LEARNING MONOCULAR 3D HUMAN POSE ESTIMATION WITH SKELETAL INTERPOLATION

Ziyi Chen[1]*, Akihiro Sugimoto*[2]*, Shang-Hong Lai*[1]

[1]National Tsing Hua University, Taiwan, [2]National Institute of Informatics, Japan

ABSTRACT

Deep learning has achieved unprecedented accuracy for monocular 3D human pose estimation. However, current learning-based 3D human pose estimation still suffers from poor generalization. Inspired by skeletal animation, which is popular in game development and animation production, we put forward an simple, intuitive yet effective interpolation-based data augmentation approach to synthesize continuous and diverse 3D human body sequences to enhance model generalization. The Transformer-based lifting network, trained with the augmented data, utilizes the self-attention mechanism to perform 2D-to-3D lifting and successfully infer high-quality predictions in the qualitative experiment. The quantitative result of cross-dataset experiment demonstrates that our resulting model achieves superior generalization accuracy on the publicly available dataset.

Index Terms— Data augmentation, skeletal interpolation, transformer, 3D human pose estimation

1. INTRODUCTION

Monocular human pose estimation [1] is the process of estimating joint position in a given image or video, which has recently received increasing attention due to its wide application in many areas, such as virtual reality and sports performance analysis. Current learning-based 3D human pose estimation models can be roughly divided into two types of popular architectures: one-stage model and two-stage model. One-stage models [2, 3] simply map perceived intensities to 3D pose representation, whose generalization ability suffers from the limited variation of indoor training pairs. Two-stage models [4, 5] estimate the 3D pose by first estimating the 2D pose from image appearance and then lifting the 2D pose by modeling the relationship between 3D skeletons and their 2D projections. Abundant in-the-wild images with 2D pose annotations can be utilized for easing the generalization dilemma in the first stage, which makes the two-stage approach increasingly popular. However, two-stage models still face a tricky problem in the second stage: poor generalization. Training sequences from the existing indoor datasets have limited variations, which lead to poor generalization for in-the-wild data. Evolution-based data augmentation [6] synthesized new-fashioned 3D human poses for developing better generalization. However, the discrete generated data cannot enable deep networks to leverage the temporal information and thus brings slight improvement for generalization.

Based on the above observations, we propose a simple yet effective data augmentation method, termed *InterAug*, for 3D human pose estimation by adopting skeletal interpolation. This augmentation strategy explicitly synthesizes enriched variations of body movement from existing 3D human skeletons, which import massive temporal information and greatly enhance the data diversity for pose estimator to learn body movement coherence and improve generalization capability jointly. We use Transformer [7, 8] as a geometric lifting function and demonstrate the feasibility and effectiveness of self-attention mechanism in the task of 2D-to-3D pose lifting.

The main contributions of this paper are summarized as follows: 1) We develop a 3D pose data augmentation method based on the skeletal interpolation technique. This method is able to synthesize abundant and reasonable 3D body movement, which effectively expands the training domain and thus enhance model generalization; 2) To our best knowledge, we are the first to investigate the continuous pose augmentation for training multi-frame 3D human pose estimation model; 3) With the powerful regularization benefited from the proposed augmentation approach, our model achieves superior generalization ability and outperforms most of the existing methods in the cross-dataset experiment.

2. INTERAUG

This section illustrates how the InterAug works. Given a training set consisting of $\mathbf{P} = \{\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \ldots, \mathbf{p}_n\}$, each 3D pose $\mathbf{p}_i \in \mathbb{R}^{3N_j}$ can be represented as the combination of N_j joints, *i.e.* $\mathbf{p}_i = \{\mathbf{j}_i^n\}_{n=1}^{N_j}$, where N_j denotes the number of body joints and $\mathbf{j}_i^n$ represents the n-th joint location in 3D world space. We can further obtain bone vectors $\vec{\mathbf{b}}$ by referencing the skeleton hierarchy of human body. The relationship between $\mathbf{j}_i^n$ and $\vec{\mathbf{b}}_i^n$ is defined as

$$\vec{\mathbf{b}}_i^n = child\left(\mathbf{j}_i^n\right) - parent\left(\mathbf{j}_i^n\right), \tag{1}$$

where $parent()$ and $child()$ denote the parent and child of the joint, and they are defined from the human joint tree.

2.1. Local coordinate transformation

In order to manipulate human pose for convenience, we need to convert bone vectors in world space to the local coordinate system [9, 6]. The local bone vector $\vec{\boldsymbol{\beta}}$ can be computed by the following equation:

$$\vec{\boldsymbol{\beta}}_i = \mathbf{R}_i \, \vec{\mathbf{b}}_i, \tag{2}$$

where $\mathbf{R}_i \in \mathbb{SO}_3$ indicates the rotation transformation of local coordinate system attached to the corresponding parent joint of bone $\vec{\mathbf{b}}_i$. It can be obtained by first computing the basis vector around hip joint and neck joint. The local bone vector $\vec{\boldsymbol{\beta}} \in \mathbb{R}^{3\times(N_j-1)}$ can be represented as $(r^n, \theta^n, \phi^n)_{n=1}^{N_j-1}$ after being transformed into the spherical coordinates, where θ and ϕ determine the bone orientation and r is the bone length. This transformation allows more flexibility in the selection of the different human subjects during the interpolation process since the limitation caused by the skeleton size $(r^n)_{n=1}^{N_j-1}$ can be eliminated.

2.2. Skeletal interpolation

Since there is no need to consider the skeletal size of the human body, two poses are randomly selected from the training set as our source pose and target pose. We can calculate the corresponding local bone vector $\vec{\boldsymbol{\beta}}_s$ and $\vec{\boldsymbol{\beta}}_t$ with Equation 2, where $\vec{\boldsymbol{\beta}}_s, \vec{\boldsymbol{\beta}}_t \in \mathbb{R}^{3\times(N_j-1)}$.

Interpolation processing. Spherical linear interpolation (SLERP) is used to generate vector between $\vec{\boldsymbol{\beta}}_s$ and $\vec{\boldsymbol{\beta}}_t$ on the unit sphere. Equation 3 and Figure 1 show how $\vec{\boldsymbol{\beta}}_{itpl}$ is interpolated.

$$\vec{\boldsymbol{\beta}}_{itpl}(T) = \frac{\sin\left[(1-T)\,\Omega\right]}{\sin\Omega}\vec{\boldsymbol{\beta}}_s + \frac{\sin\left[T\Omega\right]}{\sin\Omega}\vec{\boldsymbol{\beta}}_t, \tag{3}$$

where Ω is the angle between $\vec{\boldsymbol{\beta}}_s$ and $\vec{\boldsymbol{\beta}}_t$, and $T \in [0, 1]$ denotes the interpolation transition from the source orientation to the target orientation.

After rotating the orientation of the bone, the next step is to interpolate the position of the hip joint in world space linearly. This is because the translation of root joint can drive the whole body to translate. The linear interpolation process is defined as:

$$\mathbf{j}_{itpl}^{hip}(T) = \mathbf{j}_s^{hip} + \left(\mathbf{j}_t^{hip} - \mathbf{j}_s^{hip}\right) T, \tag{4}$$

where $\mathbf{j}_s^{hip}$ and $\mathbf{j}_t^{hip}$ denote the hip joint of source pose and target pose.

Validation function. To avoid generating unreasonable 3D human poses, such as those beyond the joint-angle limits, [9] explored a wide range of human postures and construct statistics of joint-angle limits. We utilize their statistics to validate the generated pose. Here, the pose validation function

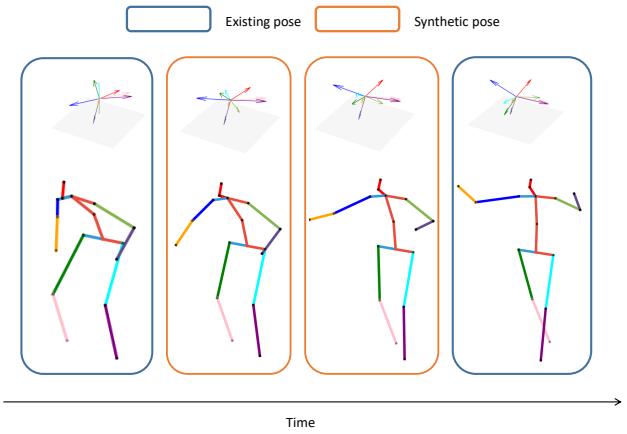

Fig. 1: Visualization of local bone vectors and interpolation process. Existing poses are indicated by blue boxes and synthesized poses are indicated by orange boxes. The upper and lower limbs are marked with different colors, corresponding to the same color vector in the local space.

$Valid\,(\cdot)$ is defined to check whether θ and ϕ are beyond the valid range by applying Equation 5:

$$Valid\left(\vec{\boldsymbol{\beta}}_{itpl}\right) = \begin{cases} 1, \forall\,(\theta^n, \phi^n)_{n=1}^{N_j-1} \in [valid] \\ 0, \exists\,(\theta^n, \phi^n)_{n=1}^{N_j-1} \notin [valid] \end{cases} \tag{5}$$

If the interpolated pose cannot satisfy the validation function, the new target pose will be re-sampled from the training set.

Postprocessing. To further generate 2D-3D pose pairs for training our pose estimator, we obtain 2D joint coordinate from the corresponding 3D pose by using perspective projection, *i.e.*

$$\mathbf{x} = \boldsymbol{M}\mathbf{p}, \tag{6}$$

where $\mathbf{x}$ denotes the 2D joint location in image plane and the camera projection matrix $\boldsymbol{M}$ is provided by the dataset.

Implementation details. During the augmentation process, we randomly sample a pose as our initial pose. N_{kf} keyframes are further selected after the initial pose to form a single training sequence. The time step between keyframes N_{ts} is set to 20. N_{kf} is set to 50. The skeleton size of the initial pose is saved and applied to all the remaining poses. After the augmentation process, the synthetic sequences will be added to the original dataset to form a larger dataset. The total number of synthetic poses is the same as the total number of poses in the original training set. Generating such an amount of data takes about one hour on a regular PC.

3. LIFTING NETWORK

Our 2D-to-3D pose lifting network consists of two fully connected layers and N_b Transformer encoders. The keypoint matrix $(\bar{\mathbf{x}}_1, \bar{\mathbf{x}}_2, \ldots, \bar{\mathbf{x}}_{N_f}) \in \mathbb{R}^{N_f\times 2N_j}$, which is obtained

by concatenating continuous N_f normalized 2D poses, is mapped to input embedding $\mathbf{X} \in \mathbb{R}^{N_f \times N_d}$ by the first fully connected layer. Here N_f denotes the number of input frames along the temporal dimension and N_d is the embedding dimension. Note that we do not apply the positional encoding in Transformer, which is used in the standard Transformer [7]. The input embedding $\mathbf{X}$ enters the Transformer encoder, which consist of multi-head self attention (MSA) module, layer normalization [10], and feed forward network (FFN). The second fully connected layer, designed for dimension reduction, outputs predicted 3D pose $\mathbf{Y} \in \mathbb{R}^{N_f \times 3(N_j-1)}$. In our experiment, we set the hyperparamaters of our model architecture as: $N_f = 5, N_d = 512, N_h = 2, N_b = 6$. N_h and N_b are the head number and block number respectively.

3.1. Transformer encoder

Multi-head self attention. The MSA module first maps input embedding $\mathbf{X}$ identically to query $\mathbf{Q} \in \mathbb{R}^{N_f \times N_d}$, key $\mathbf{K} \in \mathbb{R}^{N_f \times N_d}$ and value $\mathbf{V} \in \mathbb{R}^{N_f \times N_d}$. Then, $\mathbf{Q}$, $\mathbf{K}$ and $\mathbf{V}$ are divided into N_h heads, $\mathbf{Q}_i, \mathbf{K}_i, \mathbf{V}_i \in \mathbb{R}^{N_f \times \frac{N_d}{N_h}}$, each of which is subsequently linearly transformed by its following fully connected layer. The embedding dimension N_d should be divisible by the number of head N_h. Assuming $\mathbf{Q}_1, \mathbf{K}_1, \mathbf{V}_1 \in \mathbb{R}^{N_f \times \frac{N_d}{N_h}}$ represent the query, key, and value in head 1, respectively. The operation in head 1 is defined as:

$$\mathbf{Y}_1 = Softmax\left(\frac{\mathbf{Q}_1 \cdot \mathbf{K}_1^T}{\sqrt{N_d}}\right)\mathbf{V}_1 \tag{7}$$

where $\mathbf{Y}_1 \in \mathbb{R}^{N_f \times \frac{N_d}{N_h}}$ denotes the output of head 1. For other heads, the same procedure are performed equally as Equation 7. All the output of different heads are concatenated, and further linearly transformed, as the output of the MSA module.

Feed forward network. FFN consists of two fully connected layers with $Gelu$ [11] activation function. The simple process is defined as

$$FFN(x) = FC_2\left(Gelu\left(FC_1\left(x\right)\right)\right) \tag{8}$$

where $FC_i(x)$ are linear transformations of the form $\mathbf{W}_i \cdot x + \mathbf{b}_i$.

Overall process. Eventually, we can express the whole encoding process as follows:

$$\begin{aligned} \mathbf{X}^A &= LN(MSA(\mathbf{X}) + \mathbf{X}) \\ Encoder(\mathbf{X}) &= LN(FFN(\mathbf{X}^A) + \mathbf{X}^A) \end{aligned} \tag{9}$$

where LN indicates the layer normalization.

3.2. Loss function

We train our model by minimizing the L2 distance between all the output branches and their corresponding ground truths to enforce the model to learn not only the joint positions, but also the skeleton size consistency and movement coherence. The loss function L is defined by:

$$L = \sum_{i=0}^{N_f} \alpha_i \left\|\mathbf{Y}^i - \mathbf{P}_{GT}^i\right\|_2, \tag{10}$$

where $\mathbf{P}_{GT}$ denotes the ground truth and α_i is a hyperparameter. For the prediction at the middle of receptive field and other output poses, α_i equals 1 and 0.01, respectively. AdamW optimizer is used with linear warm-up and cosine scheduler. We set the learning rate to $8e-5$ and training epoch to 40.

Table 1: The comparison between our method and other existing methods on cross-dataset evaluation in terms of MPJPE, PCK, and AUC. † indicates less joints are included. * indicates equal experimental setup.

Method	MPJPE	PCK	AUC
Wang *et al.* (ICCVw'19) [12]	–	71.2	33.8
Habibie *et al.* (CVPR'19) [13]	127.0	69.6	35.5
Yang *et al.* (CVPR'18) [14]	–	69.0	32.0
Zeng *et al.* (ECCV'20) [15]	–	77.6	43.8
Kanazawa *et al.* (CVPR'18) [16]	113.2	77.1	40.7
Chen *et al.* (CVPR'19) [17]	–	61.4	29.4
Gong *et al.* (CVPR'21) [18] †	**73.0**	**88.6**	**57.3**
Ours (with InterAug) *	93.4	81.6	48.2
Li *et al.* (CVPR'20) [6] *	99.7	81.2	46.1
Ours (without InterAug) *	102.6	79.2	44.4
Ours (with InterAug) *	**93.4**	**81.6**	**48.2**

4. EXPERIMENTS

Our experiments aim to answer the following questions, *i.e.* 1) Does InterAug really help us to boost the generalization? 2) Can our model improve the estimation performance under the cross-dataset setup? 3) What is the best hyperparameter of model architecture? 4) Does the Transformer architecture really help us to boost the accuracy?

4.1. Cross-dataset experiment

We execute the cross-dataset experiment to quantitatively validate generalization capability. During the optimization process, we adopt the pose data of S1, S5, S6, S7, S8 of Human3.6m (H36M) [19] dataset for model training. The MPI-INF-3DHP (3DHP) [20] dataset is used for evaluation. Note that 3DHP is a more challenging 3D pose dataset compared to H36M and we do not use 3DHP's training set to train our model but directly use its test set to demonstrate the model generalization. The mean per-joint position error (MPJPE), Percentage of Correct Keypoints (PCK) under 150mm radius and Area Under the Curve (AUC) are employed in the cross-dataset evaluation.

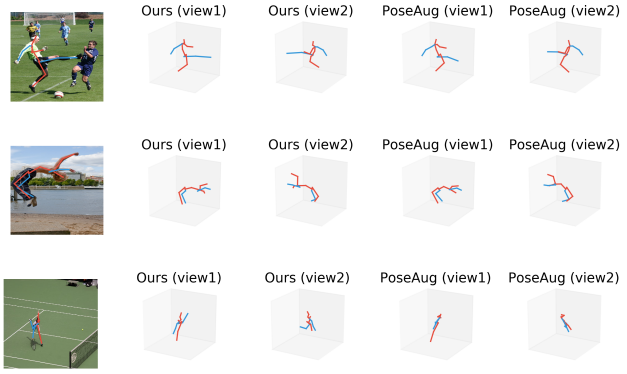

Fig. 2: Qualitative evaluation on in-the-wild dataset comparing with [18]. Please zoom in for better observation.

Quantitative experiment. Experimental result as shown in Table 1 shows that our method provides second-best generalization performance with 93.4mm MPJPE and 81.6% PCK and 48.2% AUC. We surpass all of the existing methods except [18]. However, the current state-of-the-art method, PoseAug [18] deletes the nose joint to bridge datasets' gap since H36M has the nose joint while 3DHP doesn't. So our method has more numerical errors than [18] caused by the joint definition. Under the same experimental setup, our method outperforms the previous SOTA result [6] by 6.3% MPJPE, 0.2% PCK and 2.1% AUC.

Qualitative experiment. The U3DPW [6] dataset, which contains three hundred high-quality in-the-wild images, is used for the qualitative evaluation purpose. The first and second visualization comparison in Figure 2 shows that both our method and SOTA [18] provide reasonable results and demonstrate superior generalization capability to the unseen poses. However, It is worth noting that our method infers more realistic estimation than [18] from the last case, highlighting our approach's effectiveness and robustness.

4.2. Ablation study

Ablation study on model architecture. The number of blocks N_b, heads N_h, embedding dimension N_d and receptive field N_f are explored in the Table 2. We performs controlled experiments on the different hyperparameters in turn. For example, we change the dimensionality of the hidden layer and fix the rest of the parameters to determine the best dimension based on the quantitative results.

Ablation study on model type. This ablation study explores the different model performance for 2D-to-3D lifting. Existing baseline networks, *e.g.* multi-layer perceptron (MLP) and temporal convolution network (TCN), are explored in Table 3 with regards to the inference time, intra-scenario and extra-scenario performances. All of the models are trained with the same augmented data for a fair comparison.

Experiment results indicate TCN, which obtains the best performance under intra-dataset setup but performs bad on unseen domain, tends to overfit on training domain. Another problem of TCN is that it requires a high computational cost and thus brings more computing time (18.35ms). Instead of overfitting to the training domain, Transformer generalizes best with competitive computation time. Furthermore, Although MLP performs the fastest inference, its prediction accuracy is much worse than others.

Table 2: Ablation study on the dimensionality of hidden layers, multi-attention heads, encoder blocks, and receptive fields in Transformer. Inputs are ground truth 2D keypoints provided by dataset Human3.6m, without adding position encoding.

Dim. N_d	Head N_b	Block N_b	Frames N_f	MPJPE	PCK	AUC
256	2	6	5	94.9	81.6	46.6
512	2	6	5	**93.4**	**81.6**	**48.2**
768	2	6	5	95.8	81.3	46.7
512	**2**	6	5	**93.4**	81.6	**48.2**
512	4	6	5	97.2	80.8	46.1
512	8	6	5	94.3	**81.8**	47.4
512	2	2	5	99.9	78.9	45.2
512	2	4	5	96.5	80.3	46.3
512	2	**6**	5	**93.4**	**81.6**	**48.2**
512	2	6	3	95.6	80.8	46.6
512	2	6	**5**	**93.4**	**81.6**	**48.2**
512	2	6	9	97.3	81.1	45.9

Table 3: Model comparison on H36M and 3DHP using the same augmented data. BS means batch size.

-		H36M	3DHP		
Method	CPU(BS=1)	MPJPE	MPJPE	PCK	AUC
MLP(frame=5) [21]	**5.31**ms	45.6	102.2	78.7	42.7
TCN(frame=243) [22]	18.35ms	**33.9**	148.2	59.8	30.4
Ours(frame=5)	7.13ms	37.0	**93.4**	**81.6**	**48.2**

5. CONCLUSION

In this paper, we present skeletal interpolation as a novel data augmentation approach for 3D human pose estimation. This augmentation strategy allows to synthesize rich and continuous body movements, which greatly increases variations of human pose data and effectively alleviates projection ambiguity by considering the temporal continuity of body movement. Extensive experimental results show that Transformer-based model trained with augmented poses provides superior generalization capability on cross-dataset evaluation.

Limitation Although the proposed method is simple yet effective, there still remain some improvable points. The SLERP algorithm has the limitation regarding the ability to generate realistic poses and go beyond the pose domain given by the training set. Complex augmentation methods such as introducing nonlinear curves in body movement or using GAN-based techniques to cross the training domain boundary deserve further study. Moreover, developing the Transformer architecture for lifting process is also worth trying. These possible solutions will remain our future work.

6. REFERENCES

[1] Xiaopeng Ji, Qi Fang, Junting Dong, Qing Shuai, Wen Biao Jiang, and Xiaowei Zhou, "A survey on monocular 3d human pose estimation," *Virtual Real. Intell. Hardw.*, vol. 2, pp. 471–500, 2020.

[2] Sijin Li and Antoni B. Chan, "3d human pose estimation from monocular images with deep convolutional neural network," in *ACCV*, 2014.

[3] Georgios Pavlakos, Xiaowei Zhou, K. Derpanis, and Kostas Daniilidis, "Coarse-to-fine volumetric prediction for single-image 3d human pose," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1263–1272, 2017.

[4] Bugra Tekin, Pablo Márquez-Neila, M. Salzmann, and P. Fua, "Learning to fuse 2d and 3d image cues for monocular body pose estimation," *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 3961–3970, 2017.

[5] Ching-Hang Chen and D. Ramanan, "3d human pose estimation = 2d pose estimation + matching," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5759–5767, 2017.

[6] Shichao Li, Lei Ke, K. Pratama, Yu-Wing Tai, C. Tang, and K. Cheng, "Cascaded deep monocular 3d human pose estimation with evolutionary training data," *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6172–6182, 2020.

[7] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, L. Kaiser, and Illia Polosukhin, "Attention is all you need," *ArXiv*, vol. abs/1706.03762, 2017.

[8] Ce Zheng, Sijie Zhu, Matias Mendieta, Taojiannan Yang, Chen Chen, and Zhengming Ding, "3d human pose estimation with spatial and temporal transformers," *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2021.

[9] Ijaz Akhter and Michael J. Black, "Pose-conditioned joint angle limits for 3d human pose reconstruction," *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1446–1455, 2015.

[10] Jimmy Ba, J. Kiros, and Geoffrey E. Hinton, "Layer normalization," *ArXiv*, vol. abs/1607.06450, 2016.

[11] Dan Hendrycks and Kevin Gimpel, "Gaussian error linear units (gelus)," *arXiv: Learning*, 2016.

[12] Luyang Wang, Yan Chen, Z. Guo, Keyuan Qian, Mude Lin, Hongsheng Li, and J. Ren, "Generalizing monocular 3d human pose estimation in the wild," *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, pp. 4024–4033, 2019.

[13] I. Habibie, Weipeng Xu, Dushyant Mehta, Gerard Pons-Moll, and C. Theobalt, "In the wild human pose estimation using explicit 2d features and intermediate 3d representations," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10897–10906, 2019.

[14] Wei Yang, Wanli Ouyang, X. Wang, J. Ren, Hongsheng Li, and Xiaogang Wang, "3d human pose estimation in the wild by adversarial learning," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5255–5264, 2018.

[15] Ailing Zeng, X. Sun, F. Huang, Minhao Liu, Qiang Xu, and Stephen Lin, "Srnet: Improving generalization in 3d human pose estimation with a split-and-recombine approach," in *ECCV*, 2020.

[16] A. Kanazawa, Michael J. Black, D. Jacobs, and Jitendra Malik, "End-to-end recovery of human shape and pose," *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7122–7131, 2018.

[17] Xipeng Chen, Kwan-Yee Lin, Wentao Liu, Chen Qian, X. Wang, and L. Lin, "Weakly-supervised discovery of geometry-aware representation for 3d human pose estimation," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10887–10896, 2019.

[18] Kehong Gong, Jianfeng Zhang, and Jiashi Feng, "Poseaug: A differentiable pose augmentation framework for 3d human pose estimation," in *CVPR*, 2021.

[19] Catalin Ionescu, Dragos Papava, V. Olaru, and C. Sminchisescu, "Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, pp. 1325–1339, 2014.

[20] Dushyant Mehta, H. Rhodin, D. Casas, P. Fua, Oleksandr Sotnychenko, Weipeng Xu, and C. Theobalt, "Monocular 3d human pose estimation in the wild using improved cnn supervision," *2017 International Conference on 3D Vision (3DV)*, pp. 506–516, 2017.

[21] J. Martinez, Rayat Hossain, J. Romero, and J. Little, "A simple yet effective baseline for 3d human pose estimation," *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 2659–2668, 2017.

[22] Dario Pavllo, Christoph Feichtenhofer, David Grangier, and M. Auli, "3d human pose estimation in video with temporal convolutions and semi-supervised training," *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7745–7754, 2019.