instead of joints as pose representation and defines a compositional 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 [2, 22]. *THU_VCLab* [2] use *REN* [14] to predict an initial hand pose. In the following stages, feature regions on the feature maps are extracted under the guidance of the previous estimated hand pose to estimate the current pose. *slivorezzz* [22] has five stages: (1) 2D landmark estimation using an RCN+ network [16], (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 block) and, finally (5) error compensation for noise (another 3-layer network of residual block).

Residual networks. ResNet [15] is adopted by many

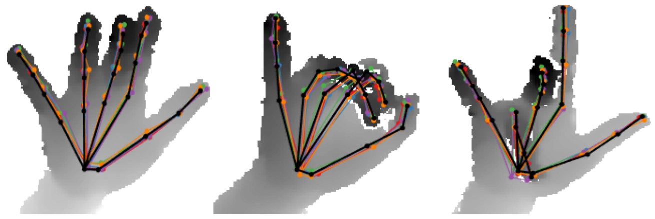
Figure 2. **Estimating annotation errors.** A few examples of *Easy Poses* overlayed with estimates by the top five methods (shown in different colors). Estimates are close to the ground truth (black), but disagree slightly.

Method \ Scenario	Seen Visible	Unseen Visible	Seen Occ	Unseen Occ
mks0601	**6.2**	11.1	**8.0**	**14.6**
slivorezzz	6.9	**10.6**	9.0	14.8
oasis	8.2	12.4	9.8	14.9
THU_VCLab	8.4	12.5	10.2	16.1
NAIST_RV	8.8	13.1	10.1	15.6
Vanora	8.8	12.9	10.5	15.5
strawberryfg	9.3	16.4	10.7	18.8
rvhand	12.2	16.1	11.9	17.6
mmadadi	10.6	15.6	13.6	19.7
LSL	11.8	18.1	13.1	19.2
Baseline	15.1	24.9	13.9	22.5
All	9.7	11.0	14.9	17.2
Top5	7.7	9.4	11.9	15.2

Table 3. **Mean error (in mm) for different scenarios in the single frame pose estimation task.** 'Occ' denotes 'Occluded'.

methods [1, 14, 22, 23, 41, 2]. *mks0601* [23] used residual blocks as main building blocks. *strawberryfg* [52] implement the Compositional Pose Regression method [41] by using ResNet-152 as basic network. *slivorezzz* [22] use two small residual blocks in its fourth and fifth stage.

4. Results

The aim of this evaluation is to identify success cases and failure modes. We use both standard error metrics [29, 48] 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

For this task, we evaluated eleven state-of-the-art methods (Table 2) and indirectly evaluated three methods that were used as components, *DeepPrior* [28], *REN* [14], and *DeepModel* [59]. Figure 3 shows the error curves. We first analyze errors with regard to different sources, including unseen hand shape, joint occlusion, hand view point, and hand articulation.

As has been noted by [14, 26], data augmentation is beneficial, especially when training data is small. Augmentation is used in nine of the evaluated methods, see Table 2. Even though the top six performers all employ data augmentation, it is still difficult to generalize to new unseen hand shapes, see Table 3. There still exists a gap of around 6 mm between seen and unseen subjects. However, some methods generalize better than others, *e.g. slivorezzz* performs the best on unseen subjects' test data, even though it is not the best on seen subjects.

4.1.1 Error for all methods and best performance

In Figure 3 (top-left), we draw the success rate (based on per-frame average joint error [29]) against a varying threshold. The top performer (*mks0601*) estimates 70% of frames with a mean error of less than 10 mm, and 20% of frames smaller than 5 mm. Across all the evaluated methods, all of them achieved a success rate larger than 60% with average error smaller than 20 mm.

Throughout the challenge, by applying new model types, as well as by experimenting with data augmentation, optimization and initialization methods, the average error was reduced from 19.7 mm to 10.0 mm. For seen and unseen subjects, the mean error dropped from 14.6 mm and 24.0 mm to 7.0 mm and 12.2 mm, respectively. 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.

4.1.2 Annotation error

To quantify the annotation error, we selected poses, for which all methods achieved a maximum error [48] of less than 10 mm. We denote these as *Easy Poses*, see Figure 2 for examples. The pose estimation task for these can be considered solved, as shown in Figure 2. The estimates of the top five methods are visually accurate and are close to the ground truth, see Figure 3 (top-left). We approximate the *Annotation Error* by the error on these poses. The error is 2.8 mm with a std dev of 0.5 mm. The sources include annotation inaccuracies, *e.g.* small differences of 6D sensor placement for different subjects, and uncertainty in the wrist joint location.

4.1.3 Analysis by occlusion and unknown subject

Average error for four scenarios: To analyze the results with respect to visibility and hand shape, we partition the joints in the test data into four groups: visible joints of seen hands, occluded joints of seen hands, visible joints of unseen hands, and occluded joints of unseen hands. Unseen hand shape and occlusions are responsible for a large proportion of errors, see Table 3 and Figure 3 (top-middle). The

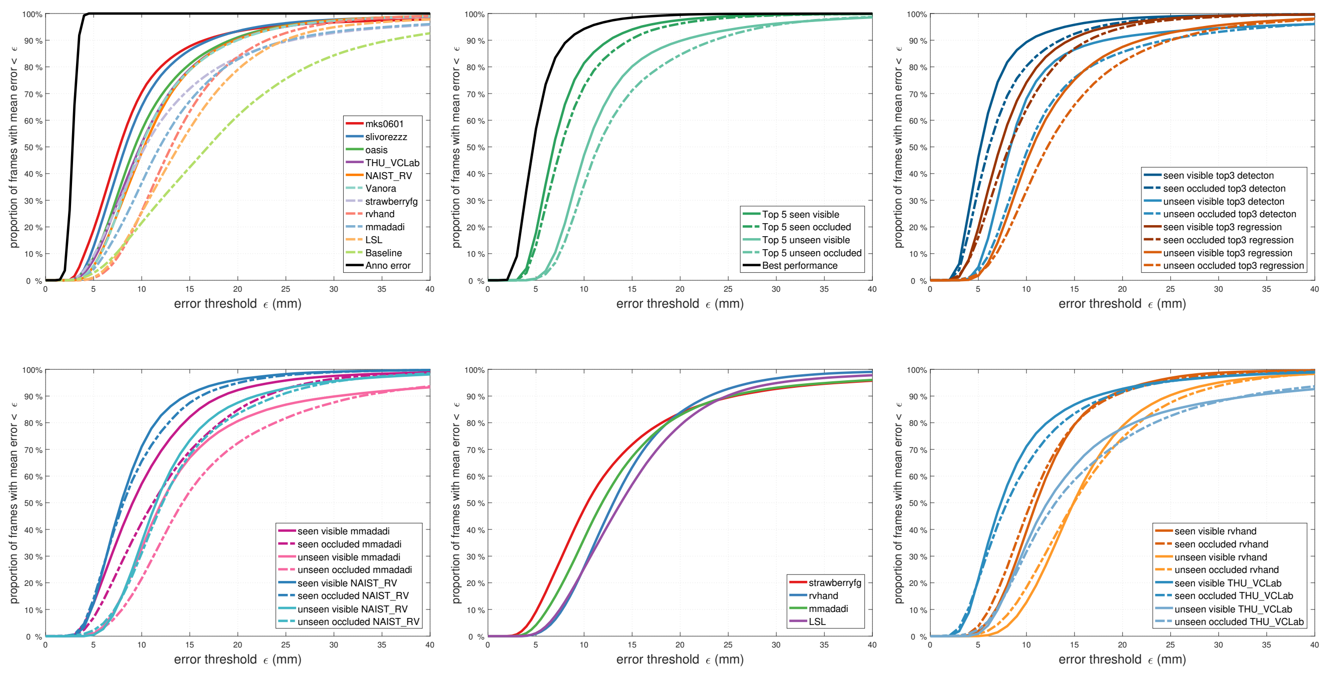

Figure 3. **Success rates for different methods.** **Top-left**: all evaluated methods using all test data. **Top-middle**: the average of top-five methods in four scenarios, the average error is 11mm. **Top-right**: the average of top-three detection-based and regression-based methods in four scenarios, the average error is 11mm. **Bottom-left**: direct comparison with 2D CNN and 3D CNN, *mmadadi* is a 2D CNN, *NAIST_RV* has the same structure but replaced 2D CNN with 3D CNN. **Bottom-middle**: comparison among structured methods. **Bottom-right**: compare a cascaded multistage method (*THU_VCLab*) with an one-off method (*rvhand*). Both of them use *REN* [14] as the backbone.

error for unseen subjects is significantly larger than for seen subjects. Moreover, the error for visible joints is smaller than for occluded joints. Based on the first group (visible, seen), we carry out a best-case performance estimate for the current state-of-the-art. For each frame of seen subjects, we first choose the best result from all methods, and calculate the success rate based on the average error for each frame, see Figure 3 (top-middle).

2D *vs*. 3D CNNs: We compare two hierarchical methods with similar structure but different representation. The bottom-left plot of Figure 3 shows *mmadadi* [20], which employs a 2D CNN, and *NAIST_RV* [53], using a 3D CNN. *mmadadi* and *NAIST_RV* have almost the same structure, but *NAIST_RV* [53] uses a 3D CNN while *mmadadi* [20] uses a 2D CNN. *NAIST_RV* [53] is better than *mmadadi* [20] for all four scenarios. Among the top 5 ranked methods, three are 3D CNNs, see Table 2.

Detection-based *vs*. regression-based methods: We compare the average of the top 3 detection-based methods with the average of the top 3 regression-based methods. In four scenarios, detection-based methods outperform regression-based methods, see the top-right plot of Figure 3. In the challenge, the top three 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 [24].

Hierarchical methods: Hierarchical constraints can help in the case of occlusion. The hierarchical models in *rvhand* [1] and *THU_VCLab* [2] both have similar performance on visible and occluded joints. *rvhand* [1] even have better performance on occluded joints when the error threshold is smaller than $15\,\mathrm{mm}$, see the bottom-right plot of Figure 3. The underlying *REN* [14] module, which includes finger and joint-type constraints seems to be critical. Methods using only per-finger constraints, *e.g*., *mmadadi* [20] and *NAIST_RV* [53], generalize less well to occluded joints, see the bottom-left plot of Figure 3.

Structural methods: We compare four structured methods *LSL* [19], *mmadadi* [20], *rvhand* [1], and *strawberryfg* [52], see the bottom-middle plot of Figure 3. *strawberryfg* [52] and *mmadadi* [20] have higher success rates when the error threshold is below $15\,\mathrm{mm}$, while *LSL* [19] and *rvhand* [1] perform better for thresholds larger than $25\,\mathrm{mm}$. This indicates that embedding structural constraints in the loss function is a better choice than into the CNN layers. *strawberryfg* [52] 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 3. Compared to other methods, *rvhand* [1] and *THU_VCLab* [2] both embed structural constraints, employing *REN* as their basic structure. *THU_VCLab* [2] takes a cascaded approach to iteratively update results from previous stages, outperforming *rvhand* [1].

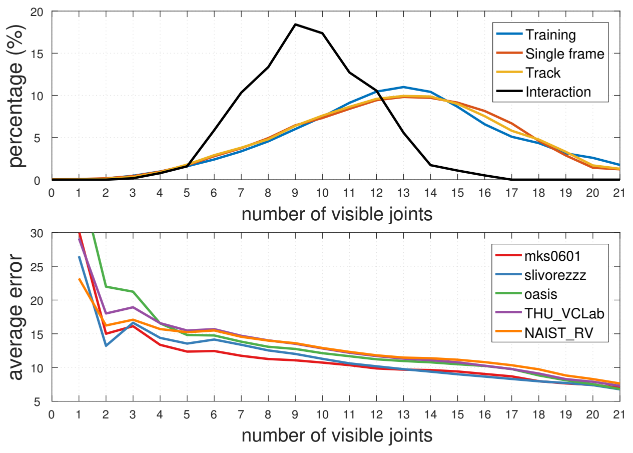

Figure 4. **Joint visibility. Top:** Joint visibility distributions for training set and testing sets. **Bottom:** Average error (mm) for different numbers of visible joints and different methods.

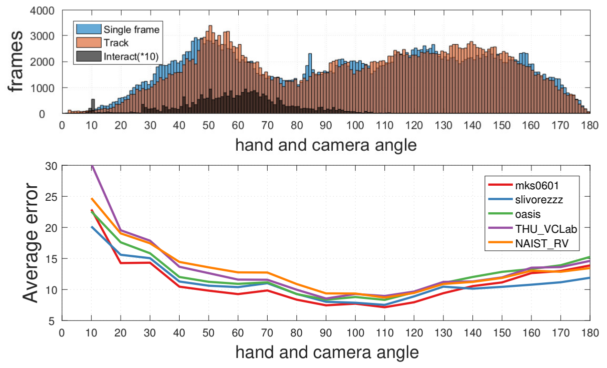

Figure 5. **View point distributions.** The error is significantly higher for small angles between hand and camera orientations.

4.1.4 Analysis by number of occluded joints

Most frames contain joint occlusions, see Figure 4 (top plot). We detect occlusion of a joint by thresholding the distance between the joint's depth annotation value and its re-projected depth value. A visible joint lies within a certain range behind the point cloud. As shown in Figure 4 (bottom plot), the average error decreases nearly monotonously with the increasing number of visible joints.

4.1.5 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 *Single frame pose estimation* task, see Figure 5 (top). View points in the [70, 120] range have a low mean error of 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 15-20 mm. View point ranges of [30,70] and [120, 180] have a similar error of 10-15 mm. Third-view and egocentric view are typically defined by the hand facing toward or away from the camera, respectively. However, as shown in Figure 5, there is no clear separation by view point, suggesting uniform treatment of both cases. Note that *slivorezzz* [22] outperforms others with a margin of 2-3 mm on extreme view points, [150,180] due to their depth prediction stage.

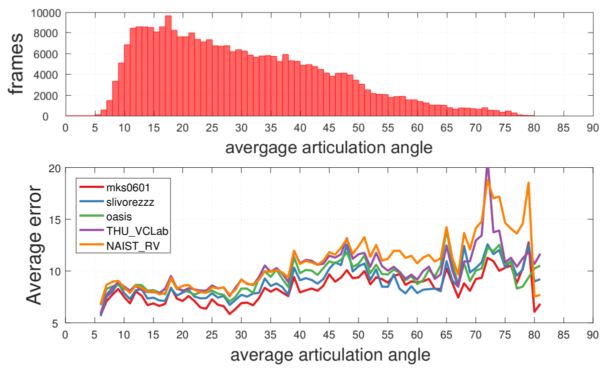

Figure 6. **Articulation distribution.** Errors increase for larger finger articulation angles, but there are less frames for these cases.

4.1.6 Analysis based on articulation

We evaluate the effect of hand articulation, measured as the average of 15 finger flexion angles, on estimation accuracy, see Figure 6. 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 6 (bottom plot). For articulation angles smaller than 30 degrees, the performance is 7 mm, when the average articulation angle get larger, the performance dropped to 9-10 mm (in the range of [35, 70]). When the average articulation angles is larger than 70, almost making a fist, the mean error increases to over 12 mm.

4.1.7 Analysis by joint type

Based on the visibility of each joint and whether the subject is in the training data or not, we cluster joints into four groups: visible joints of seen subjects, occluded joints of seen subjects, visible joints of unseen subjects and occluded joints of unseen subjects. We report the top five performers, see Figure 9. For the easiest case (the visible joints of seen subjects), all 21 joints have a similar average error of 6-7 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 consistently larger errors (10-13 mm) than that of seen subjects. Occluded joints of unseen subjects have the largest errors, with a relatively smaller error of the palm, and larger errors of finger tips (24-27 mm). We draw two conclusions: (1) all the top performers have difficulty in generalizing to new hand shapes,

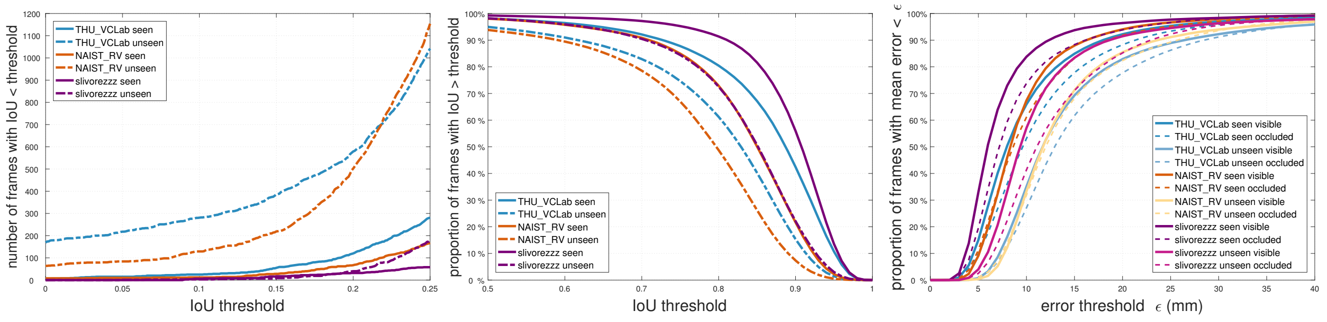

Figure 7. **Error curves for hand tracking. Left**: number of frames with IoU below threshold. **Middle**: success rate for the detection part of each method in two scenarios. **Right**: success rate for three methods in four scenarios.

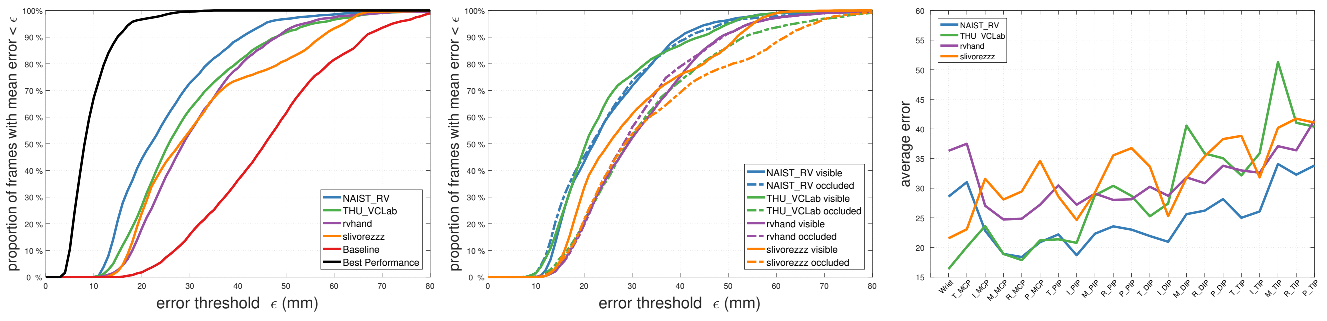

Figure 8. **Error curves for the hand-object interaction. Left**: success rate for each method using average error per-frame. **Middle**: success rate for visible and occluded joints. **Right**: average error for each joint.

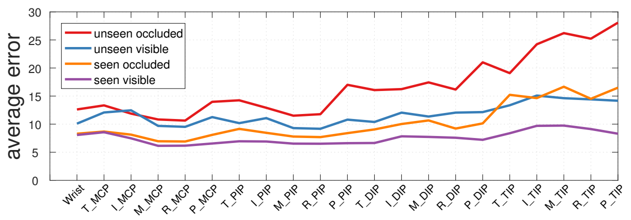

Figure 9. **Average error for each joint in the *Single frame pose estimation* task.** Finger tips have larger errors than other joints. For non-tip joints, joints on ring finger and middle finger have lower average errors than other fingers. 'T', 'I', 'M', 'R', 'P' denotes 'Thumb', 'Index', 'Middle', 'Ring', and 'Pinky' finger, respectively.

(2) occlusions have more effect on finger tips than other joints. An interesting observation is that middle and ring fingers tend to have smaller error 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.

4.2. Hand pose tracking

In this task we evaluate three state-of-the-art methods, see Table 4 and Figure 7. Discriminative methods [22, 53, 2] break tracking into two sub-tasks: detection and hand pose estimation, sometimes merging the sub-tasks [2].

Method	Model	AVG
slivorezzz_track [22]	scanning window + post-processing + pose estimator [22]	10.5
NAIST_RV_track [53]	hand detector [33] + hand verifier + pose estimator [53]	12.6
THU_VCLab_track [2]	Estimation [2] with the aid of tracking + re-initialization [32]	13.7

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

Based on the detection methods, 3D hand pose estimation can be grouped into pure tracking [29, 22], tracking-by-detection [53], and combination of tracking and re-initialization [2], see Table 4.

Pure tracking: *slivorezzz_track* estimate the bounding box location by scanning windows based-on the previous frame's result, including a motion estimate. Hand pose within the bounding box is estimated using *slivorezzz* [22].

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 developed from U-net [33] to predict a binary hand-mask, which, after verification, is passed to the pose estimator *NAIST_RV* [53]. If verification fails, the previous frame's result is chosen.

Hybrid tracking and detection: *THU_VCLab_track* [2] make use of the previous tracking result and the current frame's scanning window. The hand pose of previous frame is used as a guide to predict the hand pose in the current

Method	Model	AVG
NAIST_RV_obj [53]	Hand-object segmentation CNN + pose estimation [53]	25.0
THU_VCLab_obj [2]	Hand-object segmentation (intuitive) + pose estimation [2]	29.2
rvhand_obj [1]	Hand-object segmentation CNN + pose estimation [1]	31.3
slivorezzz_obj [22]	Two RCNs: Feature maps of first are used in the second RCN's stage 2.	32.4
baseline_obj [54]	Same as baseline [54]	46.1

Table 5. **Methods evaluated on hand pose estimation during hand-object interaction.** The last column is the average error (mm) for all frames.

frame. The previous frame's bounding box is used for the current frame. During fast hand motion, Faster R-CNN [32] is used for re-initialization.

Detection accuracy: We first evaluate the detection accuracy by evaluating the bounding box overlap, *i.e.*, IoU (Intersection over Union) of the detection and ground truth bounding boxes, see Figure 7 (middle). Overall, *slivorezzz_track* is better than *THU_VCLab_track*, which is better than *NAIST_RV_track*. Pure detection methods have a larger number of false negatives, especially when multiple hands appear in the scene, see Figure 7 (left). There are 72 and 174 missed detections (IoU of zero), for *NAIST_RV_track* and *THU_VCLab_track*, respectively. By tracking and re-initialization, *THU_VCLab_track* achieves better detection accuracy overall. *slivorezzz_track* using motion estimation and single-frame hand pose estimator has the lowest error.

Tracking accuracy is shown in Figure 7 (right). Even through *THU_VCLab* performs better than *NAIST_RV* in the *Single frame pose estimation* task, *NAIST_RV_track* performs better on the tracking tasks due to per-frame hand detection.

4.3. Hand object interaction

For this task, we evaluate five state-of-the-art methods, see Table 5 and Figure 8. Compared to the other two tasks there is significantly more occlusion, see Figure 4 (top). The approach presented in, *baseline_obj* [54], applies the network directly to this task, resulting in large errors of 46 mm. Methods explicitly handling occlusion achieve lower errors in the range of 25-29 mm: (1) *NAIST_RV_obj* [53] and rvhand_obj [1] segment the hand area from the object using a network. (2) *THU_VCLab_obj* [2] removes the object region from cropped hand images with image processing operations [35]. (3) *slivorezzz_obj* [22] modify their original network to infer the depth values of 2D keypoint locations. We provide a *Best Performance* for this task as the *Baseline* trained on extra data from *FHAD* [9] data set. The training data is from the same two subjects. See the black curve in Figure 8 (left).

The current state-of-the-art methods have difficulty generalizing to the hand-object interaction scenario. But *NAIST_RV_obj* [53] and rvhand_obj [1] have similar performance for visible joints and occluded joints, indicating CNN based segmentation can preserve better structure than image processing operations.

5. Discussions and conclusions

The analysis of 11 state-of-the-art methods (one baseline *Holi CNN* from [54] and the top 10 methods among 17 participating methods in total) from the *HIM2017* [55] challenge provides insights into the current state of 3D hand pose estimation and suggestions for the next generation methods.

(1) 3D volumetric representation used with a 3D CNN shows 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 [41], helps further. Regression-based methods perform better in extreme view point cases [22], 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 can only solve proportional hand shape change, but struggles with per-phalange 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 [40, 150] degrees. However, it is far from being solved for extreme view points, *e.g.* view point range of [0,10], where the hand is facing away from the camera. Multi-stage methods [22] 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) The 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 (*e.g.* along with modeling object) and (b) training the model with large datasets containing hand-object interaction.

References

[1] K. Akiyama, F. Yang, and Y. Wu. Naist rvlab g2's solution for 2017 hand challenge. Hands 2017. 2, 3, 5, 8

[2] X. Chen, G. Wang, H. Guo, and C. Zhang. Pose guided structured region ensemble network for cascaded hand pose estimation. *arXiv preprint arXiv:1708.03416*, 2017. 2, 3, 5, 7, 8

[3] C. Choi, S. Kim, and K. Ramani. Learning hand articulations by hallucinating heat distribution. In *ICCV 2017*. 1

[4] C. Choi, S. H. Yoon, C.-N. Chen, and K. Ramani. Robust hand pose estimation during the interaction with an unknown object. In *ICCV 2017*. 1

[5] N. Cihan Camgoz, S. Hadfield, O. Koller, and R. Bowden. Subunets: End-to-end hand shape and continuous sign language recognition. In *ICCV 2017*. 1

[6] X. Deng, S. Yang, Y. Zhang, P. Tan, L. Chang, and H. Wang. Hand3d: Hand pose estimation using 3d neural network. *arXiv 2017*. 1

[7] A. Erol, G. Bebis, M. Nicolescu, R. D. Boyle, and X. Twombly. Vision-based hand pose estimation: A review. *Computer Vision and Image Understanding*, 108(1):52–73, 2007. 2

[8] M. Everingham, S. A. Eslami, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. *IJCV*, 2015. 2

[9] G. Garcia-Hernando, S. Yuan, S. Baek, and T.-K. Kim. First-person hand action benchmark with rgb-d videos and 3d hand pose annotations. *arXiv preprint arXiv:1704.02463*, 2017. 1, 2, 8

[10] L. Ge, H. Liang, J. Yuan, and D. Thalmann. 3d convolutional neural networks for efficient and robust hand pose estimation from single depth images. In *CVPR 2017*. 2

[11] 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 *CVPR 2016*. 2

[12] L. Ge and J. Yuan. 2017 hand challenge ntu team 2d method. Hands 2017. 2, 3

[13] L. Ge and J. Yuan. 2017 hand challenge ntu team 3d method. Hands 2017. 2, 3

[14] H. Guo, G. Wang, X. Chen, and C. Zhang. Towards good practices for deep 3d hand pose estimation. *arXiv preprint arXiv:1707.07248*, 2017. 2, 3, 4, 5

[15] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR 2016*. 3

[16] S. Honari, P. Molchanov, S. Tyree, P. Vincent, C. Pal, and J. Kautz. Improving landmark localization with semi-supervised learning. *arXiv preprint arXiv:1709.01591*, 2017. 2, 3

[17] S. Honari, J. Yosinski, P. Vincent, and C. Pal. Recombinator networks: Learning coarse-to-fine feature aggregation. In *CVPR 2016*. 2

[18] M. Kristan, J. Matas, A. Leonardis, M. Felsberg, L. Cehovin, G. Fernández, T. Vojir, G. Hager, G. Nebehay, and R. Pflugfelder. The visual object tracking vot2015 challenge results. In *ICCV Workshop 2015*. 2

[19] S. Li and D. Lee. 2017 hand challenge/frame-based/team hcr: Method description. Hands 2017. 2, 3, 5

[20] M. Madadi, S. Escalera, X. Baro, and J. Gonzalez. End-to-end global to local cnn learning for hand pose recovery in depth data. *arXiv preprint arXiv:1705.09606*, 2017. 2, 3, 5

[21] M. Madadi, S. Escalera, A. Carruesco, C. Andujar, X. Baró, and J. Gonzàlez. Occlusion aware hand pose recovery from sequences of depth images. In *FG 2017*. 1

[22] P. Molchanov, J. Kautz, and S. Honari. 2017 hand challenge nvresearch and umontreal team. Hands 2017. 2, 3, 4, 6, 7, 8

[23] G. Moon, J. Chang, and K. M. Lee. 2017 hand challenge snu cvlab team. Hands 2017. 2, 3

[24] MPII Leader Board. `http://human-pose.mpi-inf.mpg.de`. 5

[25] F. Mueller, D. Mehta, O. Sotnychenko, S. Sridhar, D. Casas, and C. Theobalt. Real-time hand tracking under occlusion from an egocentric rgb-d sensor. In *ICCV 2017*. 1

[26] M. Oberweger and V. Lepetit. Deepprior++: Improving fast and accurate 3d hand pose estimation. *ICCV Workshop 2017*. 3, 4

[27] M. Oberweger, G. Riegler, P. Wohlhart, and V. Lepetit. Efficiently creating 3D training data for fine hand pose estimation. In *CVPR, 2016*. 1

[28] M. Oberweger, P. Wohlhart, and V. Lepetit. Hands deep in deep learning for hand pose estimation. In *CVWW 2015*. 2, 3, 4

[29] I. Oikonomidis, N. Kyriazis, and A. A. Argyros. Efficient model-based 3D tracking of hand articulations using kinect. In *BMVC*, 2011. 1, 4, 7

[30] C. Qian, X. Sun, Y. Wei, X. Tang, and J. Sun. Realtime and robust hand tracking from depth. In *CVPR, 2014*. 1

[31] E. Remelli, A. Tkach, A. Tagliasacchi, and M. Pauly. Low-dimensionality calibration through local anisotropic scaling for robust hand model personalization. In *ICCV 2017*. 1

[32] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NIPS 2015*. 7, 8

[33] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 234–241. Springer, 2015. 7

[34] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV 2015*. 2

[35] J. Serra. *Image analysis and mathematical morphology, v. 1*. Academic press, 1982. 8

[36] T. Sharp, C. Keskin, D. Robertson, J. Taylor, J. Shotton, D. Kim, C. Rhemann, I. Leichter, A. Vinnikov, Y. Wei, D. Freedman, P. Kohli, E. Krupka, A. Fitzgibbon, and S. Izadi. Accurate, robust, and flexible real-time hand tracking. In *CHI, 2015*. 1

[37] T. Simon, H. Joo, I. Matthews, and Y. Sheikh. Hand keypoint detection in single images using multiview bootstrapping. In *CVPR 2017*. 1

[38] A. Sinha, C. Choi, and K. Ramani. Deephand: Robust hand pose estimation by completing a matrix imputed with deep features. In *CVPR 2016*. 3

[39] S. Song and J. Xiao. Deep sliding shapes for amodal 3d object detection in rgb-d images. In *CVPR 2016*. 2

[40] S. Sridhar, A. Oulasvirta, and C. Theobalt. Interactive markerless articulated hand motion tracking using rgb and depth data. In *ICCV, 2013*. 1

[41] X. Sun, J. Shang, S. Liang, and Y. Wei. Compositional human pose regression. *ICCV 2017*. 2, 3, 4, 8

[42] X. Sun, Y. Wei, S. Liang, X. Tang, and J. Sun. Cascaded hand pose regression. In *CVPR, 2015*. 1

[43] J. S. Supancic, G. Rogez, Y. Yang, J. Shotton, and D. Ramanan. Depth-based hand pose estimation: methods, data, and challenges. *ICCV, 2015*. 1, 2

[44] D. Tang, H. J. Chang, A. Tejani, and T.-K. Kim. Latent regression forest: Structured estimation of 3D articulated hand posture. In *CVPR 2014*. 1

[45] D. Tang, J. Taylor, P. Kohli, C. Keskin, T.-K. Kim, and J. Shotton. Opening the black box: Hierarchical sampling optimization for estimating human hand pose. In *ICCV, 2015*. 1

[46] D. Tang, T.-H. Yu, and T.-K. Kim. Real-time articulated hand pose estimation using semi-supervised transductive regression forests. In *ICCV*, 2013. 1

[47] J. Taylor, L. Bordeaux, T. Cashman, B. Corish, C. Keskin, E. Soto, D. Sweeney, J. Valentin, B. Luff, A. Topalian, E. Wood, S. Khamis, P. Kohli, T. Sharp, S. Izadi, R. Banks, A. Fitzgibbon, and J. Shotton. Efficient and precise interactive hand tracking through joint, continuous optimization of pose and correspondences. In *ACM SIGGRAPH Conference on Computer Graphics and Interactive Techniques*, 2016. 2

[48] J. Taylor, J. Shotton, T. Sharp, and A. Fitzgibbon. The vitruvian manifold: Inferring dense correspondences for one-shot human pose estimation. In *CVPR 2012*. 4

[49] J. Tompson, M. Stein, Y. Lecun, and K. Perlin. Real-time continuous pose recovery of human hands using convolutional networks. In *TOG 2014*. 1

[50] C. Wan, T. Probst, L. Van Gool, and A. Yao. Crossing nets: Combining gans and vaes with a shared latent space for hand pose estimation. In *CVPR 2017*. 1

[51] C. Wan, A. Yao, and L. Van Gool. Hand pose estimation from local surface normals. In *ECCV, 2016*. 1

[52] Q. Wan. 2017 hand challenge fudan university team. Hands 2017. 3, 4, 5

[53] F. Yang, K. Akiyama, and Y. Wu. Naist rv's solution for 2017 hand challenge. Hands 2017. 2, 3, 5, 7, 8

[54] Q. Ye, S. Yuan, and T.-K. Kim. Spatial attention deep net with partial PSO for hierarchical hybrid hand pose estimation. In *ECCV, 2016*. 1, 2, 3, 8

[55] S. Yuan, Q. Ye, G. Garcia-Hernando, and T.-K. Kim. The 2017 hands in the million challenge on 3d hand pose estimation. In *arXiv:1707.02237 2017*. 1, 2, 8

[56] S. Yuan, Q. Ye, B. Stenger, S. Jain, and T.-K. Kim. Bighand2.2m benchmark: Hand pose dataset and state of the art analysis. In *CVPR 2017*. 1, 2

[57] J. Zhang, J. Jiao, M. Chen, L. Qu, X. Xu, and Q. Yang. 3d hand pose tracking and estimation using stereo matching. *arXiv 2017*. 1

[58] Y. Zhang, C. Xu, and L. Cheng. Learning to search on manifolds for 3d pose estimation of articulated objects. *arXiv 2016*. 1

[59] X. Zhou, Q. Wan, W. Zhang, X. Xue, and Y. Wei. Model-based deep hand pose estimation. In *IJCAI 2016*. 2, 3, 4

[60] C. Zimmermann and T. Brox. Learning to estimate 3d hand pose from single rgb images. In *ICCV 2017*. 1