AHRNET: ATTENTION AND HEATMAP-BASED REGRESSOR FOR HAND POSE ESTIMATION AND MESH RECOVERY

Feng Zhou[1], *Pei Shen*[1], *Ju Dai*[2]*, *Na Jiang*[3], *Yong Hu*[4], *Yu-Kun Lai*[5], *Paul L. Rosin*[5]

[1]North China University of Technology [2]Peng Cheng Laboratory [3]Capital Normal University
[4]State Key Laboratory of Virtual Reality Technology and Systems, Beihang University
[5]Cardiff University

ABSTRACT

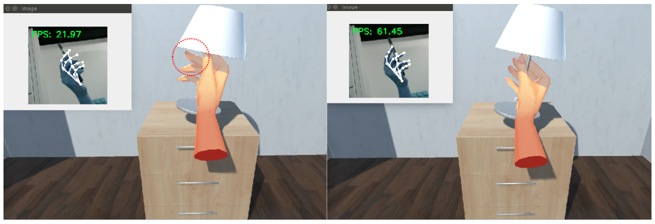
Estimating 3D hand pose and recovering the full hand surface mesh from a single RGB image is a challenging task due to self-occlusions, viewpoint changes, and the complexity of hand articulations. In this paper, we propose a novel framework that combines an attention mechanism with heatmap regression to accurately and efficiently predict 3D joint locations and reconstruct the hand mesh. We adopt a pooling attention module that learns to focus on relevant regions in the input image to extract better features for handling occlusions, while greatly reducing the computational cost. The multi-scale 2D heatmaps provide spatial constraints to guide the 3D vertex predictions. By exploiting the complementary strengths of sparse 2D supervision and dense mesh regression, our method accurately reconstructs hand meshes with realistic details. Extensive experiments on standard benchmarks demonstrate that the proposed method efficiently improves the performance of 3D hand pose estimation and mesh recovery. The reproducible recipes are available at https://github.com/SDiannn/AHRNET-Heatmap.

Index Terms— Hand Pose, Mesh Recovery, Deep Learning, Human-computer Interaction, Heatmap

1. INTRODUCTION

Hand pose estimation and surface reconstruction from single RGB images enable many applications in human-computer interaction [1, 2, 3, 4, 5], augmented/virtual reality [6, 7], and robotics [8]. However, this task remains challenging due to complex hand articulations, self-occlusions, and viewpoint variations. Whilst data-driven deep learning approaches have achieved promising results, efficiently predicting accurate 3D hand joints and reconstructing a realistic surface mesh from limited 2D supervision remains an open challenge.

Prior model-based methods use an articulated hand model and optimize model parameters to fit observations. However, these methods often fail to capture finer details of hand geometry. More recent data-driven approaches utilize neural networks and large datasets to learn a direct mapping from an input image to 3D hand pose and shape. While achieving more realistic outputs, these methods still struggle with occluded joints, limited training data, and computational constraints.



(a) (b)

Fig. 1. A simple game interaction scene. (a) A scene in which the linear blend skinning model of the hand is driven by the predicted 3D joint points of the hand in the video to interact with the object at low FPS. (b) Interactive scene at high FPS. At low FPS, there is a possibility that the prediction of the 3D joint points of the hand in a certain frame is inaccurate, resulting in abnormal hand deformation when driving the linear blend skinning model of the hand, as highlighted by the example in the red circle where the finger deforms abnormally. Best viewed in color.

Recently, driven by the evolution of the Transformer architecture in natural language processing, the Vision Transformer (ViT) [9] has successfully introduced the Transformer architecture into the field of computer vision. This groundbreaking development has brought new perspectives and approaches to computer vision research. With this trend, Transformer-based models have spurred various computer vision tasks, including object detection, semantic segmentation, and video understanding, achieving impressive results. Currently, a plethora of research has adopted this approach to address tasks such as hand pose estimation and mesh reconstruction, yielding remarkable outcomes [10, 11]. METRO [10] is a pioneering work in human mesh recovery via Transformer, showcasing significant superiority over other methods. However, it demands substantial computational resources. To address this issue, FastMETRO [11]

*Corresponding author:daij@pcl.ac.cn

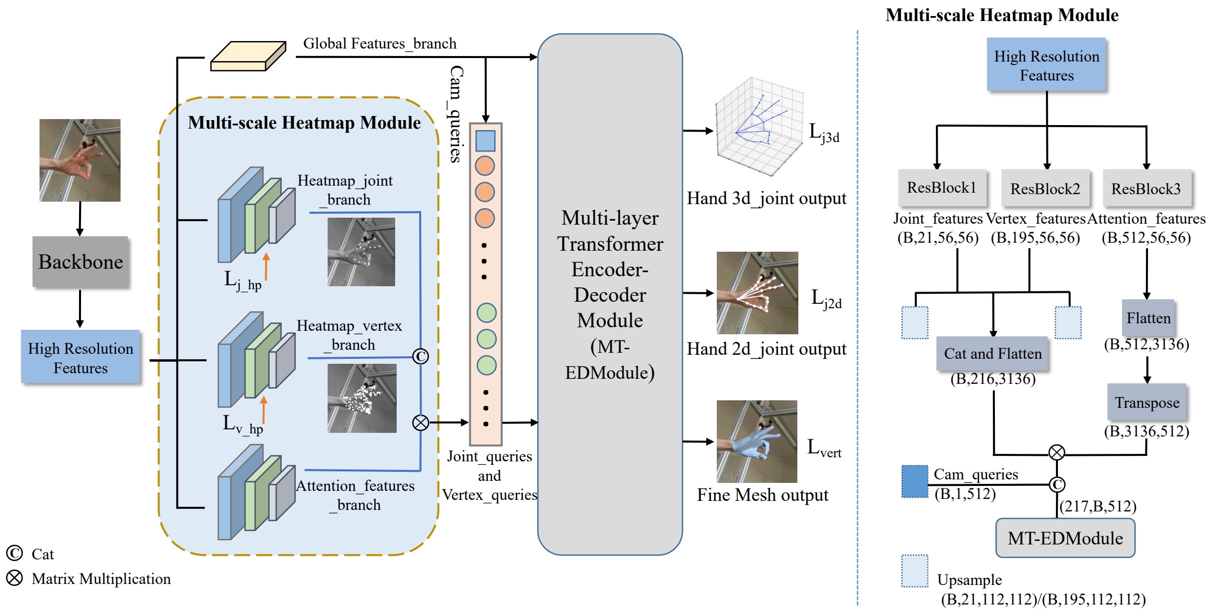


Fig. 2. The overall architecture of our AHRNET for 3D hand pose estimation. The light blue background shows the multi-scale heatmap module. The high-resolution feature map obtained by the high-resolution feature extraction module is passed to four different branches, namely the global feature branch, the joint heatmap branch, the vertex heatmap branch and the attention feature branch. The figure on the right shows the multi-scale heatmap module structure. Best viewed in color.

introduces a novel Transformer encoder-decoder architecture, tackling bottlenecks by disentangling the interactions.

While Transformer-based approaches have demonstrated promising results, they still face challenges in effectively handling tasks like hand pose estimation and mesh recovery. In this paper, we propose an **A**ttention and **H**eatmap-based **R**egressor (AHRNet) for hand pose estimation and mesh recovery. We also construct a simple game scene scenario to test and demonstrate the effectiveness and efficiency of our algorithm, as shown in Figure 1. The main contributions can be summarized as follows:

- We propose the new AHRNet for 3D hand pose estimation that has achieved comparable performance to state-of-the-art works on public datasets while offering improved efficiency.
- In the network architecture, we utilize multi-scale heatmaps and multi-layer transformer encoder-decoder modules to enhance performance.
- To improve interaction efficiency, we employ a high-resolution feature extraction module to greatly boost the algorithm's efficiency, enabling better virtual interaction for the user.

2. OUR APPROACH

The 3D hand pose estimation task based on RGB images $I \in \mathbb{R}^{H\times W}$ aims to estimate the 3D joint positions $P \in \mathbb{R}^{J\times 3}$ of the hand in the camera coordinate system from a given RGB image, where H and W denote the height and width of the input hand image, respectively, and J is the total number of hand joints, usually 21. Accurately estimating the relative depth coordinates of each finger joint in an RGB image is a challenging task.

Regression-based methods first extract joint-related features and detection-based methods utilize convolutional neural networks to generate heatmaps from extracted features and accurately obtain hand joint positions. Our proposed method combines regression-based and detection-based approaches. First, we use the regression method to predict the 3D vertex coordinates and 3D joint coordinates of the hand, and then project them into the 2D image respectively through the camera parameters (scale, translation-x, translation-y). At the same time, we use multi-scale heatmaps (vertex heatmap and joint heatmap) to supervise these two projections and obtain features related to the projection from the features extracted from the input RGB image. This feature is used to further predict the 3D vertices and 3D joints of the hand. The details of the proposed method are as shown in Figure 2, which mainly consists of three modules: the high-resolution feature extraction module, the multi-scale heatmap module, and the multi-layer transformer encoder-decoder module.

2.1. High-Resolution Feature Extraction Module

Although frameworks such as Swin-Transformer [12] could extract plausible feature representation, however, as the number of layers increases, the resolution of the feature continuously decreases. Experimental analysis shows that the high-resolution image features help the Transformer to regress the 3D coordinates of the human joints and mesh vertices. In order to obtain high-resolution feature maps, we need to aggregate different-resolution features from different layers. Specifically, we upsample the low-resolution image features from each module to obtain high-resolution features, which are then summed with the high-resolution features from the previous module to obtain fused high-resolution features. Next, we process these fused features to obtain the final high-resolution features.

2.2. Multi-scale Heatmap Module

Through the high-resolution feature extraction module, we obtain global features and high-resolution features, which are then passed to the vertex heatmap branch, joint heatmap branch, and attention feature branch. The structures of these branches are shown on the right side of Figure 2. We employ the Resblock [13] module to process the high-resolution features. Firstly, the obtained high-resolution feature map (size $D \times \frac{H}{4} \times \frac{H}{4}$, $D = 64$) is input to the Resblock module. For the vertex heatmap branch, the vertex number N_1 in the Resblock module is set to 195, representing the number of 3D hand vertices. For the joint heatmap branch, the vertex N_2 is set to 21, representing the number of 3D hand joints. The dimension of the attention map feature N_3 is set to 512.

Through this network, we obtain the vertex heatmap feature $f_v \in N_1 \times \frac{H}{4} \times \frac{H}{4}$ (N_1 = 195), the joint heatmap feature $f_j \in N_2 \times \frac{H}{4} \times \frac{H}{4}$ (N_2 = 21), and the attention map feature $f_a \in N_3 \times \frac{H}{4} \times \frac{H}{4}$ (N_3 = 512). Next, the vertex heatmap features and joint heatmap features are concatenated and multiplied with the transposed attention map feature matrix by the Softmax layer to obtain two-scale heatmap features f_s. The features are used as vertex queries and joint queries for subsequent Transformers. Additionally, the vertex heatmap feature is transformed into a feature of dimension $(1, B, 512)$ through a fully connected layer, serving as the initial weak-perspective camera parameter query (cam-queries).

2.3. Multi-layer Transformer Encoder-Decoder Module

Inspired by the progressive dimension reduction scheme in [10] and the novel transformer encoder-decoder architecture in [11], we have designed a progressive dimension reduction transformer encoder-decoder architecture for regressing the final 3D hand vertices and 3D joints. We use the global features as the Key and Value in the decoder, and the features f_s as the queries in the decoder. By incorporating the cross-attention module in the decoder, we can effectively capture the non-local relationships between hand joints and mesh vertices. Ultimately, we obtain the joint features $X_j \in \mathbb{R}^{N_2 \times 2D}$ and vertex features $X_v \in \mathbb{R}^{N_1 \times 2D}$ together. Additionally, we employ the progressive attention masking scheme proposed in [14], which focuses more on the vertices and joints within different distance thresholds during the learning process.

2.4. Loss Function

This method employs five loss functions for training, including vertex loss L_{vert}, 3D joint loss L_{j3d}, 2D joint loss L_{j2d}, and vertext heatmap loss L_{v_hp} and joint loss L_{j_hp}. These five loss functions have been introduced and utilized in previous research [10, 11, 14]. The vertex loss L_{vert} uses the L_1 loss function to calculate the difference between predicted vertices and ground truth vertices. The 3D joint loss L_{j3d} uses the mean squared error (MSE) loss function to calculate the difference between predicted joints and ground truth joints, as well as the difference between the 3D joints regressed from predicted vertices and the ground truth joints. The 2D joint loss L_{j2d} uses the L_1 loss function to calculate the difference between the projected 2D joint points of predicted vertices using camera intrinsic parameters and the ground truth 2D joint points. It also calculates the difference between the projected 2D joint points of predicted 3D joints using camera parameters and the ground truth 2D joint points. To calculate the difference, the vertex heatmap loss L_{v_hp} utilizes binary cross-entropy loss and Dice loss [14]. The calculation process for L_{j_hp} is the same as described above. In the end, our overall loss function is defined as follows:

$$\begin{aligned} L_{total} =& W_{vert} \times L_{vert} + W_{j3d} \times L_{j3d} + W_{j2d} \times L_{j2d} \\ &+ W_{v\_hp} \times L_{v\_hp} + W_{j\_hp} \times L_{j\_hp} \end{aligned} \tag{1}$$

where W_{vert}, W_{j3d}, W_{j2d}, W_{v_hp}, W_{j_hp} are the balance factors, which are set to 0.01, 0.1, 0.01, 0.5, and 0.5, respectively.

3. EXPERIMENTS AND DISCUSSIONS

3.1. Implementation details

All the experiments of our AHRNet are trained using the Adam optimizer with a batch size of 8 on Geforce 4090 GPUs by PyTorch. We record the model parameters saved by PyTorch for METRO [10], Fast METRO [11] and our AHRNet. The size of METRO and Fast METRO are 230.4M and 153.0M, respectively, and our model is 52.3M. Hence, the proposed modules do not bring substantial parameter increments to the compared models. For training time, METRO and Fast METRO take around 1.05×10^3 ms and 0.9×10^3 ms per iteration, and our AHRNet is 0.72×10^3 ms, while for inference time, METRO and Fast METRO take around 0.51×10^2 ms and 0.46×10^2 ms per iteration under batch size 1, and our AHRNet is about 0.3×10^2 ms.

3.2. Comparison with state-of-the-art methods

To demonstrate the effectiveness of the proposed AHRNet, we make comparisons with state-of-the-art methods and report the results on the FreiHand dataset in Table 1. Similar

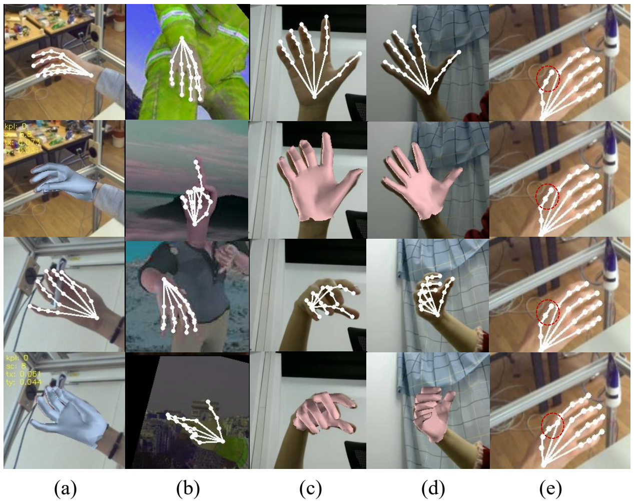


Fig. 3. (a) is the visualization of the 3D joint points and meshes of the Freihand dataset; (b) is the visualization of the two-dimensional joint points of the RHD dataset; (c) is the visualization of hand images captured by ourselves; to gain a more comprehensive understanding of our method, we conduct additional experiments with complex backgrounds, varying lighting as shown in (d); (e) gives the qualitative results with and without these three modules corresponding to Table 3, from top to bottom gives the result with A_1, (b) is with A_2, (c) is with A_2 + G, and (d) is our AHRNet. The detail improvement please refer to the red dash circle.

Table 1. Performance comparison on the FreiHand dataset with state-of-the-arts.

Methods	FPS	Params	FLOPs	PA-MPJPE↓	F@15 mm
FreiHAND [18]	–	–	–	–	0.935
Pose2Mesh [19]	–	–	–	7.7	0.969
I2LMeshNet [20]	–	–	–	7.4	0.973
METRO [10]	19.55	183.8M	41.47G	6.8	0.981
FastMETRO [11]	21.88	133.9M	30.56G	6.5	0.982
Our AHRNet-cls-S12	42.02	27.68M	33.07G	7.2	0.978
Our AHRNet-cls-S24	32.29	37.24M	34.69G	6.7	0.981

Table 2. Performance comparison on RHD dataset and DO with state-of-the-arts.

Methods	AUC of PCK↑	AUC of PCK↑
Zhang et al [21]	0.901	0.825
Rong [22]	0.934	–
Cui-GCN [23]	0.933	0.77
Gu [24]	0.936	–
Our AHRNet	0.943	0.942

Table 3. Ablation study on FreiHand dataset. A_1 denotes the multi-layer transformer encoder-decoder module, A_2 denotes the high-resolution feature extraction module, G denotes the multi-scale heatmap module.

	Methods	PA-MPJPE↓	F@15 mm
a.	Baseline	7.24	0.978
b.	Baseline + A_1	7.06	0.979
c.	Baseline + A_2	7.13	0.979
d.	Baseline + A_2 + G	6.86	0.980
e.	Baseline + A_1 + A_2 + G	6.74	0.981

to [15], we increase the number of blocks to boost the performance. The last two rows represent small and large models respectively. From Table 1, it can be observed that our model obtains 6.7 PA-MPJPE. The comparison results reveal that our model achieves nearly identical performance but at a higher frame per second (FPS). We also provide the comparative results of the RHD and DO dataset in Table 2. While some methods outperform our proposed approach in terms of performance [16, 17], they either fail to acquire high-quality hand mesh data or suffer from slow processing speeds, making them inadequate for the requirements of interactive tasks in virtual environments. We believe that our AHRNet can extract plausible and powerful features, and also be more efficient. Comparative experiments on the FreiHand, RHD and DO datasets verify the superiority of our method.

3.3. Ablation studies

For a deeper understanding of the proposed three modules, we provide quantitative and qualitative ablation studies with and without these modules on the FreiHand dataset in Table 3 and Fig. 3(e). Since the multi-scale heatmap and multi-layer transformer encoder-decoder module these two modules are used in a multi-level manner, we add them into the baseline one by one. Finally, based on the multi-scale heatmap module, this paper achieves the best accuracy and performance. To verify AHRNet is workable and beneficial, we also conduct qualitative experiments as shown in Figure 3.

4. CONCLUSION

In this paper, we have proposed a new attention and heatmap-based network AHRNet for 3D hand pose estimation and mesh recovery from monocular RGB images. The core of our framework is the incorporation of multi-scale heatmaps and attention masks at multiple network stages. The heatmaps provide 2D spatial constraints to guide 3D pose regression, while the attention masks model complex vertex relationships to handle occlusions. By focusing computational resources on key image regions, these modules greatly improve efficiency and enable real-time performance. Through experiments on public benchmarks like FreiHand, we have demonstrated that our method achieves comparable state-of-the-art performance in accuracy while being 1.5x more efficient.

Acknowledgements

This work is supported by National Natural Science Foundation of China (62102208), Beijing Natural Science Foundation (4232023), R&D Program of Beijing Municipal Education Commission (KM202310009002) and supported in part by BeiHang University Yunnan Innovation Institute Yunding Technology Plan(2021) of Yunnan Provincial Key R&D Program(202103AN080001-003).

5. REFERENCES

[1] James M Rehg and Takeo Kanade, "DigitEyes: Vision-based hand tracking for human-computer interaction," in *IEEE Workshop on MNAO*. IEEE, 1994, pp. 16–22.

[2] Tsung-Han Tsai, Chih-Chi Huang, and Kung-Long Zhang, "Design of hand gesture recognition system for human-computer interaction," *Multimedia tools and applications*, vol. 79, pp. 5989–6007, 2020.

[3] Seungryul Baek, Kwang In Kim, and Tae-Kyun Kim, "Pushing the envelope for RGB-based dense 3d hand pose estimation via neural rendering," in *CVPR*, 2019, pp. 1067–1076.

[4] Yana Hasson, Gul Varol, Dimitrios Tzionas, Igor Kalevatykh, Michael J Black, Ivan Laptev, and Cordelia Schmid, "Learning joint reconstruction of hands and manipulated objects," in *CVPR*, 2019, pp. 11807–11816.

[5] Nikos Kolotouros, Georgios Pavlakos, and Kostas Daniilidis, "Convolutional mesh regression for single-image human shape reconstruction," in *CVPR*, 2019, pp. 4501–4510.

[6] Shangchen Han, Beibei Liu, Randi Cabezas, Christopher D Twigg, Peizhao Zhang, Jeff Petkau, Tsz-Ho Yu, Chun-Jung Tai, Muzaffer Akbay, Zheng Wang, et al., "MEgATrack: monochrome egocentric articulated hand-tracking for virtual reality," *TOG*, pp. 87–1, 2020.

[7] André Zenner and Antonio Krüger, "Estimating detection thresholds for desktop-scale hand redirection in virtual reality," in *IEEE VR*. IEEE, 2019, pp. 47–55.

[8] Ankur Handa, Karl Van Wyk, Wei Yang, Jacky Liang, Yu-Wei Chao, Qian Wan, Stan Birchfield, Nathan Ratliff, and Dieter Fox, "DexPilot: Vision-based teleoperation of dexterous robotic hand-arm system," in *ICRA*, 2020, pp. 9164–9170.

[9] Alexey Dosovitskiy, Lucas Beyer, and etc. Kolesnikov, Alexander, "An image is worth 16×16 words: Transformers for image recognition at scale," in *ICLR*, 2020.

[10] Kevin Lin, Lijuan Wang, and Zicheng Liu, "End-to-end human pose and mesh reconstruction with transformers," in *CVPR*, 2021, pp. 1954–1963.

[11] Junhyeong Cho, Kim Youwang, and Tae-Hyun Oh, "Cross-attention of disentangled modalities for 3d human mesh recovery with transformers," in *ECCV*, 2022.

[12] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo, "Swin transformer: Hierarchical vision transformer using shifted windows," in *ICCV*, 2021, pp. 10012–10022.

[13] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun, "Deep residual learning for image recognition," in *CVPR*, 2016, pp. 770–778.

[14] Jeonghwan Kim, Mi-Gyeong Gwon, Hyunwoo Park, Hyukmin Kwon, Gi-Mun Um, and Wonjun Kim, "Sampling is matter: Point-guided 3d human mesh reconstruction," in *CVPR*, 2023, pp. 12880–12889.

[15] Ce Zheng, Xianpeng Liu, Guo-Jun Qi, and Chen Chen, "POTTER: Pooling attention transformer for efficient human mesh recovery," in *CVPR*, 2023, pp. 1611–1620.

[16] Dominik Kulon, Riza Alp Guler, Iasonas Kokkinos, Michael M Bronstein, and Stefanos Zafeiriou, "Weakly-supervised mesh-convolutional hand reconstruction in the wild," in *CVPR*, 2020, pp. 4990–5000.

[17] Yuxiao Zhou, Marc Habermann, Weipeng Xu, Ikhsanul Habibie, Christian Theobalt, and Feng Xu, "Monocular real-time hand shape and motion capture using multi-modal data," in *CVPR*, 2020, pp. 5346–5355.

[18] Nikos Kolotouros, Georgios Pavlakos, Michael J Black, and Kostas Daniilidis, "Learning to reconstruct 3d human pose and shape via model-fitting in the loop," in *ICCV*, 2019, pp. 2252–2261.

[19] Hongsuk Choi, Gyeongsik Moon, and Kyoung Mu Lee, "Pose2mesh: Graph convolutional network for 3d human pose and mesh recovery from a 2d human pose," in *ECCV*. Springer, 2020, pp. 769–787.

[20] Gyeongsik Moon and Kyoung Mu Lee, "I2l-meshnet: Image-to-lixel prediction network for accurate 3d human pose and mesh estimation from a single RGB image," in *ECCV*. Springer, 2020, pp. 752–768.

[21] Xiong Zhang, Qiang Li, Hong Mo, Wenbo Zhang, and Wen Zheng, "End-to-end hand mesh recovery from a monocular RGB image," in *ICCV*, 2019, pp. 2354–2364.

[22] Yu Rong, Takaaki Shiratori, and Hanbyul Joo, "FrankMocap: A monocular 3d whole-body pose estimation system via regression and integration," in *ICCV*, 2021, pp. 1749–1759.

[23] Yuan Cui, Moran Li, Yuan Gao, Changxin Gao, Fan Wu, Hao Wen, Jiwei Li, and Nong Sang, "Camera distance helps 3d hand pose estimated from a single RGB image," *Graphical Models*, vol. 127, pp. 101179, 2023.

[24] Kerui Gu, Linlin Yang, Michael Bi Mi, and Angela Yao, "Bias-compensated integral regression for human pose estimation," *TPAMI*, vol. 45, pp. 10687–10702, 2023.