SeñoritaHand: Analytical 3D Skeleton Renderer and Patch-based Refinement for HANDS19 Challenge Task 1 - Depth-Based 3D Hand Pose Estimation

Qingfu Wan
Fudan University
Shanghai
strawberryfgalois@gmail.com

Abstract

This essay describes our method for the HANDS2019 Challenge Task 1: Depth-Based 3D Hand Pose Estimation track, which is essentially built upon one recent technique patch-based pose refinement. We additionally propose a render-and-compare stage that synthesizes 3D skeleton volumes from 3D pose, thereby re-parameterizing the prediction. Also, we explore different forms of input data in our system. The approach is conceptually simple and ready to use here `https://github.com/strawberryfg/Senorita-HANDS19-Pose`.

1. Related Work

Skeleton Representation 2D skeleton [2] representation has been widely studied [17][10] [18] [16]. Not long ago [6] proposed an effective renderer to produce a clean 2D skeleton image from 2D keypoints.

In the 3D space, 3D bone heatmap (probability map) [4] (also called occupancy map in [13]), entails the connectivity of consecutive keypoints. Nevertheless, this representation is typically learned from networks which oftentimes results in multiple peaks.

In contrast, we provide a univocal analytical 3D skeleton renderer that enables voxel-wise supervision.

Patch-based Refinement [15] refine the initial pose estimation with local patches of segmentation and RGB image. [8] reveal the usage of patches around keypoints. We draw inspiration from both works.

Input 3D Data Normally the input to the depth-based 3D hand pose estimation system is the depth image. Apart from that, the literature is fraught with other modalities *e.g.* 3D points projection [3], multi-layer depth map [11], depth voxel [7]. We treat these as potential input to our system.

2. Method

Our ultimate goal is to learn a mapping from the depth image I to 3D joints $X \in R^{N\times 3}$ ($N = 21$). Hereafter we interchangeably use the terminology *joints* and *keypoints*.

In Fig. 1, we sketch the complete pipeline. Which starts with the transformed input data from the depth image (Sec. 2.1), and delivers an initial pose estimate through the integral pose regression module (Sec. 2.2). In addition, a 3D skeleton volume renderer (Sec. 2.3) re-parameterizes the initial pose to enforce voxel-wise supervision in training. On the grounds of local hand bone patches as well as the initial pose prediction, the patch-based refinement module (Sec. 2.4) is added to yield the refined pose result.

2.1. System Input

Basically, we consider four input representations for their popularity: (1) *depth image* (2) *3D points projection* (3) *multi-layer depth* (4) *depth voxel*. (2) is the result of projecting the 3D depth point cloud onto three orthogonal planes *i.e.* x-y, x-z, y-z. (3) is calculated by multi-hit ray tracing from the camera. (4) is computed via voxelizing the 2.5D point cloud in the depth image. Notice all the representations are derivative from the depth image. We present four different combinations of these input representations, which we call *Model 1*, *Model 2*, *Model 3* and *Model 4*. The combination here refers to plain concatenation. We see in Fig. 2 *Model 1* includes (1); *Model 2* includes (1), (2), (3); *Model 3* includes (1), (2), (4); *Model 4* includes (1), (2), (3), (4). We shall return to the evaluation of these four models in Sec. 4.

2.2. Integral Pose Regression

We employ the integral regression approach [12], which realizes the function $\mathbf{F} : I \rightarrow X$ turning the depth image to 3D keypoint locations. We follow the same network architecture.

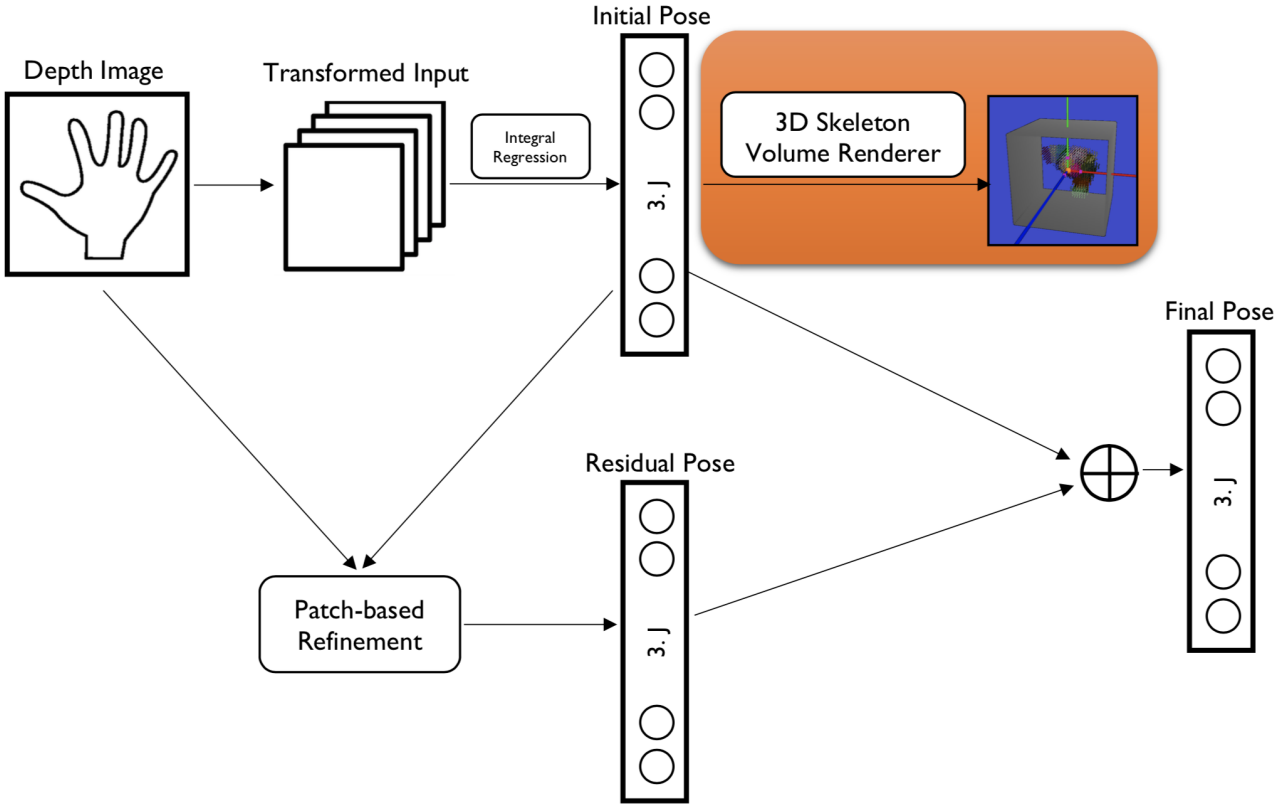

Figure 1. **Overview** of our method. The input depth image is transformed and forwarded to the integral regression network, which achieves the initial pose. When training the system, a 3D skeleton volume renderer generates 3D skeleton volumes from the initial pose. Afterward, the volumes are compared with ground truth volumes to guide the learning of the initial 3D pose. This part displayed in *orange* is removed during testing. Further, we refine the initial pose under the patch-based refinement module. The refinement takes as input the cropped patches from the depth image and the initial pose, then outputs a residual pose. Eventually, the residual pose and the initial pose add up to the refined pose that is our final output. The ground truth volumes deduced from ground truth pose are not plotted for clarity. The form of transformed input is explained in Fig. 2 & Sec. 2.1

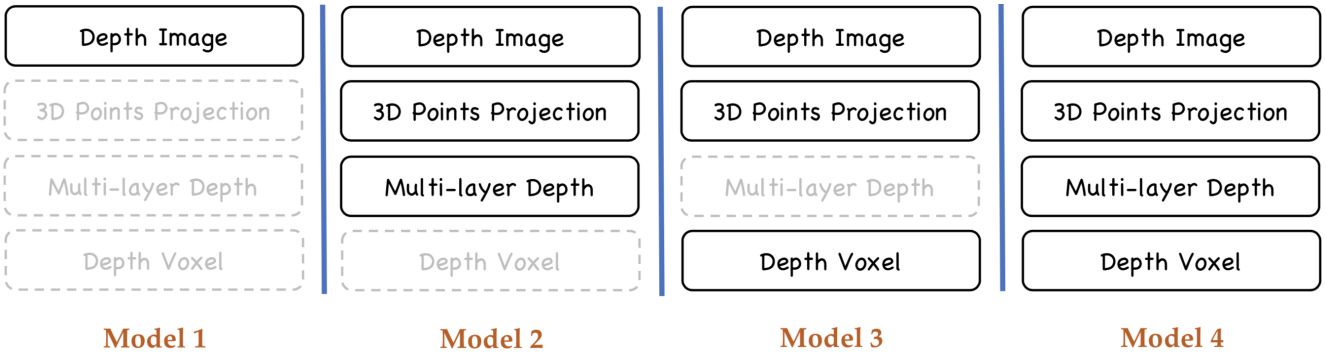

Figure 2. **Input data** in four models. Data used is shown in a *solid* rounded rectangle, and data unused is shown in *shadow* (*dashed* rounded rectangle). All four models differ only in the input data form.

2.3. 3D Skeleton Volume Renderer

After analysis by the integral regression, this synthesis process transits 3D pose vector to feature map that is able to acknowledge gradients.

The renderer has no network parameters to learn, rather it is formulated as an explicit function mapping the pose vector to the feature map space. The differentiable function $\mathbf{G} : X \rightarrow V$ maps 3D pose to 3D skeleton volumes, wherein dense voxel-wise supervision can be given. We visualize a synthesized 3D skeleton volume sample in Fig. 3.

Let E be the set of kinematic bones connecting adjacent keypoint pairs (u_i, v_i) ($i \in [0, 19]$, see [19] for the parent-child relationships). For each bone i, we render a 3D skeleton volume V^i. Let $p \in \{0, ..., D-1\} \times \{0, ...H-1\} \times \{0, ...W-1\}$ be a voxel grid therein, then we have

$$V_p^i = \mathbf{G}(X)_p^i = exp(-\gamma \cdot dist(p, X_{u_i} \rightarrow X_{v_i})) \quad (1)$$

Here $dist(p, X_{u_i} \rightarrow X_{v_i}))$ simply means the distance of a 3D voxel point p to the i-th 3D bone segment. Formally the distance is

$$\min_{r \in [0,1]} ||p - (rX_{u_i} + (1-r)X_{v_i})||^2 \quad (2)$$

This synthesis process can be conceived as a kernel that projects 3D joints to the feature space, and dense per-voxel supervision signal can be introduced in there. Notwithstanding the integration of per-voxel probability brought by the integral operation, the final output 3D joint coordinates vector still lies in the high-dimensional space. Eq. 1 in this context re-parameterizes the 3D pose into another space where guidance is more conveniently offered.

During training, we do this rendering for both the predicted and ground truth 3D pose, whereby a loss is brought into play. During inference, this render-and-compare step is discarded.

We will explicate the loss function in Sec. 2.5.

2.4. Patch-based Pose Refinement

The refinement component lays the foundation of our algorithm. We borrow the patch cropping module in Sec. 3.2 of [15] except for segmentation, since the depth image already contains sufficient depth information. The design purpose behind the patch cropping module is to bring the initial pose estimate closer to the ground truth location. In the original paper, the learning outcome is the residual orientation which subtracts predicted orientation from ground truth orientation. The essential point is that limb orientation, instead of keypoint location, is easier to model from per-part local appearance. [15] nonetheless does not impart the initial pose estimate information to the refinement module, which is included in this work.

Technically for each bone, we train a separate refinement network (Fig. 4). We build upon the architecture of [8] (*Green* rounded rectangle in the figure; called *Patch Stream*), aside from which is a simple multi-layer fully connected network (*Orange* rounded rectangle in the figure; called *Initial Pose Stream*). The FC layers in [8] are replaced with one FC layer of 256 neurons. Taking the initial prediction into account, we obtain the initial orientation from the initial pose estimate (Sec. 2.2). And then we pass this bone vector to the *Initial Pose Stream*, which will generate a 256-d vector. Finally, we concatenate the output from both the *Patch Stream* and the *Initial Pose Stream*, upon which another FC layer produces the ultimate orientation increment. Similar to [15], we convert orientation to 3D pose using the hand shape (*i.e.* bone length) statistics on the training set of HANDS19 dataset. Note batch normalization and ReLU are always attached after each *Conv* and *FC* layer.

We dwell on the detail about how to train the patch-based refinement module in the next section.

2.5. Losses

We elaborate on all the losses by virtue of which we train the whole system including the initial estimator and the refinement part.

The integral loss $\mathbf{I}^*$ in [12] is adopted to train the initial pose estimator. Denote the initial pose estimate and ground truth pose as $\hat{X}^{(0)}$, X^{gt}, the loss is therefore

$$\mathcal{L}_{jt} = ||\hat{X}^{(0)} - X^{gt}||_1 \quad (3)$$

Let us write $\hat{V}^{(0)}$, V^{gt} as the generated skeleton volumes of initial pose estimate and ground truth pose, the loss is L1 distance between them:

$$\mathcal{L}_{3d\_ske} = ||\hat{V}^{(0)} - V^{gt}||_1 \quad (4)$$

Using the notation in [15], the loss term on regressed residual orientation reads

$$\mathcal{L}_{res\_ori} = ||\Delta U - (U^{gt} - \hat{U}^{(0)})||_1 \quad (5)$$

, meaning the L1 distance between predicted residual orientation and ground truth orientation. Besides, we empirically find that further supervising the refined 3D pose benefits the learning, which leads us to

$$\mathcal{L}_{jt\_ref} = ||\hat{X}^{(1)} - X^{gt}||_1 \quad (6)$$

3. Implementation Details

We crop the depth image in a *coarse-to-fine* manner. Provided the bounding box, we extend it to a square on which Otsu's thresholding [9] is initially performed to single out the foreground. Assuming the hand is the closest object to

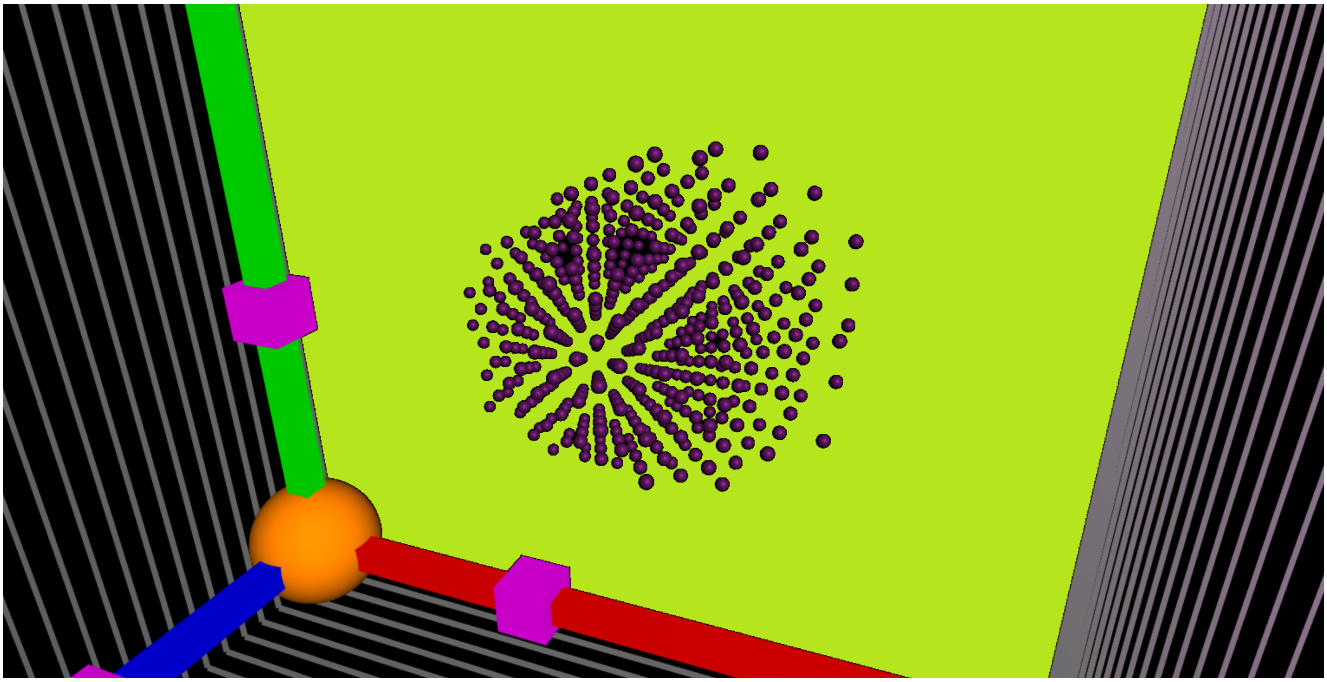

Figure 3. A vivid example presenting the **synthesized 3D Skeleton Volume**. *Purple* spheres indicate non-zero voxel grids regardless of detailed magnitudes. The $x\text{-}axis$, $y\text{-}axis$, $z\text{-}axis$ of the cartesian coordinate system are colorized in *red*, *green*, *blue* respectively. Illustrational purpose only.

the camera, the rough center of mass is computed by taking the closest non-zero pixel to the box center. To finalize the crop size, the box is scaled by a factor inversely proportional to the average depth value of the hand pixels, so that the hand size ($\frac{500.0}{avgZ} * 210.0$) is depth invariant. After the coarse step, we conduct fine background subtraction by pruning outlier pixels whose depth values fall out of range $[avgZ - 100, avgZ + 100]$. Next, we re-calculate the center of mass around which a fixed $200 \times 200 \times 200$ cube is defined and resize the image to a 256×256 depth patch of values normalized to $[-1, 1]$.

The projection of depth point clouds onto $x\text{-}y$, $x\text{-}z$, $y\text{-}z$ planes are resized to 256×256. The number of multi-layer depth maps is 64. The depth dimension of depth voxel is 32, while the other two dimensions are both 256 to align with other input representations.

γ in Eq. 1 is 0.1. The shape of 3D skeleton volumes is $(32, 32, 32)$.

The tight bounding box $h \times w$, defined by the 2D keypoints of each bone, is center padded to $max(10, h) \times max(10, w)$. The rescaling factor is empirically set to 1.5. The cropped patches are resized to 96×96, 48×48 and 24×24 to fulfill the multi-scale architecture in [8].

3.1. Training

We make use of one Dell Alienware 17 R4 laptop equipped with a single 8GB GeForce GTX 1070 card. RMSProp solver is taken throughout the learning. Data augmentation includes random rotation ([-30, 30] degrees), random scaling ([0.75, 1.25]) with a probability of 0.5, and random translation ([-5, 5] pixels). The following steps apply to all four models. We refer interested readers to our Github repo `https://github.com/strawberryfg/Senorita-HANDS19-Pose` for a comprehensive training schedule. Directly training the initial pose estimator and the refinement component from scratch is anything but easy, and so we take two steps below.

Step 1 - Initial Pose We first train the integral regression network on the Monocular 3D Human Pose Estimation Task. As done in [12], we perform joint 2D and 3D training on Human3.6M[5] and MPII[1]. The pre-trained weights are directly transferred to the HANDS19 Depth-Based 3D Hand Pose Estimation task up to the penultimate layer. The penultimate layer, which originally outputs the 3D heatmap cube of 16 keypoints, is adjusted to encompass 21 keypoints. After transferring weights, we train the network on HANDS19 with only $\mathcal{L}_{jt}$ using a batch size of 6. $\mathcal{L}_{3d\_ske}$ comes into play when the network converges on the validation set. The weight ratio of $\mathcal{L}_{jt}$ to $\mathcal{L}_{3d\_ske}$ is set on a case-by-case basis, which is roughly $200 : 1$, through observing the training loss magnitudes. The running iteration for this step is around $70k$. Generally, the base learning rate is 0.0007, and will be divided by 5 at $10k$, $34k$, $58k$ iterations.

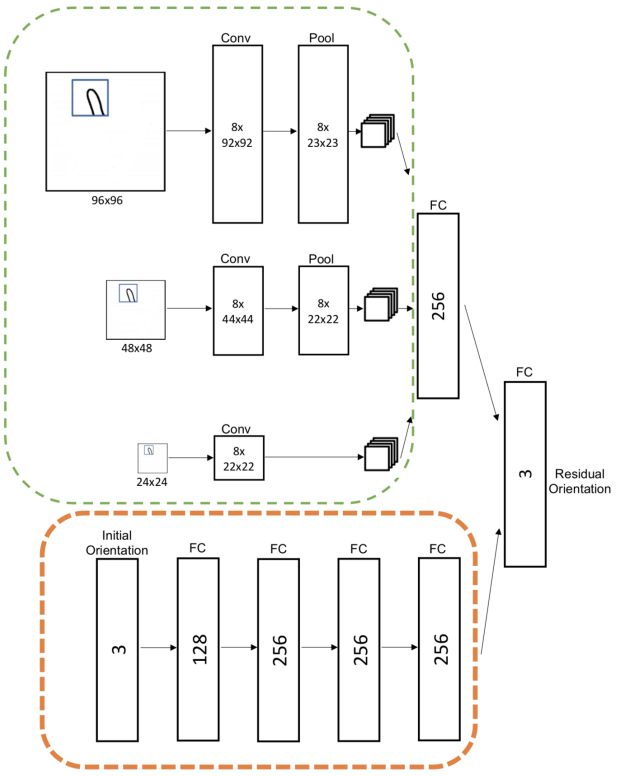

Figure 4. The illustration of the **patch-based refinement**. We utilize two branches, one for local patches (*Green*; *Patch Stream*) and one for the initial pose prediction (*Orange*; *Initial Pose Stream*). The final output is a 3-d vector denoting the residual orientation. See [15] for more details about residual orientation. The conv-pool layers dashed in *Green* are the same as that in [8]. We append batch normalization and ReLU activation, which are omitted for brevity, after each convolution or fully connection layer.

Step 2 - Pose Refinement This step features end-to-end training with the aid of $\mathcal{L}_{jt}$, $\mathcal{L}_{res\_ori}$ and $\mathcal{L}_{jt\_ref}$, as such extends beyond the confines of separate training in [15]. In doing so, the integral regression component also receives gradient flow, unlike [15]. Batch size is reduced to 4. Learning rate is scaled by $\frac{1}{10}$. To balance disparate losses, we compute the gradient magnitude of each loss w.r.t. each output neuron set and set the ratio accordingly [14]. The intention is to ensure that the gradient ascribed to each loss is of the same or similar level. This step runs for $\sim 18k$ iterations.

Concerning the network initialization, all *Conv* layers use xavier initialization. All *FC* layers, precluding the last one that yields residual orientation, are also initialized with xavier. The last *FC* layer is filled with gaussian weights, the standard variance thereof is 0.01.

4. Results

We evaluate *Model 1*, *Model 2*, *Model 3*, *Model 4* on HANDS19 Challenge Dataset (`https://sites.google.com/view/hands2019/challenge`), and exhibit the results here. It is abundantly clear in Tab. 1 that the best single-model is *Model 1*, which reports an average error of *20.99 mm* on test samples that have **hand shapes, viewpoints and articulations** *not present* in the training set. It follows from Tab. 2 that merging the output of these four models advances the performance to *19.63 mm* on EXTRAP., reflecting the nature of ensembling.

5. Conclusion

We introduce a neat solution for HANDS19 Challenge Task 1: Depth-based 3D Hand Pose Estimation. Based on the initial pose estimate, a differentiable 3D skeleton volumes renderer and a patch-based refinement module constitute our system. These two components are efficacious and efficient adds-on that apply to any existing method.

References

[1] Mykhaylo Andriluka, Leonid Pishchulin, Peter Gehler, and Bernt Schiele. 2d human pose estimation: New benchmark and state of the art analysis. In *Proceedings of the IEEE Conference on computer Vision and Pattern Recognition*, pages 3686–3693, 2014.

[2] Harry Blum. Biological shape and visual science (part i). *Journal of theoretical Biology*, 38(2):205–287, 1973.

[3] 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 *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3593–3601, 2016.

[4] Fuyang Huang, Ailing Zeng, Minhao Liu, Jing Qin, and Qiang Xu. Structure-aware 3d hourglass network for hand pose estimation from single depth image. *arXiv preprint arXiv:1812.10320*, 2018.

[5] Catalin Ionescu, Dragos Papava, Vlad Olaru, and Cristian Sminchisescu. Human3. 6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE transactions on pattern analysis and machine intelligence*, 36(7):1325–1339, 2013.

[6] Tomas Jakab, Ankush Gupta, Hakan Bilen, and Andrea Vedaldi. Learning landmarks from unaligned data using image translation. *arXiv preprint arXiv:1907.02055*, 2019.

[7] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5079–5088, 2018.

[8] Markus Oberweger, Paul Wohlhart, and Vincent Lepetit. Hands deep in deep learning for hand pose estimation. *arXiv preprint arXiv:1502.06807*, 2015.

Approach	EXTRAP.	INTERP.	SHAPE	ARTIC.	VIEWP.
Model 1	20.99	9.35	14.85	8.42	14.70
Model 2	21.39	9.17	15.21	8.25	15.34
Model 3	21.02	9.61	15.30	8.52	16.12
Model 4	21.19	9.32	15.23	8.36	15.78

Table 1. Results on HANDS19 Challenge Dataset. Numbers are in *mm*.

Approach	EXTRAP.	INTERP.	SHAPE	ARTIC.	VIEWP.
Ours	19.63	8.42	14.21	7.50	14.16

Table 2. Finalized result.

[9] Nobuyuki Otsu. A threshold selection method from gray-level histograms. *IEEE transactions on systems, man, and cybernetics*, 9(1):62–66, 1979.

[10] Wei Shen, Kai Zhao, Yuan Jiang, Yan Wang, Xiang Bai, and Alan Yuille. Deepskeleton: Learning multi-task scale-associated deep side outputs for object skeleton extraction in natural images. *IEEE Transactions on Image Processing*, 26(11):5298–5311, 2017.

[11] Daeyun Shin, Zhile Ren, Erik B Sudderth, and Charless C Fowlkes. Multi-layer depth and epipolar feature transformers for 3d scene reconstruction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 39–43, 2019.

[12] Xiao Sun, Bin Xiao, Fangyin Wei, Shuang Liang, and Yichen Wei. Integral human pose regression. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 529–545, 2018.

[13] Gul Varol, Duygu Ceylan, Bryan Russell, Jimei Yang, Ersin Yumer, Ivan Laptev, and Cordelia Schmid. Bodynet: Volumetric inference of 3d human body shapes. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 20–36, 2018.

[14] Qingfu Wan. Re-implementation of integral human pose regression I1 loss. `https://github.com/strawberryfg/int-3dhuman-I1`, 2018. [Online].

[15] Qingfu Wan, Weichao Qiu, and Alan L Yuille. Patch-based 3d human pose refinement.

[16] Qingfu Wan, Wei Zhang, and Xiangyang Xue. Deepskeleton: Skeleton map for 3d human pose regression. *arXiv preprint arXiv:1711.10796*, 2017.

[17] Saining Xie and Zhuowen Tu. Holistically-nested edge detection. In *Proceedings of the IEEE international conference on computer vision*, pages 1395–1403, 2015.

[18] Weijian Xu, Gaurav Parmar, and Zhuowen Tu. Geometry-aware end-to-end skeleton detection.

[19] 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.