# Interactive Hand Pose Estimation: Boosting accuracy in localizing extended finger joints

***Cairong Zhang, Guijin Wang, Hengkai Guo, Xinghao Chen, Fei Qiao, Huazhong Yang;***
***Department of Electronic Engineering, Tsinghua University; Beijing, China***

## Abstract

*Accurate 3D hand pose estimation plays an important role in Human Machine Interaction (HMI). In the reality of HMI, joints in fingers stretching out, especially corresponding fingertips, are much more important than other joints. We propose a novel method to refine stretching-out finger joint locations after obtaining rough hand pose estimation. It first detects which fingers are stretching out, then neighbor pixels of certain joint vote for its new location based on random forests.*

*The algorithm is tested on two public datasets : MSRA15 and ICVL. After the refinement stage of stretching-out fingers, errors of predicted HMI finger joint locations are significantly reduced. Mean error of all fingertips reduces around 5mm (relatively more than 20%). Stretching-out fingertip locations are even more precise, which in MSRA15 reduces 10.51mm (relatively 41.4%).*

## Keywords

Random Forest, Hand Pose Estimation, Human Machine Interaction, Fingertip Detection

## 1. Introduction

Hand pose estimation plays an important role in Human Machine Interaction (HMI), such as virtual reality (VR), augmented reality (AR) and remote control. Estimating hand poses from individual depth images has drawn lots of attention from researchers [1, 2, 3, 4, 5, 6, 7], thanks to the availability of depth cameras [8, 9, 10, 11], such as Microsoft Kinect, Intel Realsense Camera etc.

Recently convolutional neural networks [12, 13, 14, 6, 15, 7] and random forests [2, 3, 16, 17, 4, 5] are two mainstreams in hand pose estimation. The performances of these two methods are roughly comparable. But random forests can achieve faster prediction without any GPUs, so it is beneficial to implement online real-time application with only CPUs based on random forests.

Joints in palm such as the wrist and five finger roots have lower degree of freedom (DOF) than those in fingers. Considering kinematic constraints of hand joints, [3] proposed a hierarchical regression method, which first regresses locations of joints with relatively low DOF and then those with higher DOF. He also introduced a cascaded framework to refine joint locations iteratively.

However, due to extremely high fingertip flexibility, similar part confusion, large range of movement and poor depth quality around fingertips, the errors of fingertip locations are improperly high, almost always the highest in all joints. In the reality of HMI, crucial actions such as clicking and zooming depend heavily on stretching-out fingers, especially extended fingertips. Therefore, more accurate estimation of stretching-out finger joints, especially fingertips, is in urgent need.

In this work, we present a novel scheme named Interactive Hand Pose Estimation (IHPE) to address the conflicts between HMI demands and large errors of fingertip locations in hand pose estimation. We first use cascaded hierarchical regression in [3] to get rough locations of hand joints. This module is implemented by reproducing corresponding algorithms in [3] as our baseline, including two kind of stages: palm stages and finger stages. Inspired by fingertip detection and corresponding root localization proposed in [18], we then introduce a refinement stage to re-estimate joint locations of stretching-out fingers. In the refinement stage, we first detect where the stretching-out fingers are, based on *Key Joints Localization* algorithms in our previous work [18]. Then we incorporate the information provided by prediction of our baseline, to distinguish which fingers these detected extended fingers are. Finally, for each joint to be re-estimated, its neighbor foreground pixels (the pixels inside the hand region after segmentation are defined as foreground pixels) vote for its location using random forests. The average result of all votes won by the joint is its new location after refinement. As shown in our experiments, after the refinement stage of stretching-out finger joints, the errors of stretching-out finger joint locations are significantly reduced, especially the fingertips.

Our main contribution is the refinement method proposed to improve estimating locations of extended finger joints, particularly fingertips, after getting some poor results while estimating all joints. The accuracy gap between estimating locations of extended fingertips and other joints can be remarkably reduced after our refinement stage. It benefits the users by providing them more satisfying experience in HMI applications, such as VR and AR.

## 2. Cascaded Hierarchical Regression

In this section, we give a brief introduction of our baseline: Cascaded Hierarchical Regression.

As clarified in [3], joints in different parts of the hand have large variance of flexibilities, so regressing all joints together may cause unnecessarily high complexity and degrade model's performance. We divide the whole regression task into two subtasks, palm stages and finger stages, as Sun. do in [3]. In palm stages, we regress locations of only those joints in palm such as the wrist, finger roots and the palm center. Then we regress joints in five fingers respectively with locations of joints in palm fixed. Besides, we use cascaded architecture in both palm and finger stages like [3] does. Specifically in this paper, we assign three stages to both palm regression and finger regression. Each stage produces coordinate differences between ground truth and predictions of last stage (Eq.(1)). Details of the training algorithm is shown in Table.1. While testing, the output models of the training algo-

**Table.1. Our baseline training procedure**

*Algorithm 1* Training algorithm for our baseline

1. *input:* depth image $I_i$, ground truth pose $\theta_i$, and initial palm pose $\theta_i^{p,0}$ for all training samples $i$
2. *for* $t = 1$ *to* $T_1$ *do* *[palm stages]*
3. $\delta\theta_i^p = \theta_i^p - \theta_i^{p,t-1}$
4. learn random forest $R^{p,t}$ to approximate $\delta\theta_i^p$
5. $\theta_i^{p,t} = \theta_i^{p,t-1} + R^{p,t}(I_i, \theta_i^{p,t-1})$, update palm joint locations
6. initialize finger poses $\theta_i^{f,0}(f = 1,2,3,4,5)$, so that five fingers are in the direction of a vector pointing from wrist (or palm center) to middle finger root
7. *for* $t = 1$ *to* $T_2$ *do* *[finger stages]*
8. *for* $f = 1$ *to* 5 *do*
9. $\delta\theta_i^f = \theta_i^f - \theta_i^{f,t-1}$
10. learn random forest $R^{f,t}$ to approximate $\delta\theta_i^f$
11. $\theta_i^{f,t} = \theta_i^{f,t-1} + R^{f,t}(I_i, \theta_i^{f,t-1})$, update finger joint locations
12. $\theta_i^t = \theta_i^{p,T_1} \bigcup \{\theta^{f,t}\}_{f=1,2,3,4,5}$ update all joints
13. *output:* $\{R^{p,t}\}_{t=1,2,\cdots,T_1}, \{R^{f,t}\}_{t=1,2,\cdots,T_2;f=1,2,\cdots,5}$

rithm are used to perform regression in a similar procedure as training.

$$\delta\theta^t = \theta_{gt} - \theta^{t-1} \tag{1}$$

Note that we don't use *3D Pose Normalization* or *3D Pose Indexed Features* as in [3]. We use traditional pixel difference features instead (Eq.(3)).

## 3. Stretching-Out Finger Joint Refinement

We introduce our refinement of stretching-out finger joint locations in this section. It is divided into two stages. First we detect which fingers are stretching out. Then for each joint in these fingers, its neighbour pixels vote for its location through random forests.

### 3.1 Stretching-out fingers detection

We detect stretching-out fingers based on hand mask images (Fig.1(a)), using the method proposed in our previous work [18]. As clarified in [18], palm center can be defined as the point with maximal distance to the nearest palm boundary point (Fig.1(b)). After the palm center is localized, it scans the palm boundary and get distances between boundary points and palm center. Those points with local maximal distance are stretching-out fingertips, if their distance exceeds a global threshold and the curvature is large enough. Then corresponding finger roots can be localized based on locations and fingertips and the palm center (Fig.1(c)).

Then we obtain rough locations of other joints in these detected stretching-out fingers through interpolation according to finger root and tip locations. To distinguish which these extended fingers are, index, ring or others, we calculate the distances between each detected finger and five fingers predicted from our baseline (Fig.1(d)) respectively. Each detected stretching-out finger is distinguished as the one with the minimum distance to it (Eq.(2)).

$$F = \arg\min_{f=1,2,3,4,5} \sum_{j \in \theta^f} (\|\theta^f_{mask,j} - \theta^f_{baseline,j}\|_2^2) \tag{2}$$

where $\theta^f_{mask,j}$ is location of the $j^{th}$ joint in finger $f$ from stretching-out fingers detection, $\theta^f_{baseline,j}$ is the location of the $j^{th}$ joint in finger $f$ from baseline prediction.

(a) (b) (c) (d)

***Figure 1.*** *Stretching-out fingers detection. (a)Hand mask. (b)Distance transform, the brightest point is palm center. (c)Detected finger tips and roots. (d)Prediction of the baseline.*

### 3.2 Neighbour pixels voting

We introduce our refinement stage in this section. After detecting stretching-out fingers, we aim at re-estimating locations of joints in detected stretching-out fingers, except the finger root which is in palm area. Briefly speaking, for each stretching-out fingers, the joints will get votes from foreground pixels around them. The average of votes each joint got is set to be its location after this refinement stage.

#### 3.2.1 Depth difference feature

As proposed in [19], depth difference features (Eq.(3)) are capable to distinguish background and foreground in depth images. Besides, different parts in hand will have different features. Also due to its mini cost in calculating, depth difference features are commonly used in hand pose estimation from depth images with random forests. We use such features in both cascaded hierarchical regression and neighbor pixels voting.

$$u_i = u + \delta u_i, i = 1,2; \quad depth\ difference\ := I(u_1) - I(u_2) \tag{3}$$

Where u is the reference pixel, while specifically in our neighbor pixels voting, it is the foreground pixel which will vote for a joint's location. $\delta u_i, i = 1,2$ are random offsets within a predefined range. $I(u_i), i = 1,2$ are depths in pixel position $u_i$, which can be read from depth images directly.

#### 3.2.2 Training

Our training algorithm is shown in Table.2.

*Training samples collecting* Each foreground pixel $pixel_{i,j}$ in depth image $I_i$ is extracted as a training sample $\{I_i, pixel_{i,j}\}$. There are usually thousands of foreground pixels in one depth image, so we will extract thousands of training samples in each depth image. Considering the huge memory resume, we randomly select $N$ depth images in training set, ie. not all images are used in training procedure. $N$ has to be large enough, so selected depth images can roughly cover the distribution of hand poses in original training set.

**Table.2. Training procedure of neighbor pixels voting**

| *Algorithm 2* Training algorithm for Neighbor Pixels Voting |
|---|
| 1. *input:* depth image $I_i$, and ground truth pose $\theta_i$ for randomly N samples in training set |
| 2. *training samples:* $\{I_i, pixel_{i,j}\}$, for $i = 1,2,\cdots,N$, $pixel_{i,j}$ is a foreground pixel in $I_i$ |
| 3. $k^* = \arg\min_k \|pixel_{i,j} - joint_{i,k}\|_2$, *find the nearest joint* ($joint_{i,k^*}$) *of* $pixel_{i,j}$ *in* $\theta_i$ |
| 4. $\delta x = joint_{i,k^*} - pixel_{i,j}$ |
| 5. learn random forest $R$ to approximate $\delta x$ |
| 6. *output:* $R$ |

All training samples collected in this way are fed into a random forest for training. We use depth difference feature (See Eq.(3) in Section.3.2.1) for split of tree nodes.

*Prediction of leaf nodes* For each training sample $\{I_i, pixel_{i,j}\}$, we first find the nearest joint of $pixel_{i,j}$ in $\theta_i$ (Eq.(4)). The label of $\{I_i, pixel_{i,j}\}$ is the coordinate difference between $pixel_{i,j}$ and $joint_{i,k^*}$ (Eq.(5)). Then the prediction of a leaf node is set to be the average of all labels of training samples reaching it.

$$k^* = \arg\min_k \|pixel_{i,j} - joint_{i,k}\|_2 \tag{4}$$

$$\delta x = joint_{i,k^*} - pixel_{i,j} \tag{5}$$

Where $joint_{i,k}$ is the position of the $k^{th}$ joint in depth image $I_i$, and $joint_{i,k^*}$ is the position of the nearest joint of $pixel_{i,j}$ in depth image $I_i$.

### *3.2.3 Testing*

After the detection of stretching-out fingers in Section.3.1, we refine predictions of only those joints in stretching-out fingers. Locations of other joints remain as predictions of our baseline in Section.2. Our testing procedure is shown in Table.3.

**Table.3. Testing procedure of neighbor pixels voting**

| *Algorithm 3* Testing algorithm for Neighbor Pixels Voting |
|---|
| 1. *input:* depth image $I_i$ and hand pose $\theta_i^0$ after interpolation for all samples $i$ in testing set, random forest $R$ |
| 2. $dist = \min_k \|pixel_{i,j} - joint_{i,k}\|_2$, $pixel_{i,j}$ is a foreground pixel in $I_i$ |
| 3. *testing samples:* $\{I_i, pixel_{i,j}\}$, if $dist < thres_{dist}$, for all $i$, $thres_{dist}$ is a distance threshold |
| 4. $k^* = \arg\min_k \|pixel_{i,j} - joint_{i,k}\|_2$, *find the nearest joint* ($joint_{i,k^*}$) *of* $pixel_{i,j}$ *in* $\theta_i^0$, for each testing sample |
| 5. $predict_{i,j}^{k^*,v} = R(I_i, pixel_{i,j}^v)$, $pixel_{i,j}^v$ is the v-th foreground pixel voting for joint $k^*$ |
| 6. for each updating joint $k^*$, update its location: $joint_{i,k^*}^{new} = \frac{1}{V}\sum_{v=1}^{V}(pixel_{i,j} + predict_{i,j}^{k^*,v})$ |
| 7. $\theta_i^{refine} = \{joint_{i,k}\}_{not\ updated\ joints\ in\ \theta_i^0} \bigcup \{joint_{i,k^*}\}_{updated\ joints}$, *update hand pose after refinement* |
| 8. *output:* $\theta_i^{refine}$ for all testing images |

The construction of testing sample set is similar as that in training. One difference is that not all foreground pixels are fed into random forest while testing. We first find the nearest joint of a certain foreground pixel $pixel_{i,j}$ in $\theta_i^0$, and calculate corresponding minimum distance. $\theta_i^0$ is the hand pose after interpolation in Section.3.1, locations of joints in stretching-out fingers are constrained within the detected finger contour. If the minimum distance is smaller than a predefined threshold, the foreground pixel is treated as a testing sample and fed into random forest for testing.

Each foreground pixel satisfied the mentioned distance threshold condition votes for the location of its nearest joint through random forest. In each tested depth image, a to-be-updated joint will obtain several votes from different pixels. We set the average of all votes got by a to-be-updated joint as its new location after refinement.

## 4. Experiments And Discussions

In this section, we first introduce the evaluated datasets and metrics in our experiments. Then we compare results of our refinement and the baseline. Finally, we focus on the performance of proposed IHPE on interactive hand joint estimation, especially stretching-out finger tips.

### *4.1 Experiment setup*

We show the parameters we used in neighbor pixels voting in Table.4.

**Table.4. Parameters we used in Neighbor Pixels Voting**

| *parameter* | *value* |
|---|---|
| number of trees | 8 |
| maximum depth of trees | 20 |
| number of features for node split | 200 |
| number of thresholds for feature division | 50 |
| minimum information gain for node split | 1e-6 |
| minimum samples contained in a node | 5 |
| number of depth images for training | 10000 |
| distance threshold in testing (mm) | 10 |

#### *4.1.1 Datasets*

Just like [3], we conducted our experiments on two publicly RGB-D datasets: ICVL hand pose dataset [2] and MSRA hand pose dataset [3].

*MSRA dataset* The MSRA dataset contains 9 subjects with 17 gestures for each subject. 76.5K depth images with 21 annotated joints are collected with Intel's Creative Interactive Camera. Each subject is used for testing in turn while the remain eight subjects for training data. Totally 9 experiments are repeated, and the average result is reported.

*ICVL dataset* The training set of ICVL dataset contains 10 subjects with 26 gestures. About 22K depth images with 16 annotated joints are captured by Intel RealSense with the range of view about 120 degrees. The testing set contains 1.6K images.

#### *4.1.2 Evaluation metrics*

We employ several evaluation metrics following the literatures [2, 12]. For both MSRA dataset and ICVL dataset, the

performance is evaluated by three metrics: (1) *average 3D distance error* is calculated as the average Euclidean distance between ground truth and prediction for each joint (in millimeters). (2) *all fingers error* is computed as the average Euclidean distance for joints in each finger (in millimeters). (3) *all finger tips error* is defined as the average Euclidean distance for finger tip in each finger. For MSRA dataset, in which the depth images of 17 gestures are separated into different directories, we can evaluate the performance of our IHPE by two extra metrics: (4) *stretching-out fingers error* is computed as the average Euclidean distance for joints in each finger (in millimeters), only those fingers stretching out are taken into account. (5) *stretching-out finger tips error* is defined as the average Euclidean distance for finger tip in each finger, considering only those stretching-out fingers.

### 4.2 Comparison with previous work

We compare the experiment result after Stretching-out Finger Joint Refinement in Section.3 with the baseline in Section.2. Note that our baseline is achieved by re-implementing the cascaded framework and hierarchical regression in [3], except that we use the traditional depth difference feature but not *3d pose indexed feature* in [3].

The average 3D distance error on MSRA and ICVL datasets are shown in Fig.2. Mean error of all joints on MSRA is 18.65mm for baseline, and 17.11mm for refinement. Results on ICVL dataset are 15.07mm and 12.98mm respectively. The increasing error of the middle joint in thumb on ICVL is probably caused by the definition of this joint, which is almost dropped in the palm area. We can see significant improvements on the predictions of finger tip locations on both datasets.

Our refinement of stretching-out fingers achieves pleasing performance. This is beneficial from the the pulling force of stretching-out fingers detection and neighbor pixels voting, which drags those joints with large error in the baseline towards more precise areas inside fingers.

### 4.3 Improvements in interactive joint estimation

We show the mean of five fingers (or five finger tips) of *all fingers error* and *all finger tips error* of MSRA dataset and ICVL dataset in Table.5.

No matter *all fingers error* or *all finger tips error*, there is significant improvement on both datasets. We emphasize two conclusions here: (1)All finger tips error reduces 5.14mm on MSRA dataset (relatively 20.3%) and 4.85mm on ICVL dataset (relatively 25.3%). (2)The average error gap after refinement between fingers and fingertips is smaller than the baseline. This means that the conflict between HMI demand and model performance is alleviated. Locations of fingertips can be predicted as precise as other finger joints.

The results of *all fingers error* and *all finger tips error* of MSRA dataset are shown in Fig.3.

**Table.5. Average of *all fingers error* and *all finger tips error***

| dataset | MSRA | | ICVL | |
|---|---|---|---|---|
| method | baseline | refine | baseline | refine |
| all fingers (mm) | 20.18 | 18.02 | 17.00 | 13.65 |
| all fingertips (mm) | 25.26 | 20.12 | 19.15 | 14.30 |

Stretching-out fingers error and stretching-out fingertips error on MSRA are shown in Fig.4. The average result of five fingers is listed in Table.6. Compared with the baseline, the average stretching-out fingers error reduces 4.54mm (relatively 22.9%) after refinement. More significant improvement can be seen in the average stretching-out fingertips error, which reduces 10.51mm (relatively 41.4%). In the reality of HMI, the most important joints are those tips of stretching-out fingers. Our IHPE system achieves more satisfying performance on hand pose estimation.

**Table.6.Average of *stretching-out fingers error* and *stretching-out finger tips error* on MSRA[3] dataset**

| method | baseline | refine |
|---|---|---|
| stretching-out fingers (mm) | 19.81 | 15.27 |
| stretching-out fingertips (mm) | 25.39 | 14.88 |

Fig.5 and Fig.6 show some examples of predictions of the baseline and corresponding one after refinement. The predictions of stretching-out finger joint locations are far more accurate after refinement.

***Figure 5.*** *Example results of MSRA[3]: baseline (top), refinement (bottom). Note that after refinement, the wrist and finger roots remain the same as the baseline.*

***Figure 6.*** *Example results of ICVL[2]: baseline (top), refinement (bottom). Note that after refinement, palm center and finger roots remain the same as the baseline.*

*Run Time* We run the testing procedure of the whole IHPE system in a single thread at $Intel® \ Core^{TM} \ i7-7700K \ CP \ @ \ 4.20GHz \times 8$. The overall speed achieves $18fps$, almost meeting real-time requirements.

## 5. Conclusion

We present a novel scheme named Interactive Hand Pose Estimation (IHPE) to address the conflicts between HMI demands and large error of fingertip locations in hand pose estimation. After obtaining rough locations of hand joints, we detect which fingers are stretching out based on the rough locations and depth images. For each joint in each stretching-out finger, its neighbor foreground pixels vote for its new location through random forests, and its final location is set to the average of voting results. As shown in our experiments, after the refinement stage of stretching-out finger joints, the errors of stretching-out finger joint locations are significantly reduced, especially the fingertips. The performance on ICVL dataset is good enough to perform rough Human Machine Interaction, although it needs to be better for more delicate operations, maybe smaller than 10mm for the error of localizing fingertips. Our future researches focus on further improving the accuracy, include raising the accuracy of detection of stretching-out fingers, and improving the performance of neighbor pixels voting by introducing weighted voting mechanism.

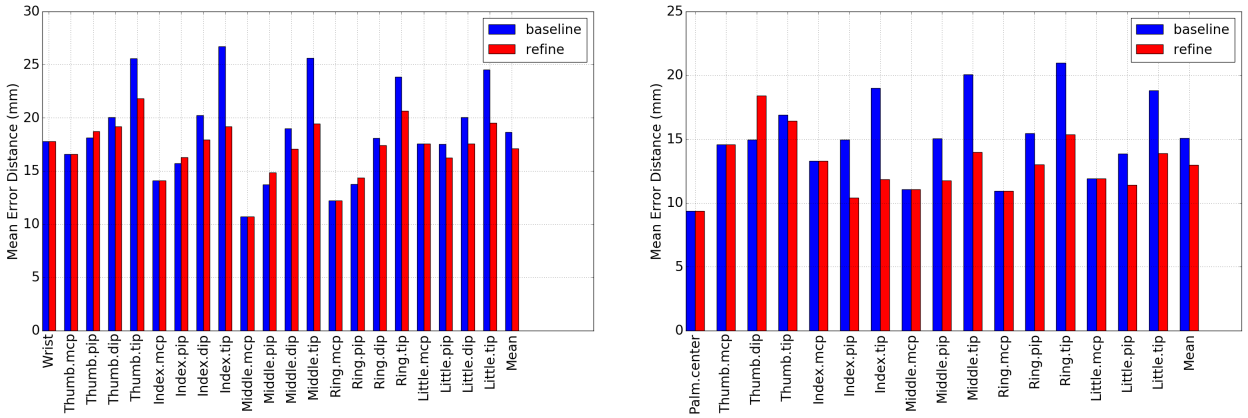


***Figure 2.*** *Distance error of our baseline[3] and refinement on MSRA[3] dataset (left), and ICVL[2] dataset (right)*

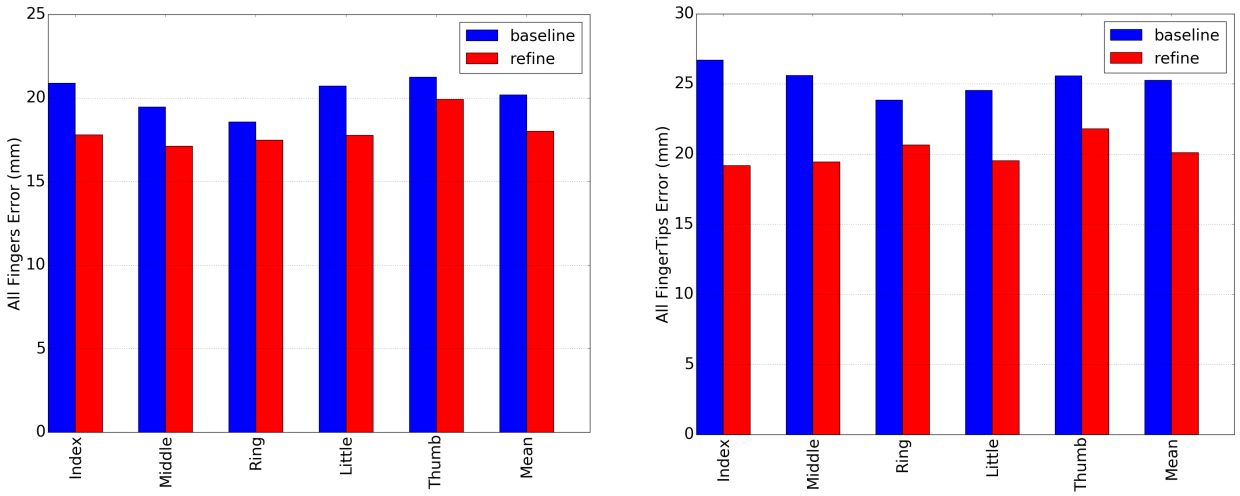


***Figure 3.*** *Improvements in finger joint estimation on MSRA[3] dataset: all fingers error (left) and all finger tips error (right).*

## Acknowledgments

This work was partially supported by State High-Tech Research and Development Program of China (863 Program, No. 2015AA016304).

## References

[1] C. Xu and L. Cheng, *Efficient hand pose estimation from a single depth image*, in *Proceedings of the IEEE International Conference on Computer Vision* (2013), pp. 3456–3462

[2] D. Tang, H. Jin Chang, A. Tejani and T.-K. Kim, *Latent regression forest: Structured estimation of 3d articulated hand posture*, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2014), pp. 3786–3793

[3] X. Sun, Y. Wei, S. Liang, X. Tang and J. Sun, *Cascaded hand pose regression*, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2015), pp. 824–832

[4] J. S. Supancic, G. Rogez, Y. Yang, J. Shotton and D. Ramanan, *Depth-based hand pose estimation: data, methods, and challenges*, in *Proceedings of the IEEE international conference on computer vision* (2015), pp. 1868–1876

[5] C. Wan, A. Yao and L. Van Gool, *Hand Pose Estimation from Local Surface Normals*, in *European Conference on Computer Vision* (Springer, 2016), pp. 554–569

[6] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), pp. 3593–3601

[7] H. Guo, G. Wang, X. Chen, C. Zhang, F. Qiao and H. Yang, *Region ensemble network: Improving convolutional network for hand pose estimation*, in *2017 IEEE International Conference on Image Processing* (2017)

[8] Z. Zhang, "Microsoft kinect sensor and its effect", IEEE multimedia, **vol. 19, 2**:4 (2012)

[9] G. Wang, X. Yin, X. Pei and C. Shi, "Depth estimation for speckle projection system using progressive reliable points growing matching", Applied optics, **vol. 52, 3**:516 (2013)

[10] C. Shi, G. Wang, X. Yin, X. Pei, B. He and X. Lin, "High-accuracy stereo matching based on adaptive ground control points", IEEE Transactions on Image Processing, **vol. 24, 4**:1412 (2015)

[11] L. Keselman, J. I. Woodfill, A. Grunnet-Jepsen and A. Bhowmik, "Intel realsense stereoscopic depth cameras", arXiv preprint arXiv:1705.05548, (2017)

[12] J. Tompson, M. Stein, Y. Lecun and K. Perlin, "Real-time continuous pose recovery of human hands using convolutional networks",

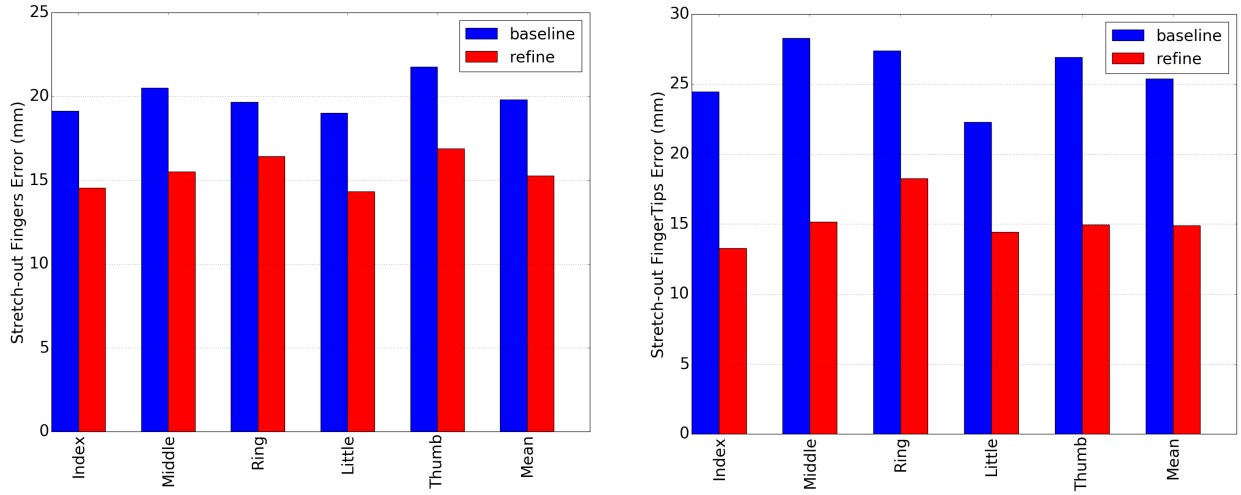


***Figure 4.*** *Improvements in interactive finger joint estimation on MSRA[3] dataset: stretching-out fingers error (left) and stretching-out finger tips error (right).*

ACM Transactions on Graphics (ToG), **vol. 33, 5**:169 (2014)

[13] M. Oberweger, P. Wohlhart and V. Lepetit, "Hands deep in deep learning for hand pose estimation", arXiv preprint arXiv:1502.06807, (2015)

[14] 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* (2015), pp. 3316–3324

[15] A. Sinha, C. Choi and K. Ramani, *Deephand: Robust hand pose estimation by completing a matrix imputed with deep features*, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2016), pp. 4150–4158

[16] 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 *Proceedings of the IEEE International Conference on Computer Vision* (2015), pp. 3325–3333

[17] P. Li, H. Ling, X. Li and C. Liao, *3d hand pose estimation using randomized decision forest with segmentation index points*, in *Proceedings of the IEEE International Conference on Computer Vision* (2015), pp. 819–827

[18] X. Chen, C. Shi and B. Liu, *Static hand gesture recognition based on finger root-center-angle and length weighted Mahalanobis distance*, in *SPIE Photonics Europe* (International Society for Optics and Photonics, 2016), pp. 98970U–98970U

[19] J. Shotton, T. Sharp, A. Kipman, A. Fitzgibbon, M. Finocchio, A. Blake, M. Cook and R. Moore, "Real-time human pose recognition in parts from single depth images", Communications of the ACM, **vol. 56, 1**:116 (2013)

## Author Biography

***Cairong Zhang*** *received his B.S. degree from Department of Electronic Engineering, Tsinghua University, Beijing, China, in 2017, where he is currently working toward his M.S. degree. His research interests include human pose estimation and hand pose estimation.*

***Guijin Wang*** *received his B.S. and Ph.D. degrees (with honor) from Tsinghua University, China, in 1998 and 2003 respectively, all in electronic engineering. From 2003 to 2006, he was a researcher at Sony Information Technologies Laboratories. Since October 2006, he has been with the Department of Electronic Engineering, Tsinghua University, China, as an associate professor. His research interests focus on wireless multimedia, depth sensing, pose recognition, intelligent human-machine UI, intelligent surveillance, industry inspection, and online learning.*

***Hengkai Guo*** *received his B.S. and M.S. degree from Tsinghua University, Beijing, China, in 2014 and 2017 respectively. His research interests include human pose estimation, hand pose estimation and deep learning.*

***Xinghao Chen*** *received his B.S. degree from Department of Electronic Engineering, Tsinghua University, Beijing, China, in 2013, where he is currently pursuing the Ph.D. degree. From Sept. 2016 - Jan. 2017, he was a visiting Ph.D. student with Imperial College London, UK. His research interests include deep learning, hand pose estimation and gesture recognition.*

***Fei Qiao*** *received his B.S. degree from Lanzhou University, China, in 2000, and his Ph.D. degree from Tsinghua University, Beijing, China, in 2006, both in electronic engineering. He is now an associate professor in Department of Electronic Engineering in Tsinghua University. His research interests include Smart Camera Architectures for Depth Perception Applications, X Computing Architectures and Heterogeneous Integration of Electronic-Eyes.*

***Huazhong Yang*** *received his B.S. and Ph.D. degree from Department of Electronic Engineering, Tsinghua University, Beijing, China, in 1993 and 1998 respectively. Since July 1998, he has been with the Department of Electronic Engineering, Tsinghua University, China, as a professor. His research interests focus on new structure, synthesis and verification of micro-system chip, and design of analog and mixed-signal system.*