SRN: Stacked Regression Network for Real-time 3D Hand Pose Estimation

Pengfei Ren
rpf@bupt.edu.cn
Haifeng Sun
sunhaifeng_1@ebupt.com
Qi Qi
qiqi@ebupt.com
Jingyu Wang
wangjingyu@bupt.edu.cn
Weiting Huang
huangweiting@ebupt.com

State Key Laboratory of Networking and Switching Technology,
Beijing University of Posts and Telecommunications,
Beijing, China

Abstract

Recently, most of state-of-the-art methods are based on 3D input data, because 3D data capture more spatial information than the depth image. However, these methods either require a complex network structure or time-consuming data preprocessing and post-processing. We present a simple and accurate method for 3D hand pose estimation from a 2D depth image. This is achieved by a differentiable re-parameterization module, which constructs 3D heatmaps and unit vector fields from joint coordinates directly. Taking the spatial-aware representations as intermediate features, we can easily stack multiple regression modules to capture spatial structures of depth data for accurate and robust estimation. Furthermore, we explore multiple good practices to improve the performance of the 2D CNN for 3D hand pose estimation. Experiments on four challenging hand pose datasets show that our proposed method outperforms all state-of-the-art methods with faster inference speed.

1 Introduction

3D hand pose estimation plays an important role in applications of human-computer interaction and virtual reality. With the development of deep learning and deep camera, it has made significant progress in recent years [9, 15, 29, 31, 35]. Nevertheless, due to large variations in hand orientations, high similarity among fingers, severe self-occlusion and poor quality of depth images, 3D hand pose estimation still suffers from the issues of estimation accuracy and algorithm efficiency.

Recently, deep neural network based approaches have achieved drastic performance improvement in 3D hand pose estimation. Most of these methods [9, 12, 14, 15, 31, 39] directly regress 3D pose parameters such as joint angles, 3D joint locations or heat maps from depth images with 2D CNNs. By treating the depth map as a 2D image, these methods under-utilize the 3D spatial information of the depth map, thus having relatively low precision.

A key question for improving the accuracy of estimation is how to make better use of the spatial structure information in depth data [38]. One straightforward solution is converting the depth image to 3D data, such as voxels and points, and then applying a 3D deep learning method. 3D data based methods have shown state-of-the-art performance in terms of accuracy, but they come at the cost of memory inefficiency [13] or time-consuming preprocessing as well as post-processing [2, 7, 8]. An alternative way is to incorporate spatial-aware representations into 2D CNNs. Wan et al. [33] propose a dense pixel-wise estimation method, which is based on a 2D CNN but can leverage 3D geometric properties of depth input. Although this method is free from tedious preprocessing, it still suffers from some limitations. First, this method adopts an inefficient decoding network to perform pixel-wise estimations, which increase the number of network parameters and inference time. Second, generating estimations for background pixels of the 2D depth image may distract the neural network from learning effective features in the hand region. Furthermore, inferring the joint coordinates from the pixel-wise estimation, which is unreliable when the depth data near the target joint points are missing, because the inference process relies heavily on local voting.

In order to capture the spatial structure of depth map efficiently, in this paper, we propose a differentiable re-parameterization module to construct spatial-aware representations from joint coordinates directly. In this module, for each hand joint, we calculate the 3D offset of the pixels relative to its coordinates and re-parameterize the 3D offsets as 3D heat maps and unit vector fields, reflecting the closeness and directions from pixel to the target joints, respectively. Taking the spatial-aware representations as intermediate features, we can stack multiple regression modules to repeatedly predict joint coordinates, which allows the network to capture the 3D spatial structure of depth data and reevaluate the initial estimations using the 3D information and multi-joint spatial context. Furthermore, we integrate and explore multiple good practices including data augmentation, smooth L1 loss, localization refinement and coordinate decoupling to improve the performance of 2D CNN for 3D hand pose estimation. Compared with 3D input data based methods [2, 7, 8, 13], regressing joint coordinates from the 2D depth image avoids tedious pre-processing and post-processing. Compared with [33], directly constructing the 3D heat maps and unit vector fields from the regressed pose is more efficient. At the same time, non-parametric re-parameterization module can generate stable and high-quality spatial-aware representations, which help the subsequent stages to better reevaluate the initial estimations. In addition, benefiting from the holistic regression, which is able to capture global constraints and correlations among different joints [36], our method is robust to self-occlusion and low-quality images. To the best of our knowledge, this is the first work that combines bottom-up, top-down repeated inference with 2D CNN regression-based method for 3D hand pose estimation.

We evaluate our proposed method on four publicly available 3D hand pose datasets (Hands17 [36], NYU [31], ICVL [29], MSRA [27]). Our method outperforms all previous state-of-the-art approaches on four datasets with runtime speed of 263.1FPS and network model size of 21.3MB.

Our main contributions can be summarized as follows: **(1)** We combine the regression-based method with stacked architecture through a differentiable pose re-parameterization module. Our proposed method is able to efficiently utilize the 3D spatial information in the depth image for accurate 3D hand pose estimation. At the same time, holistic regression allows us to capture global constraints and correlations among different joints, which improves the robustness of the network. **(2)** We explore multiple good practices for 3D hand pose estimation based on 2D CNN, which improve the accuracy of the estimation. **(3)** We design a relatively lightweight architecture, which can achieve better performance over state-of-the-art approaches on four publicly available datasets.ie., Hands17 [36], NYU [31], ICVL [29] and MSRA [27] with fewer parameters and faster frame rate. The source code will be made available at https://github.com/RenFeiTemp/SRN.

2 Related work

3D hand pose estimation methods can be divided into discriminative approaches [1, 3, 4, 6, 7, 8, 9, 14, 15, 29, 31, 33], generative approaches [11, 18, 22, 24, 25, 28, 30] and hybrid approaches [21, 23, 26]. In this section, we will focus on the deep neural network based methods.

Deep neural network based methods can be classified into regression-based and detection-based methods. Regression-based methods directly predict the 3D joint locations from a single depth image. Oberweger et al.[15] integrate a prior on hand joint distribution space into the 2D CNN and use initial joint locations to extract multi-scale input patches to improve the accuracy of the local estimation. Then, they [14] improve their previous work by adopting a more powerful network architecture, data augmentation and better initial hand localization. Guo et al. [9] present a region ensemble network and use the smooth L1 loss to reduce the sensitivity of the network to outliers. Ren et al. [1] use the initial pose to divide the feature maps and get more representative features for refining stage. Different from those methods that treat depth maps as 2D images, our method can capture the spatial structure of the depth data by pose re-parameterization. For detection-based method, a probability density map of each joint is predicted. Tompson et al. [31] predict a set of 2D heat maps and infer 3D hand poses with inverse kinematics. Ge et al. [6] predict heat maps on multiple views and fuse them to get 3D joint locations. Moon et al. [13] use a 3D CNN to predict 3D heat map of each joint, resulting in the best overall performance in the HIM17 challenge. However, using 3D CNNS is computationally inefficient. Ge et al. [7] use PointNet [20] as backbone to estimate the hand pose from point clouds, which achieves satisfying performance. However, their methods require tedious pre-processing and post-processing, which includes oriented bounding box calculation, surface normal estimation and fingertip refinement.

Recently, Wan et al. [33] and Ge et al. [8] adopt a dense pixel-wise estimation method that applies an encode-decode structure to generate 3D heat maps as well as 3D unit vector fields, from which the 3D hand joint locations can be inferred. Our method is inspired by these works [8, 33] but is essentially different from them. Firstly, the methods proposed in [8, 33] require a complex network structure to perform stable pixel-wise estimation, which significantly increases network parameters and inference time. Different from them, we adtop an efficiently pose re-parameterization module which can generate high-quality 3D heat maps and 3D unit vector fields with little overhead in computation and storage. Secondly, benefiting from regressing the joint locations directly, our method better captures global constraints and association between different joints, thus is more robust to self-occlusion and low-quality images, while the methods proposed in [8, 33] must set the divergence of the candidate estimations carefully to mitigate the effect of missing depth data near the hand joint. In addition, our method does not require any additional preprocessing such as point cloud normalization [8] or post-processing like the mean shift [33], which greatly simplified the entire framework. Wu et al. [34] combine the detection-based method with the regression-based method via intermediate dense guidance map supervision. Instead of adopting feature space constraints, our method incorporates coordinate space constraints on the joint predictions, which is more efficient. Furthermore, our method can project the joint coordinate to

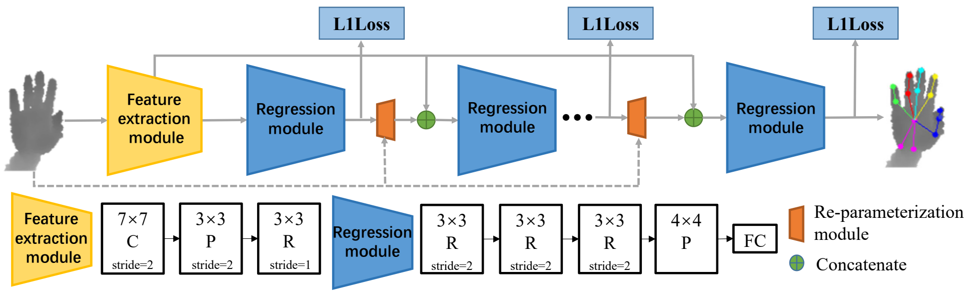


Figure 1: Network architecture. The abbreviations C, P, R, FC stand for convolution layer, pooling layer, residual module and fully connected layer respectively. Batch normalization and Relu are introduced after all convolution layers.

the image space and does not require searching for hyper-parameters to balance different loss items, the prior knowledge such as bone structure [39] or bottleneck layers [15], can be easily adopted into the intermediate stage to provide the explicit spatial constraints to the subsequent stages.

3 Methodology

4 Pose Re-parameterization

Most existing CNN-based methods for 3D hand pose estimation treating depth maps as a 2D image [9, 14, 15, 16, 39], which under-utilize the present information. Different from them, our method aims at incorporating the 3D spatial relationship between joint locations and the depth image in networks inference. This relationship can be defined as offsets from the pixels on the depth map to the joint locations. The multi-joint offset information contains rich context which provides strong disambiguating cues to a subsequent stage. At the same time, the 3D spatial relationships information between the estimated pose and the depth data allow subsequent stages to leverage the 3D geometric properties of the depth image to predict a more accurate result. However, due to the large variance of offsets, it is difficult to train a neural network with the offset field as input directly. Similar to [8, 33], we decompose the joint coordinates into 3D heat maps H and directional unit vector fields D as follows:

$$H_j(p) = \begin{cases} \lambda\left(\Theta - \|p-p_j\|_2\right) & \|p-p_j\|_2 \leq \Theta, \\ 0 & \text{otherwise;} \end{cases} \tag{1}$$

$$D_j(p) = \begin{cases} \lambda\left(\frac{p-p_j}{\|p-p_j\|_2}\right) & \|p-p_j\|_2 \leq \Theta, \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

where $p \in R^3$ and $p_j \in R^3$ are the coordinates of a pixel in the depth map and of joint j respectively; Θ denote the maximum length between the p and p_j, which we called the kernel size of the 3D heat map; λ equals to zero if p belongs to background and equals to one otherwise.

A decoding network [8, 33] is often used to generate heat maps and unit vector fields. However, the decoding network usually takes up half of the network parameters and inference time, which greatly reduces the efficiency of the whole network. In addition, the spatial-aware representations predicted by the neural network may contain some noise estimations. Although this problem can be alleviated by adopting a weighted averaging strategy when recovering joint position from the representations, the quality of the representations will still affect subsequent stages in a stacked architecture. Thus, we propose a differentiable re-parameterization module, which uses Eq.1, 2 to directly construct 3D heat maps and unit vector fields from joint coordinates and the depth map. The 3D heat maps and unit vector fields generated by the non-parametric function are stable and high-quality, which are able to more accurately reflect the spatial relationship between the pose and depth data and boost the estimation accuracy. Most important of all, this module is very efficient, it can be easily integrated into the regression-based network and almost does not increase the inference timeand the number of parameters of the network.

4.1 Network architecture

In this work, we exploit the 2D CNN for the 3D hand pose estimation from a depth image. Fig. 1 illustrates the architecture of our Stacked Regression Network(SRN). SRN consists of a feature extraction module and multiple regression modules which are connected by the re-parameterization module. ResNet [10] is the most common backbone network for image feature extraction, and it has shown powerful learning ability. Bounded by the first residual block, our method simply divides ResNet into two parts which are used as the feature extraction module and the pose regression module, respectively. Firstly, the input image is passed through the feature extraction module, which helps our network reduce the computational cost by working on lower resolution features which are called preliminary features. Then the regression modules estimate the joint coordinates in turn, while intermediate supervision is applied at the end of each module. Specifically, in addition to the first regression module, whose input contains only the preliminary features, other regression modules take the heat maps and unit vector fields together with the preliminary features as input and output a more accurate pose. For a re-parameterization module, it takes the joint coordinates and the depth map as input, and outputs a set of 3D heat maps and unit vector fields. Such spatial-aware representations can provide the 3D spatial structure of the depth data and strong disambiguate cue to the subsequent regression module for refining the initial pose. What's more, compared with generating pixel-wise or point-wise estimations, directly regressing the joint coordinates is able to better capture global constraints among different joints and avoid the complex preprocessing and post-processing. Different from [8, 33] that train the network in a multi-task manner, in our framework, the tasks of the intermediate supervision and the final regression stage are unified, which makes our method easily combined with other methods without carefully choosing hyper-parameters to balance different loss items. Specifically, the loss function for J joints from T stages is defined as:

$$\mathcal{L} = \sum_{t=1}^{T} \sum_{j=1}^{J} smooth_{L1} \left(C_j^t - C_j^* \right) \tag{3}$$

where C_j^* represent the ground-truth coordinates of joint j and C_j^t is corresponding estimation from tth stage.

4.2 Implementation details

The network is implemented with PyTorch [19] and is trained using stochastic gradient descent (SGD) with a momentum of 0.9. We choose the batch size as 32 and the weight decay as 0.0005. For NYU [31] and MSRA [27] datasets, the initial learning rate is set to 0.3 and is divided by 10 after every 25 epochs with 80 epochs. For Hands17 [36] and ICVL [29] datasets, the initial learning rate is set to 0.3 and is divided by 10 after every 5 epochs with 20 epochs. We adopt Resnet18 as the backbone and use 2 stacks by default, unless otherwise noted. Moreover, we adopt several important strategies for training.

Coordinate Decoupling In standard hand pose estimation pipelines, an image patch will be extracted by a fixed-size 3D bounding box around the hand from the original depth image. Then, the patch will be scaled to fixed size as input image to the network. Let $(X_c, Y_c, Z_c), (U_c, V_c, D_c)$ be the center of the bounding box in camera coordinate system and image coordinate system, respectively. We define a world coordinate system whose origin is (X_c, Y_c, Z_c) as bounding-box coordinate system (BB C.S) and an image pixel coordinate system whose origin is the center of the scaled image as scaled-image coordinate system(SI C.S). Then, let (x_i, y_i, z_i) and (u_i, v_i) be the location of point i in BB C.S and in SI C.S., respectively. The x_i is computed as:

$$\begin{aligned} x_i &= \frac{(u_i \alpha_u + U_c - f_u)(z_i + Z_c)}{f_x} - X_c, \\ \alpha_u &= \frac{c f_x}{w z_c} \end{aligned} \tag{4}$$

where w, c, f_x, f_u are the width of the input image, the size of the 3D bounding box, the camera focal length and original image center for x axis, respectively. As shown in Eq. 4, although the camera parameters, input size of the image and the bounding box size are fixed for a dataset, the relationship between a point in BB C.S and its corresponding position in the SI C.S is also affected by the center of the bounding box (U_c, V_c, D_c) and the depth of the point (z_i). Thus, directly predicting the joint locations in BB C.S from the input image is difficult due to the instability mapping relationship between the depth point and its coordinates on the input image. To make the estimated joint coordinates consistent with the appearance of the input image, we decouple the regression task into predicting image coordinates and their corresponding depths, which can be converted to 3D coordinates in the real-world coordinates using Eq. 4 during testing.

Localization refinement For evaluation and real-time applications, getting an accurate 3D bounding box that contains the hand is important for the final estimation. In the early work, the bounding box is extracted around the center of mass after simple depth thresholding around the hand region. However, the computed center-of-mass does not guarantee that the hand is correctly contained in the acquired bounding box when other objects are also within this threshold. Thus, similar to [13], we train a simple 2D CNN to obtain an accurate bounding box by estimating a 3D offset from the center-of-mass to the center of ground-truth joint locations. Furthermore, We used this refinement process not only in the test phase but also in the training phase.

Data augmentation The data augmentation helps to prevent overfitting and can enhance the generalization ability of the network. We take the data augmentation method used in [14], which augments depth data in real-world coordinate system, because original topology will be destroyed when resizing or translating the cropped image directly. We perform

Methods	Average error(mm)
Resnet18	34.11
+Data augmentation	15.42
+Localization refinement(only test)	13.70
+Localization refinement	10.64
+L1loss	9.82(5.3,4.4,4.9)
+Coordinate Decoupling	8.91(4.3,4.1,4.9)

Table 1: Incremental study of our basic method on NYU [31]. Numbers in parentheses indicate the average error corresponding to three coordinate axes. 'only test' stands for performing the localization refining procedure only during the test phase.

data augmentation including rotation ([-180,180]), random scaling ([0.9,1.1]) and random translation ([-10,10]).

Smooth L1 loss Same as [9], we adopt smooth L1 loss, which is less sensitive to outliers than L2 loss. The smooth L1 loss function is defined as follows:

$$smooth_{L1} = \begin{cases} 0.5x^2 & |x| = 0.01 \\ 0.01\left(|x| - 0.005\right) & \text{otherwise} \end{cases} \tag{5}$$

5 Experiments

5.1 Dataset and Evaluation Metric

We conduct experiments on 4 publicly available datasets: Hands17 dataset [36], NYU dataset [31], ICVL dataset [29], MSRA dataset[27]. **The Hands17 dataset** [36] is currently the largest dataset available. It consists of 957K training and 295K testing depth images that are sampled from BigHand2.2M dataset [37] and First-Person Hand Action datasets(FHAD) [5]. There are five subjects in the training set and ten subjects in the testing set, of which five are the same as in the training set and five are new. The ground-truth of this dataset is the 3D coordinates of 21 hand joints. **NYU dataset** [31] consists of 72K training and 8.2K testing depth images. Following previous works [1, 13], we selected 14 joints from all annotated joints during training and testing. **ICVL hand dataset** [29] consists of 330K training and 1.6K testing depth images. The frames are collected from 10 different subjects with 16 joints for each subject. **MSRA hand dataset** [27] contains 76.5K images from 9 subjects with 17 hand gestures for each subject and 21 annotated joints. For evaluation, we adopt the leave-one-subject-out cross-validation strategy. We evaluate our approach using two widely used metrics: the average joint error and the success rate. The average joint error is the average Euclidean distance between the predicted joint location and ground truth for each joint overall test frames. The success rate is the fraction of the frames whose average joints error within a maximum distance threshold.

5.2 Ablation study

The ablation experiment is conducted on the NYU [31] datasets since it has larger variance in pose compare to the other dataset and far from saturated.

Stack Number T	1	2	3
	9.81 mm	7.78 mm	7.75 mm

Kernel size Θ	0.3	0.4	0.5	0.6	0.7	0.8
	7.99 mm	7.78 mm	7.86 mm	7.87 mm	7.87 mm	7.90 mm

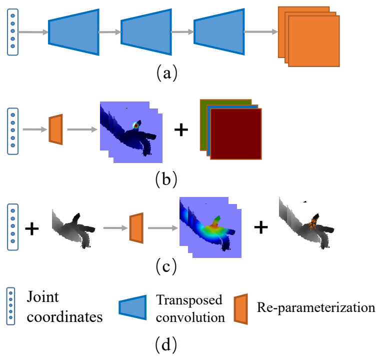
Table 2: Impact of hyperparameters. Θ is the kernel size for 3D heat map and directional unit vector. T is the number of network stacks. We report the average joint error on NYU [31]

Firstly, we incrementally introduce strategies on the single-stage network (Resnet18 [10]). As shown in Table 1, using data augmentation results in an increase in accuracy over 19mm. This indicates that there is a serious over-fitting without data augmentation. Adopting the localization refining procedure significantly boosts the accuracy of our model, which shows that the hand bounding box location has a crucial influence on the estimation. It is worth mentioning that our localization refining procedure has brought greater improvements than [14] because we use the center of ground-truth joint locations as the target center instead of the metacarpophalangeal joint of the middle finger and we adopt this method in both training phase and testing phase. L1 loss is helpful because it is more suitable for labels with large noise. Benefit from regressing the joint locations in image coordinates, the error of the xy coordinates has dropped significantly, which indicates that our method can better capture the relationship between the joint coordinates and its appearance in the depth image.

Secondly, we experiment on the number of network stacks T and the kernel size Θ of the 3D heat map as well as directional unit vector fields. As shown in table 2, compared with direct regression, performing refinement process once is able to make the error drop rapidly. On this basis, continuing to stack the network has little effect on the results. In addition, the average joint error is the smallest when the kernel size Θ is between 0.4 and 0.5. As such, we choose 2 stacks and kernel size Θ =0.4 in the following experiments.

Then, to demonstrate the validity of our re-parameterization module, we compared the performances of two different stacked methods which are designed for the regression-based method as Fig. 2 shows. Specifically, for deconvolution method, the joint coordinates are reshaped to feature maps whose size is $1 \times 1 \times 3J$ and then we map the feature maps to the desired resolution by three deconvolution layers. For 2D heat map based method, the joint coordinates are reparameterized into 2D heat maps. In order to incorporate the depth information into this representation, we created extra depth maps, which have the same resolution as the 2D heat maps and denoting the depth of a joint at the corresponding location. We choose the 2D heat maps with kernel size 3.4. As is shown in Fig. 2, due to the extreme mismatch between pose space and image feature space, directly projecting joint locations back to the feature space through deconvolution is of little help to the prediction in the next stage. 2D heat maps provide rich spatial context information for subsequent stages, thus it is helpful to improve the accuracy of estimation. In addition, providing depth information further improves refining effectiveness, indicating that it is important to explicitly consider depth information. Nevertheless, our method outperforms it for both metrics, which shows that the 3D heat maps and directional unit vectors are able to better capture the geometric and spatial property of depth data.

To further prove the effectiveness of pose re-parameterization, we compare the computational complexity and accuracy of our stacked network with the basic regression method.



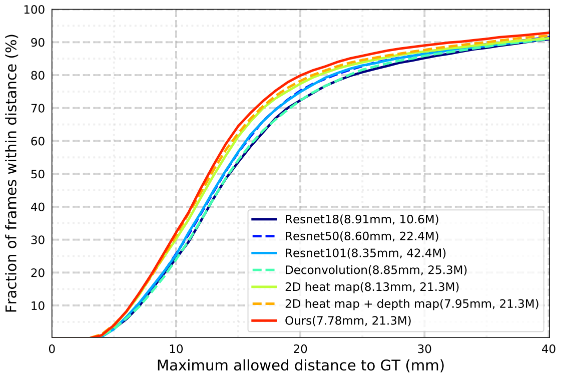


Figure 2: Left: Baseline network architectures. (a) Deconvolution method; (b) 2D heat map based method; (c) Ours method. (d) Detailed architecture configurations. To ensure a fair comparison to our proposed method, for deconvolution method and 2D heatmap method, we also use the Resnet18 as the backbone and stack two networks together. We choose 32×32 as the size of the output resolution of all three modules. Right: Self-comparison of different methods on NYU [31]. The overall mean error distances and the number of parameters shown in parentheses.

Method	Vanora	THU VCLab	oasis	RCN-3D	V2V-PoseNet	Ours
SEEN	9.55	9.15	8.86	7.55	6.97	**6.06**
UNSEEN	13.89	13.83	13.33	12.00	12.43	**10.33**
AVG	11.91	11.70	11.30	9.97	9.95	**8.39**

Table 3: Comparison with state-of-the-art methods on Hands17 [36]

As shown in Fig. 2, the number of parameters in our stacked network is much less than Resnet50 and Resnet101 with a smaller average joint error. It shows that stacking the refinement stage by a pose re-parameterization module is more effective than directly increasing network depth.

5.3 Comparisons with State-of-the-arts

We compared the performance of our proposed method on the four 3D hand pose datasets with most of the state-of-the-art methods which include latent random forest (LRF) [29], hand model based method (DeepModel) [39], improved DeepPrior (DeepPrior++) [14], region ensemble network (REN-9×6×6) [9], adversarial training architecture (CrossingNets) [32], pose guided structured REN (Pose-REN) [1], global-to-local prediction method (Global-to-Local) [12], regression using 3D CNN method (3DCNN) [4], dense 3D regression using 2D CNN (DenseReg) [33], 3D heat map estimation using 3D hourglass network (V2V-poseNet) [13], pose regression from point cloud (HandPointNet) [7], semantic hand pose regression from point clouds (SHPR) [2] and dense 3D regression using stacked pointnet (Point-to-Point) [8].

On Hands17 [36] dataset, we compare our proposed method with [1, 7, 13, 36]. As shown in table 3, our method achieves the lowest overall mean joint error of 8.39 mm on

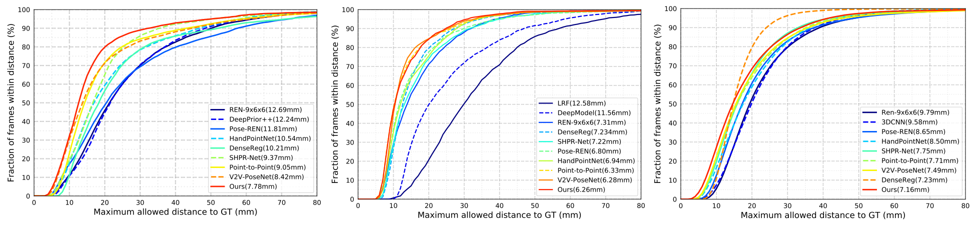


Figure 3: Comparison of the proposed method (SRN) with state-of-the-art methods. The proportions of good frames and the overall mean error distances (in parentheses) are presented in this figure. Left: NYU [31], middle: ICVL [29], right: MSRA [27].

the test set. For seen subjects hand and unseen subjects hand, the mean joint errors are 6.06 mm and 10.33 mm respectively. The results show that our method has high accuracy and good generalization ability. On NYU [31] dataset, we compare our proposed method with [1, 2, 7, 8, 9, 13, 14, 33]. Our approach outperforms all state-of-the-art methods with a large margin for both metrics. What's more, our method significantly improves the percentage of successfully predicted frames from 52% to 60% (relatively increased by 16%) on the threshold of 15mm and from 72% to 80% (relatively increased by 10%) on 20mm when compared to most accurate methods published to date. On ICVL [29] dataset, we compare our proposed method with [1, 2, 7, 8, 9, 13, 29, 33, 39], our method achieves similar accuracy as [8, 13] and outperforms the rest. When the error threshold is less than 10mm or more than 20mm, the proportions of good frames of our method is better than [8, 13]. On MSRA [27] dataset, we compare our proposed method with [1, 2, 4, 7, 8, 9, 33], our method outperforms other methods over almost all the error thresholds, except for the [33] method. However, as mentioned in [17] and shown in supplementary material, some of the 3D hand joint annotations in ICVL [29] dataset and MSRA [27] exhibit significant errors, which may limit the learning ability of our deep neural network and cause our method in these two datasets compared with state-of-the-art methods advantage is not obvious.

The model size is 21.3 MB for our method, while the model size of the 3D CNN in [13] is about 457.5MB. For the testing stage, the runtime of our method is 3.8 ms per frame, whereas DenseReg[33] requires 36 ms on a single GPU environment.

6 Conclusion and future work

In this paper, we present a simple but powerful network called Stack Regression Network(SRN) for 3D hand pose estimation from single depth map inputs. Firstly, by exploring some good practices, we make full use of the potential of 2D CNN. Then, we stack multiple regression networks together by pose re-parameterization, which allows the network to consider the 3D properties of the depth map and reevaluate the initial estimations. Furthermore, repeated regression process helps our network capture global constraints and correlations between different joints, which makes our network more robust to self-occlusion and image missing. Experimental results on four challenging hand pose datasets show that our method achieves superior accuracy and robust performance with less complexity and inference time. In future work, we plan to further extend our method for the cases of hand-object and hand-hand interaction.

References

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

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

[3] Chiho Choi, Sangpil Kim, and Karthik Ramani. Learning hand articulations by hallucinating heat distribution. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3104–3113, 2017.

[4] 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.

[5] Guillermo Garcia-Hernando, Shanxin Yuan, Seungryul Baek, and Tae-Kyun Kim. First-person hand action benchmark with rgb-d videos and 3d hand pose annotations. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[6] 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 *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[7] Liuhao Ge, Yujun Cai, Junwu Weng, and Junsong Yuan. Hand pointnet: 3d hand pose estimation using point sets. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[8] Liuhao Ge, Zhou Ren, and Junsong Yuan. Point-to-point regression pointnet for 3d hand pose estimation. In *The European Conference on Computer Vision (ECCV)*, September 2018.

[9] Hengkai Guo, Guijin Wang, Xinghao Chen, Cairong Zhang, Fei Qiao, and Huazhong Yang. Region ensemble network: Improving convolutional network for hand pose estimation. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 4512–4516. IEEE, 2017.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[11] Sameh Khamis, Jonathan Taylor, Jamie Shotton, Cem Keskin, Shahram Izadi, and Andrew Fitzgibbon. Learning an efficient model of hand shape variation from depth images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2540–2548, 2015.

[12] Meysam Madadi, Sergio Escalera, Xavier Baró, and Jordi Gonzalez. End-to-end global to local cnn learning for hand pose recovery in depth data. *arXiv preprint arXiv:1705.09606*, 2017.

[13] 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 *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[14] Markus Oberweger and Vincent Lepetit. Deepprior++: Improving fast and accurate 3d hand pose estimation. In *The IEEE International Conference on Computer Vision (ICCV) Workshops*, Oct 2017.

[15] Markus Oberweger, Paul Wohlhart, and Vincent Lepetit. Hands deep in deep learning for hand pose estimation. In *CVWW*, 2015.

[16] Markus Oberweger, Paul Wohlhart, and Vincent Lepetit. Training a feedback loop for hand pose estimation. In *The IEEE International Conference on Computer Vision (ICCV)*, December 2015.

[17] Markus Oberweger, Gernot Riegler, Paul Wohlhart, and Vincent Lepetit. Efficiently creating 3d training data for fine hand pose estimation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[18] Iason Oikonomidis, Nikolaos Kyriazis, and Antonis A Argyros. Efficient model-based 3d tracking of hand articulations using kinect. In *BMVC*, volume 1, page 3, 2011.

[19] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in PyTorch. In *NIPS Autodiff Workshop*, 2017.

[20] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems*, pages 5099–5108, 2017.

[21] Chen Qian, Xiao Sun, Yichen Wei, Xiaoou Tang, and Jian Sun. Realtime and robust hand tracking from depth. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

[22] Chen Qian, Xiao Sun, Yichen Wei, Xiaoou Tang, and Jian Sun. Realtime and robust hand tracking from depth. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1106–1113, 2014.

[23] Toby Sharp, Cem Keskin, Duncan Robertson, Jonathan Taylor, Jamie Shotton, David Kim, Christoph Rhemann, Ido Leichter, Alon Vinnikov, Yichen Wei, et al. Accurate, robust, and flexible real-time hand tracking. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 3633–3642. ACM, 2015.

[24] Toby Sharp, Cem Keskin, Duncan Robertson, Jonathan Taylor, Jamie Shotton, David Kim, Christoph Rhemann, Ido Leichter, Alon Vinnikov, Yichen Wei, et al. Accurate, robust, and flexible real-time hand tracking. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 3633–3642. ACM, 2015.

[25] Srinath Sridhar, Franziska Mueller, Antti Oulasvirta, and Christian Theobalt. Fast and robust hand tracking using detection-guided optimization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3213–3221, 2015.

[26] Srinath Sridhar, Franziska Mueller, Michael Zollhoefer, Dan Casas, Antti Oulasvirta, and Christian Theobalt. Real-time joint tracking of a hand manipulating an object from rgb-d input. In *Proceedings of European Conference on Computer Vision (ECCV)*, 2016.

[27] Xiao Sun, Yichen Wei, Shuang Liang, Xiaoou Tang, and Jian Sun. Cascaded hand pose regression. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015.

[28] Andrea Tagliasacchi, Matthias Schroeder, Anastasia Tkach, Sofien Bouaziz, Mario Botsch, and Mark Pauly. Robust articulated-icp for real-time hand tracking. *Computer Graphics Forum (Proc. Symposium on Geometry Processing)*, 2015.

[29] Danhang Tang, Hyung Jin Chang, Alykhan Tejani, and Tae-Kyun Kim. Latent regression forest: Structured estimation of 3d articulated hand posture. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2014.

[30] Anastasia Tkach, Mark Pauly, and Andrea Tagliasacchi. Sphere-meshes for real-time hand modeling and tracking. *ACM Transaction on Graphics (Proc. SIGGRAPH Asia)*, 2016.

[31] Jonathan Tompson, Murphy Stein, Yann Lecun, and Ken Perlin. Real-time continuous pose recovery of human hands using convolutional networks. *ACM Transactions on Graphics (ToG)*, 33(5):169, 2014.

[32] Chengde Wan, Thomas Probst, Luc J. Van Gool, and Angela Yao. Crossing nets: Dual generative models with a shared latent space for hand pose estimation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[33] Chengde Wan, Thomas Probst, Luc Van Gool, and Angela Yao. Dense 3d regression for hand pose estimation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[34] Xiaokun Wu, Daniel Finnegan, Eamonn O'Neill, and Yong-Liang Yang. Handmap: Robust hand pose estimation via intermediate dense guidance map supervision. In *The European Conference on Computer Vision (ECCV)*, September 2018.

[35] Chi Xu, Lakshmi Narasimhan Govindarajan, Yu Zhang, and Li Cheng. Lie-x: Depth image based articulated object pose estimation, tracking, and action recognition on lie groups. *International Journal of Computer Vision*, 123(3):454–478, 2017.

[36] Shanxin Yuan, Qi Ye, Guillermo Garcia-Hernando, and Tae-Kyun Kim. The 2017 hands in the million challenge on 3d hand pose estimation. *arXiv preprint arXiv:1707.02237*, 2017.

[37] Shanxin Yuan, Qi Ye, Bjorn Stenger, Siddhant Jain, and Tae-Kyun Kim. Bighand2.2m benchmark: Hand pose dataset and state of the art analysis. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[38] Shanxin Yuan, Guillermo Garcia-Hernando, Bj?rn Stenger, Gyeongsik Moon, Ju Yong Chang, Kyoung Mu Lee, Pavlo Molchanov, Jan Kautz, Sina Honari, Liuhao Ge, Junsong Yuan, Xinghao Chen, Guijin Wang, Fan Yang, Kai Akiyama, Yang Wu, Qingfu

Wan, Meysam Madadi, Sergio Escalera, Shile Li, Dongheui Lee, Iason Oikonomidis, Antonis Argyros, and Tae-Kyun Kim. Depth-based 3d hand pose estimation: From current achievements to future goals. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

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