CASCADED POINT NETWORK FOR 3D HAND POSE ESTIMATION*

Yikun Dou[1,2], *Xuguang Wang*[1], *Yuying Zhu*[2], *Xiaoming Deng*[2] †, *Cuixia Ma*[2,3,5], *Liang Chang*[4], *Hongan Wang*[2]

[1] Department of Automation, North China Electric Power University
[2] Beijing Key Lab. of HCI, [3] State Key Lab. of CS, Institute of Software, Chinese Academy of Sciences
[4] College of Information Science and Technology, Beijing Normal University [5] UCAS

ABSTRACT

Recent PointNet-family hand pose methods have the advantages of high pose estimation performance and small model size, and it is a key problem to get effective sample points for PointNet-family methods. In this paper, we propose a two-stage coarse to fine hand pose estimation method, which belongs to PointNet-family methods and explores a new sample point strategy. In the first stage, we use 3D coordinate and surface normal of normalized point cloud as input to regress coarse hand joints. In the second stage, we use the hand joints in the first stage as the initial sample points to refine the hand joints. Experiments on widely used datasets demonstrate that using joints as sample points is more effective and our method achieves top-rank performance.

Index Terms— Hand pose estimation, point cloud CNN

1. INTRODUCTION

Accurate hand pose estimation is of great importance in human-computer interactions and augmented reality [1], especially applications using hand gestures as input such as Oculus Rift, Microsoft Hololense and HTC Vive etc. Recently, hand pose estimation from depth has progressed rapidly, while it is still a challenging task to recover accurate and real-time 3D hand pose due to the self-occlusion and high degree-of-freedom of hand articulations.

Previous convolutional neural network (CNN) based hand pose estimation methods using depth images can be classified into three types according to the input modality of CNNs. 1) 2D CNN based methods [2, 3, 4, 5, 6, 7]. The methods extract features from depth using 2D CNN, and estimate hand pose using regression. As observed in [8], due to the domain difference between 2D depth map and 3D joint, it is difficult to train an effective CNN to learn the mapping between depth and 3D joints. 2) 3D CNN based methods [9, 10, 11]. These methods encode depth as volumetric features, feed the volumtric features to 3D CNN to get hand skeleton joints using regression [9, 10] or classification [11]. These methods can encode effective volumetric features, which are useful for hand pose estimation, and can get high quality hand pose performance, while they have larger number of parameters compared with the methods using depth as input and may not be suitable for devices with small display memories such as mobile devices. 3) Point cloud based methods [12, 13, 14]. These methods encode a depth image as point clouds, and feed point clouds into a point cloud network to get the hand skeleton joints. These methods are usually built upon PointNet [15] or PointNet++ [16], which have the advantage of small number of network weights. These methods use multiple perceptron machine (MLP) to abstract point information, which loses part of 3D context information of point clouds, thus the performance of hand pose estimation is still not satisfactory. We find that the performance of these methods heavily relies on the sample point of the first layer of network. Thus, it is a key problem to select effective sample points that can enhance the performance of hand pose estimation.

In this work, we propose a two-stage PointNet framework for hand pose estimation. We convert depth into point cloud, and then point cloud is downsampled and normalized following the standard processing pipelines in [16, 12, 14]. In the first stage of our network, we use 3D coordinate and surface normal of normalized point cloud as input to regress coarse hand joints. In the second stage, we use the hand joints in the first stage and point cloud as input to further refine the hand joints. The main difference of the two stages is the initial sample points. The first stage uses random sample points, and the second stage uses rough joints as sample point. Using hand joints as sample points extracts the pose-index features (similar to [17]) and benefits the performance of hand pose estimation. Experiments show that our two-stage network can consistently improve the performance of PointNet-based hand pose estimations [14], and get very competitive performance in two widely-used datasets. Moreover, our method has advantages of small network parameters, and can inspire hand gesture applications on mobile devices etc.

Our main contributions are summarized as follows:

1. We propose a coarse to fine network to regress hand joint from point cloud. In the fine joint regression, we select hand joints and additional points as sample point, which encodes pose-index features and benefit the hand pose estimation task. As a comparison, the coarse joint regression use randomly sampled point as same as PointNet-family hand pose works.
2. Our work achieves top-ranked performance on two widely used hand pose dataset, and the model weight is small. Our work can inspire the related research in this field.

2. RELATED WORK

CNN-based Hand Pose Estimation. Hand pose estimation from depth image can be classified three categories: discriminate methods, generative methods and hybrid methods. The discriminate methods [3, 4, 5, 6, 7, 12, 13, 14] set network structure and optimize network parameter by training data to estimate hand pose .The generative methods [18, 19, 20] estimate hand pose by fitting a hand

*THIS WORK IS SUPPORTED BY NATIONAL KEY R&D PROGRAM OF CHINA UNDER GRANT 2016YFB1001201, NATIONAL NATURAL SCIENCE FOUNDATION OF CHINA GRANT NO. 61473276 AND 61402040, NATURAL SCIENCE FOUNDATION OF BEIJING, CHINA, GRANT NO. L182052, AND THE FUNDAMENTAL RESEARCH FUNDS FOR THE CENTRAL UNIVERSITIES NO. 2017MS128. THIS WORK IS CONDUCTED WHEN Y. DOU IS AN INTERN AT ISCAS.

†Corresponding author. Email:xiaoming@iscas.ac.cn

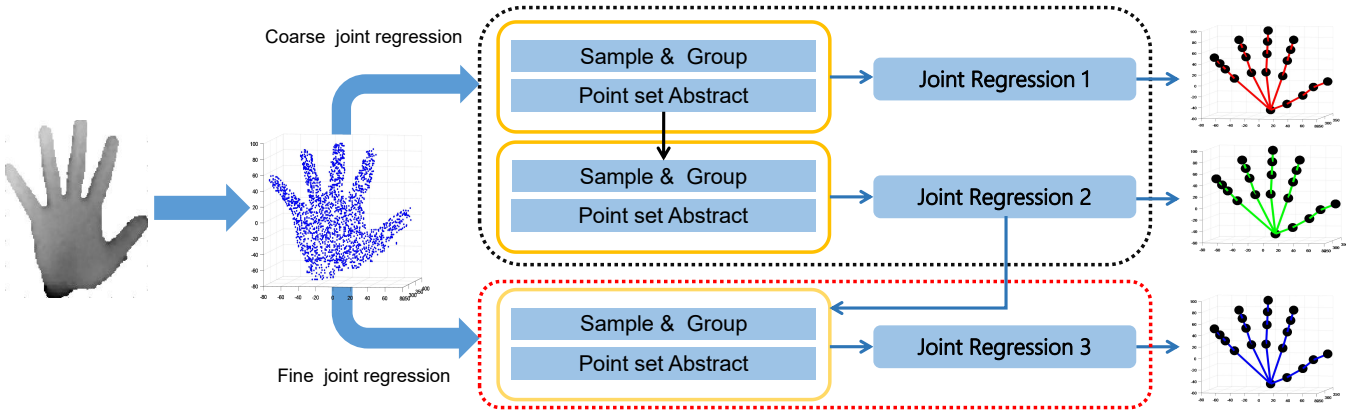


Fig. 1. Overview of our two-stage network. We feed point cloud and surface normal to the first network to get coarse hand joints, then feed coase hand joints and point cloud to the second network to refine hand joint locations.

model. The hybrid methods [4, 21, 6, 7, 22], estimate hand pose by a generative hand model from training data and a model-based optimization.

Recent work on hand pose estimation has shift to deep learning. There are two pose estimation solutions from depth image: 1) Estimate hand joints via classified from each joints' heat map, which is probability map of the joint belong to each pixel [4, 5]. Tompson *et al.*[4] estimate hand joints only by single-view 2D heat-map which lack 3D information, to overcome this weakness, Ge *et al.*[5] project the depth image onto multiple views and estimate 3D hand pose from multi-view heat-maps. 2) Predict 3D coordinates of joints via regression [3, 23]. Zhou *et al.*[23] first integrate a kinematic layer into CNN regression model for hand pose estimation, which ensures the geometric validity of hand pose. Oberweger *et al.*[3] use principle component analysis (PCA) to reduce the dimension of hand pose parameters, use a network to regress the low-dimensional parameters, and reconstruct the hand pose by multiplying the low-dimensional parameters with PCA basis. Oberweger *et al.*[6] also propose a feedback loop to refine hand joints iteratively. For leveraging more spatial information from depth image, 3D-CNN are also applied for hand joints estimation [9, 10, 11]. Deng *et al.*[9] propose a Hand3D methods to regress hand joints from TSDF volume, at same time this methods use a 3D FCN to refine the TSDF which completes missing depth. Moon *et al.*[11] propose the V2V-PoseNet to estimate hand joints by 3D heat map via classification.

Point Cloud CNNs and Applications in Hand Pose Estimation. Point cloud is a subset of points from an Euclidean space. It has three main properties: disordered, interaction among points and invariance under transformation. Qi *et al.*propose PointNet [15] and PointNet++ [16], two seminal CNN network architectures on point cloud. PointNet and PointNet++ both use multiple perceptron machine (MLP) to predict on every isolated point feature , and extract representative features from the whole point cloud using max-pooing. Wang *et al.*[24] propose a novel operation for point cloud, Edge-Conv, to capture more effective local geometric features than MLP. Klokov *et al.*[25] do not process directly on unordered point clouds, and they construct a point cloud of a certain order structure by kd-tree and learn feature on this structured point cloud. Xu *et al.*[26] propose a new convolution unit, namely SpiderConv, which extends convolutional operations from regular grids to irregular point sets to extract features from point set. Inspired by Qi *et al.*[15, 16], several recent work use PointNet-family networks to estimate hand pose. Ge *et al.*[12] estimate hand pose on raw point cloud by a hierarchy network. Ge *et al.*[14] propose a hand pose estimation method using 3D joint heat-maps, in which point-wise estimations are used to estimate 3D joint locations with weighted fusion. Chen *et al.*[13] predict hand part segmentation first, and then use per-point 3D coordinate and part segmentation to estimate hand joints.

Our method differs from previous methods, especially recent PointNet-family hand pose estimation methods [12, 14] as follows: we use a cascaded network, which uses the output of the first stage to refine the second stage output. In the second stage pose regression, we select hand joints and additional points as sample points, which is more effective than randomly sampled points in previous PointNet-based methods.

3. METHOD

We aim to regress hand joints from point cloud. The pipeline of our network is shown in Fig. 1. First, we transform a depth image into point cloud, and normalize the scale and principal direction. Then we feed the normalized point cloud into two-stage coarse-to-fine network to recover the hand pose. The coarse joint regression gets initial joint positions, and the fine joint regression uses the initial joints and point cloud as input and refine the joint estimation.

3.1. Data Processing

In the data processing, we first convert a hand depth image into point cloud. The whole 3D point cloud is downsampled to N points to reduce data redundancy and improve computation efficiency. In order to handle scale diversity, we normalize point cloud in an oriented bounding box. We scale the point cloud scale between $[-1, 1]$ in each axis. In order to handle the large variation of palm orientations, we conduct 3D derotation to transform the point cloud to a canonical direction. We use principal component analysis (PCA) to find two principal directions $[d_x, d_y]$ from 3D hand point cloud. Then use the product of the direction vectors of the two principal directions to find the third orthogonal direction $d_z = d_x \times d_y$. The point cloud $\mathbf{P}_{norm}$ is obtained by rotating the scaled point cloud $\mathbf{X}$ using the rotation transformation matrix $\mathbf{R} = [d_z, d_x, d_y]^T$. The output $\mathbf{P}_{norm}$ is invariant to scale and orientation of the input point cloud, and helps to improve the robustness against diversity inputs. In this work, the input of our network is $\mathbf{X}_{norm} = [\mathbf{P}_{norm}, \mathbf{N}_{norm}]$ [14],

which is a set of normalized point $\mathbf{P}_{norm}$ and the surface normal vector $\mathbf{N}_{norm}$ at each point.

Although we aim to regress hand joints $\mathbf{J}$ from network, we find that the degree of freedom of human hand is less than hand joints parameters. Inspired by DeepPrior [3] and HandPointNet [12], we learn low dimensional subspace of hand pose with principal component analysis (PCA), which can enforce the validity of hand pose better. We learn a principle components $\mathbf{V} = [\mathbf{v}_1, \mathbf{v}_2, ..., \mathbf{v}_m]^T (m = 42)$ and data mean $\mathbf{m}$ from the ground truth hand joints. The ground truth low dimensional feature $\mathbf{w}$ could be calculated by

$$\mathbf{w} = \mathbf{V}^T(\mathbf{J} - \mathbf{m}) \tag{1}$$

After we learn a network to estimate $\hat{\mathbf{w}}$ from point cloud, the hand pose can be reconstructed by back-projection into the joint space as follows

$$\hat{\mathbf{J}} = \mathbf{V}\hat{\mathbf{w}} + \mathbf{m} \tag{2}$$

3.2. Network Details

We adopt a two-stage framework to estimate hand pose. The coarse joint regression network uses two Point Sample & Group + Pointset Abstract units [16], and the fine joint regression network adopts one Point Sample & Group + Pointset Abstract unit. Each Point Sample & Group + Pointset Abstract unit is followed with fully connected layers to predict hand joint positions.

Point Sampling + Grouping. Point sampling [16] selects a subset of point from the input, which can represent the local structure in point cloud, and grouping [16] constructs local point sets using neighbor points of the output of point sampling. We use farthest point sampling (FPS) [16] for point sampling, and use K-nearest neighbor clustering (KNN) for grouping.

FPS randomly selects a point in the point cloud, and then extracts a point as the starting point from the remaining point cloud, which is the farthest from the starting point. Then, add the sampling point to the sampled point list and continue to iteratively sample the point, which is farthest from the sampled point list. In the coarse joint regression, we use two point sampling layers, the first sampling downsamples from the input 1024 input points to 512 points, and the second sampling downsamples again to 128 points. After each point sampling, we use KNN clustering (in our experiment, $K = 64$) on the sampling points. In order to ensure the clustering density, the clustering is also enforced with a radius limit.

Coarse Regression. After each clustering, the point cloud will be fed to a multi-layer perceptron model (MLP) in the network to increase the features channels of each point, and then extract the point cloud global feature by feeding each cluster into the max-pooling layer. Then the generated features and sample points are concatenated and fed to the next clustering layer and regress the hand joints through a fully connected layer.

When we conduct a complicated regression problem, it is necessary to make the network deeper for extracting effective features. However, it is quite difficult to supervise entire network. Inspired by [27], we add one more intermediate hand pose supervision into the coarse joint regression network, which is helpful to improve the performance (refer to Table 1).

Joint Clustering and Hand Pose Regression. The joint prediction using coarse regression is only rough estimation, but the rough joints can guide us to get contain pose-indexed feature (similar to [17]) to refine the joint estimation. We utilize rough joints as part of the sample points to cluster neighboring point (refer to yellow regions in Fig. 2), feed the point groups into joint regression net, and then get the final hand joint predictions. At clustering step, we use additional

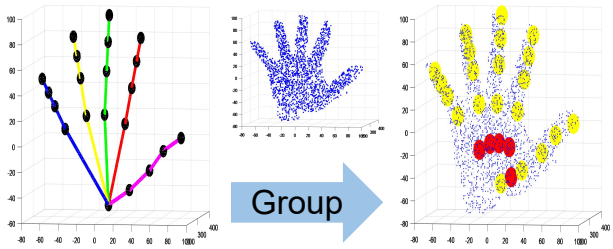


Fig. 2. **Group with joint**: Use the joint estimation in the first stage as sample point to group point in the hand point cloud. Since there is only one joint on palm, we also add five points between root joints of five finger and wrist point to the sample points.

five sample points between finger root joint and palm joint (refer to the red region in Fig. 2) to enforce the spatial structure of the sampled points. After that, we group $K = 64$ nearest points of the sample points (joints + additional five points), which the same strategy of coarse net. In the fine joint regression network, we have much sparse sample points than in the coarse regression network. In order to maintain the 3D spatial structure of the point cloud well, we do not cluster radius in Grouping and search K-nearest neighbor points. After Grouping base on joints, we use point set abstraction layer to increase per-point feature channel, and recover hand joints using joint regression layer.

Loss Function. We have two joint regression loss functions L_{j_1} and L_{j_2} in the coarse joint regression network, and one joint regression loss function L_{j_3} in the fine joint regression network. The loss functions L_{j_1}, L_{j_1} and L_{j_3} are all defined using the Euclidean distance between the estimated low-dimensional joint feature $\hat{\mathbf{w}}$ and the ground truth $\mathbf{w}$. The total loss function L is defined as follows

$$L = \alpha L_{j_1} + \beta L_{j_2} + \gamma L_{j_3} \tag{3}$$

where α, β, γ are the weights of the loss functions.

3.3. Implementation Detail

We implement the experiment on a workstation with two Intel Xeon E5-2667v4 CPU, 256GB of RAM and an NVIDIA P100 GPU. The deep neural network base on Pytorch framework. When training the deep neural network, we use ADAM optimizer with initial learning rate 0.001, batch size 32. The learning rate is divided by 10 for every 10 epochs. The training is stop after 30 epochs to prevent over-fitting.

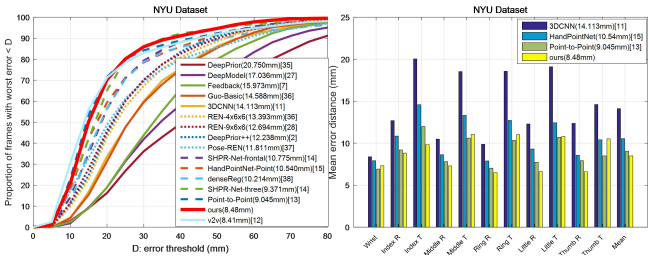
4. EXPERIMENT

We evaluate our method on two-widely hand pose datasets: NYU hand dataset [4] and MSRA hand dataset [17].

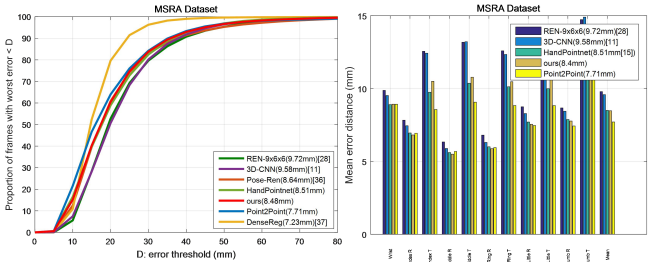
4.1. Dataset

NYU Hand Dataset [4] contains three views of hand pose, each view contains 72,757 frames training data and contains 8,252 testing frames. The dataset is annotated with 36 ground truth hand joints, but we evaluate the performance using 21 hand joints as in [4].

MSRA Hand Dataset [17] contains 9 subjects each subject with 17 hand gestures, each hand gestures contain 500 frames, the datsaet contains annotations of 21 hand skeleton joints. We adopt standard subject-independent nine-fold cross validation, in which we choose eight subjects as training data and test on the remaining one subject.



(a)



(b)

Fig. 3. Comparison with state-of-the-art methods on NYU (top) and MSRA (bottom) hand dataset.

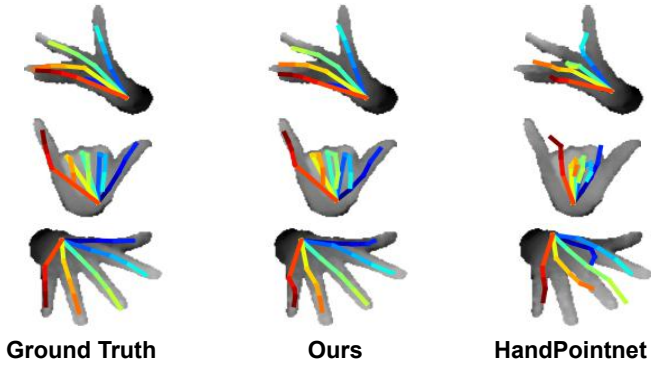


Fig. 4. Quantitative comparison on MSRA hand dataset.

4.2. Evaluation Metrics

We use two standard metrics to evaluate the performance of 3D hand pose estimation method: 1) Per-joint mean error distance over all test frames and overall mean error distance for all joint on all test frames; 2) The proportion of good frames in which the worst joint error is below a threshold [28].

4.3. Comparisons with State-of-the-art Methods

NYU Hand Dataset. We compare our methods with 14 state-of-the-art methods: DeepPrior [29], DeepModel [23], FeedBack [6], REN9x6x6 [24], and GUO-Basic [30], Pose-REN [31], DenseReg [32], 3D-CNN [10], DeepPrior++ [3], SHPR [13], HandPointnet [14], and point-to-point [12]. In the comparison, SHPR, HandPointnet and point-to-point are the state-of-the-art PointNet-based hand pose estimation methods, 3D CNN [10] is 3D CNN method using volume as input, and the other methods are all 2D CNN based methods. Fig. 3 (a) shows the proportion of good frames over different error thresholds and the pre-joint mean error distance of different methods on NYU dataset.

MSRA Hand Dataset. We compare our methods with 4 state-of-the-art methods. Fig. 3 (b) shows the proportion of good frames over different error thresholds and the pre-joint mean error distance of different methods on MSRA dataset.

Table 1. The impact of refinement and loss weights on NYU dataset.

Method	α	β	γ	Mean error
w/o refine	0	1	/	10.5mm
with refine	0	1	1	9.55mm
with refine	1	1	1	9.73mm
with refine	0.33	1	2	9.47mm
with refine	0.5	1	1	9.42mm
with refine+3view	0.5	1	1	8.48mm

Experiment Analysis. As shown in Fig. 3, our method outperforms almost all the methods on NYU and MSRA datasets. In NYU dataset, V2V [11] is only 0.07 mm better than our method, but the model size of V2V (457MB) is more than 13 times of our model (34MB), FeatureMapping [33] is about 1 mm better than our method, but it uses additional 500K synthetic dataset than our method and it further requires pairs of ground truth and synthetic depth images for training, which is not available in most dataset except NYU data. Thus, FeatureMapping [33] can not provide the performance on the other hand datasets. Our method is 1.2mm, 0.7mm, 0.07mm worse than the top-3 methods (DenseReg [32], Point2Point [12] and Handpointnet [14]) on MSRA dataset, while our method is 1.8mm, 0.5mm, 2.1mm better than them on NYU dataset. Fig. 4 shows the comparison with HandPointNet [14], and we can observe that our method performs better. In summary, our method can achieve almost the state-the-art performance in both dataset using small model size, which can benefit many applications.

Timing. Our methods runs in real-time at 70+fps. The average running time per frame is 14.3 ms, 10.1 ms for depth to point cloud conversion, point sampling and surface normal calculation, 4.2 ms for hand pose prediction.

4.4. Self Comparison

The Effect of Cascade Refinement. As shown in Table 1, cascade refinement reduces the mean error by 0.95 mm (10.5 mm vs. 9.55 mm), which confirms that our coarse to fine strategy is useful.

The Effect of Loss Weights. Loss weight is important for multi-task CNNs. If we set $\beta = \gamma = 1$ and select $\alpha = 0, 0.5, 1$, we find the mean joint error varies as 9.55, 9.42 and 9.73. Thus, we select $\alpha = 0.5, \beta = \gamma = 1$ as loss weights.

The Effect of Data Augmentation. For NYU dataset, we have depth images from three views. If we use the training data from three views, the mean joint error of our method reduces 1 mm (9.42 mm vs. 8.48 mm). Therefore, data augmentation with all the data from three views is helpful.

5. CONCLUSION

In this paper, we propose a two-stage coarse to fine hand pose estimation method, which belongs to PointNet family methods. We first use hand pointnet to get initial hand pose, and then use the initial hand joints as sample points for the fine joint regression stage. Experiments on NYU and MSRA hand datasets show that our method can achieve very competitive hand pose performance and run in real-time. Our work can inspire related researches such as augmented reality using hand gesture as input etc.

6. REFERENCES

[1] Shihong Xia and Zhaoqi Wang, "Recent advances on virtual human synthesis," *Science in China Series F: Information Sciences*, vol. 52, no. 5, pp. 741–757, 2009.

[2] Shanxin Yuan, Guillermo Garcia-Hernando, Björn Stenger, Gyeongsik Moon, Ju Yong Chang, Kyoung Mu Lee, Pavlo Molchanov, Jan Kautz, Sina Honari, Liuhao Ge, et al., "Depth-based 3d hand pose estimation: From current achievements to future goals," in *CVPR*, 2018.

[3] Markus Oberweger and Vincent Lepetit, "Deepprior++: Improving fast and accurate 3d hand pose estimation," in *ICCV workshop*, 2017.

[4] Jonathan Tompson, Murphy Stein, Yann Lecun, and Ken Perlin, "Real-time continuous pose recovery of human hands using convolutional networks," *ACM TOG*, vol. 33, no. 5, pp. 169, 2014.

[5] Liuhao Ge, Hui Liang, Junsong Yuan, and Daniel Thalmann, "Robust 3d hand pose estimation in single depth images: from single-view cnn to multi-view cnns," in *CVPR*, 2016.

[6] Markus Oberweger, Paul Wohlhart, and Vincent Lepetit, "Training a feedback loop for hand pose estimation," in *CVPR*.

[7] Chengde Wan, Thomas Probst, Luc Van Gool, and Angela Yao, "Crossing nets: Combining gans and vaes with a shared latent space for hand pose estimation," in *CVPR*, 2017.

[8] James Steven Supancic III, Gregory Rogez, Yi Yang, Jamie Shotton, and Deva Ramanan, "Depth-based hand pose estimation: methods, data, and challenges," *arXiv preprint arXiv:1504.06378*, 2015.

[9] Xiaoming Deng, Shuo Yang, Yinda Zhang, Ping Tan, Liang Chang, and Hongan Wang, "Hand3d: Hand pose estimation using 3d neural network," *arXiv preprint arXiv:1704.02224*, 2017.

[10] Liuhao Ge, Hui Liang, Junsong Yuan, and Daniel Thalmann, "3d convolutional neural networks for efficient and robust hand pose estimation from single depth images," in *CVPR*, 2017.

[11] Gyeongsik Moon, Ju Yong Chang, and Kyoung Mu Lee, "V2v-posenet: Voxel-to-voxel prediction network for accurate 3d hand and human pose estimation from a single depth map," in *CVPR*, 2018.

[12] Liuhao Ge, Zhou Ren, and Junsong Yuan, "Point-to-point regression pointnet for 3d hand pose estimation," *ECCV*, 2018.

[13] Xinghao Chen, Guijin Wang, Cairong Zhang, Tae-Kyun Kim, and Xiangyang Ji, "Shpr-net: Deep semantic hand pose regression from point clouds," *IEEE Access*, vol. 6, pp. 43425–43439, 2018.

[14] Liuhao Ge, Yujun Cai, Junwu Weng, and Junsong Yuan, "Hand pointnet: 3d hand pose estimation using point sets," in *CVPR*, 2018.

[15] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," *CVPR*, 2017.

[16] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," in *NIPS*, 2017, pp. 5099–5108.

[17] Xiao Sun, Yichen Wei, Shuang Liang, Xiaoou Tang, and Jian Sun, "Cascaded hand pose regression," in *CVPR*, 2015.

[18] Luca Ballan, Aparna Taneja, Jürgen Gall, Luc Van Gool, and Marc Pollefeys, "Motion capture of hands in action using discriminative salient points," in *ECCV*, 2012.

[19] Dimitrios Tzionas, Luca Ballan, Abhilash Srikantha, Pablo Aponte, Marc Pollefeys, and Juergen Gall, "Capturing hands in action using discriminative salient points and physics simulation," *IJCV*, vol. 118, pp. 172–193, 2016.

[20] Anastasia Tkach, Andrea Tagliasacchi, Edoardo Remelli, Mark Pauly, and Andrew Fitzgibbon, "Online generative model personalization for hand tracking," *ACM TOG*, vol. 36, no. 6, pp. 243, 2017.

[21] Jonathan Taylor, Lucas Bordeaux, Thomas Cashman, Bob Corish, Cem Keskin, Toby Sharp, Eduardo Soto, David Sweeney, Julien Valentin, Benjamin Luff, et al., "Efficient and precise interactive hand tracking through joint, continuous optimization of pose and correspondences," *ACM TOG*, vol. 35, no. 4, pp. 143, 2016.

[22] Qi Ye, Shanxin Yuan, and Tae-Kyun Kim, "Spatial attention deep net with partial pso for hierarchical hybrid hand pose estimation," in *ECCV*, 2016.

[23] Xingyi Zhou, Qingfu Wan, Wei Zhang, Xiangyang Xue, and Yichen Wei, "Model-based deep hand pose estimation," in *IJCAI*, 2016.

[24] Yue Wang, Yongbin Sun, Ziwei Liu, Sanjay E Sarma, Michael M Bronstein, and Justin M Solomon, "Dynamic graph cnn for learning on point clouds," *arXiv preprint arXiv:1801.07829*, 2018.

[25] Roman Klokov and Victor Lempitsky, "Escape from cells: Deep kd-networks for the recognition of 3d point cloud models," in *ICCV*, 2017.

[26] Yifan Xu, Tianqi Fan, Mingye Xu, Long Zeng, and Yu Qiao, "Spidercnn: Deep learning on point sets with parameterized convolutional filters," in *ECCV*, 2018.

[27] Alejandro Newell, Kaiyu Yang, and Jia Deng, "Stacked hourglass networks for human pose estimation," in *ECCV*, 2016.

[28] Jonathan Taylor, Jamie Shotton, Toby Sharp, and Andrew Fitzgibbon, "The vitruvian manifold: Inferring dense correspondences for one-shot human pose estimation," in *CVPR*, 2012.

[29] Markus Oberweger, Paul Wohlhart, and Vincent Lepetit, "Hands deep in deep learning for hand pose estimation," 2015.

[30] Hengkai Guo, Guijin Wang, Xinghao Chen, Cairong Zhang, Fei Qiao, and Huazhong Yang, "Region ensemble network: Improving convolutional network for hand pose estimation," in *ICIP*.

[31] Xinghao Chen, Guijin Wang, Hengkai Guo, and Cairong Zhang, "Pose guided structured region ensemble network for cascaded hand pose estimation," *Neurocomputing*, 2018.

[32] Chengde Wan, Thomas Probst, Luc Van Gool, and Angela Yao, "Dense 3d regression for hand pose estimation," in *CVPR*.

[33] Mahdi Rad, Markus Oberweger, and Vincent Lepetit, "Feature mapping for learning fast and accurate 3d pose inference from synthetic images," in *CVPR*, June 2018.