3D Human Pose Estimation with Two-step Mixed-Training Strategy

Yingfeng Wang[1] Zhengwei Wang[1] Muyu Li[1,3,*] Hong Yan[1,2]
[1]Centre for Intelligent Multidimensional Data Analysis, Science Park, Hong Kong
[2]City University of Hong Kong
[3]Dalian University of Technology, China
yingfeng@innocimda.com, zhengwei@innocimda.com, muyuli@dlut.edu.cn, h.yan@cityu.edu.hk

Abstract

In monocular 3D human pose estimation, target motions are generally stable and continuous, which indicates that joint velocity can provide valuable information for better estimation. Therefore, it is critical to learn the joint motion trajectory and spatio-temporal information from velocity. Previous works have shown that Transformers are effective in capturing the relationship between tokens. However, in practice, only 2D position is available and 3D velocity has not been explicitly used as a model input. To address this challenge, we propose TMT (Two-step Mixed-Training strategy), a transformer-based approach that effectively incorporates 3D velocity into the input vector during training, allowing for better learning of relevant features in the shallow layers. Extensive experiments demonstrate that TMT significantly improves the performance of state-of-the-art models, such as MixSTE, MHFormer, and PoseFomer, on two datasets: Human3.6M and MPI-INF-3DHP. TMT outperforms the state-of-the-art approach by up to 13.8% on the Human3.6M dataset.

1. Introduction

The primary objectives of human pose estimation are to localize joints and provide a representation of the human body from pictures or videos. Current 3D human pose estimation (3DHPE) methods can be classified into two types. The first type predicts 3D estimations directly from raw images [20, 21]. The second type, known as 2D-to-3D lifting approaches [3, 16, 34], elevates the provided 2D estimation results to 3D positions. Generally, the second method performs better than the first method due to the two-stage inference process with state-of-the-art 2D pose detectors [4, 26]. In TMT, the focus is on these lifting methods. 3D human pose estimation has a wide range of applications, such as human-robot interaction [28] and action recognition [36, 38].

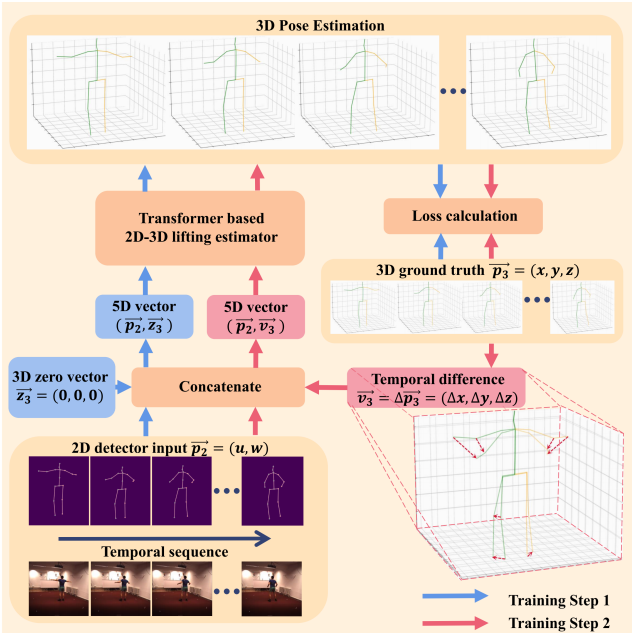


Figure 1. The overview of the TMT. Blue and red arrows show the data flow of training steps 1 and 2 respectively. The joints' velocity vectors comes from the 3D target position difference. Inference only uses step 1 with a zero vector.

However, the mapping from 2D to 3D is not unique, which is different for subjects with different body shapes. To alleviate depth ambiguity and occlusion, considering the success of Transformer [31], Zheng *et al*. [40] model the spatial correlations among all joints in a frame and the temporal correlations among consecutive frames. It takes a video as input and predicts the 3D pose estimation for the central frame. Compared with PoseFormer [40] one-time spatial and temporal encoding, MixSTE [37] is proposed to utilize spatial and temporal blocks alternately to

* Muyu Li is the corresponding author. This work is funded by Hong Kong Innovation and Technology Commission (InnoHK Project CIMDA).

obtain better spatio-temporal feature encoding. Additionally, MixSTE extends the output from the central frame to the entire sequence of input video, significantly reducing computation as the sequence length increases. With a seq2seq model, MixSTE achieves accurate and fast estimation simultaneously.

In previous studies, the 3D velocity vector has played various roles, primarily in loss functions and additional smoothing models. In addition to innovating the 2D-to-3D lifting model structures, SmoothNet [35] utilizes velocity and acceleration to mitigate the jitters in 3DHPE by incorporating an additional stage into the lifting output. SmoothNet significantly enhances the temporal smoothness of existing pose estimators by effectively capturing the long-term temporal relations of each joint, while disregarding the noisy correlations among joints. In [1, 9], velocity vectors are employed as a loss function during training to enforce temporal smoothness constraints, ensuring temporal consistency across sequences.

However, current methods have not considered training neural networks to effectively capture 3D velocity features in the shallow layers. This highlights the need to explicitly incorporate 3D velocity vectors as feature inputs into the model. In practice, the velocity can be approximated by calculating the position difference between adjacent frames. From a statistical perspective, our experiments demonstrate a positive correlation between the Mean Per Joint Position Error (MPJPE) for each joint and the norm of its velocity vector, as illustrated in Fig. 2. Both intuition and statistics suggest that exploring the potential connections between joint position and velocity is worthwhile. To address this, transformer-based models, which have been proven to effectively capture relationships between tokens [37, 40], are employed to capture these connections.

Further experiments have demonstrated that 3D velocity vectors outperform 2D vectors as input features. Unlike 2D velocity, which is ambiguous and corresponds to multiple solutions, 3D velocity contains sufficient information to accurately describe 3D motion. Additionally, the absence of 3D velocity during inference has motivated us to develop a new training strategy that incorporates this valuable feature as input. As shown in Tab. 7, it has been proven that solely using 3D velocity as input during training, while excluding it during inference, can actually detriment the generalization of models. This leads to inconsistency between the information in the training data and the inference data.

To address the aforementioned issues, we propose a solution called TMT. TMT consists of two training steps and its structure is depicted in Fig. 1. We conducted experimental evaluations on two widely used human pose estimation benchmarks, namely Human3.6M [10] and MPI-INF-3DHP [19]. The results demonstrate that TMT achieves state-of-the-art performance when using ground truth as the 2D input. Our contributions can be summarized as follows:

- TMT is the first approach to propose a training strategy that directly incorporates the 3D velocity vector as input, without requiring it during inference. By employing TMT, we effectively enhance the model accuracy with minimal overhead.
- TMT introduces the synthesis of 2D keypoints with varying qualities, enabling us to explore the impact of input quality using synthesized data.
- Our approach achieves state-of-the-art performance on both Human3.6M and MPI-INF-3DHP datasets, with losses reduced to 18.6 mm and 48.3 mm respectively.

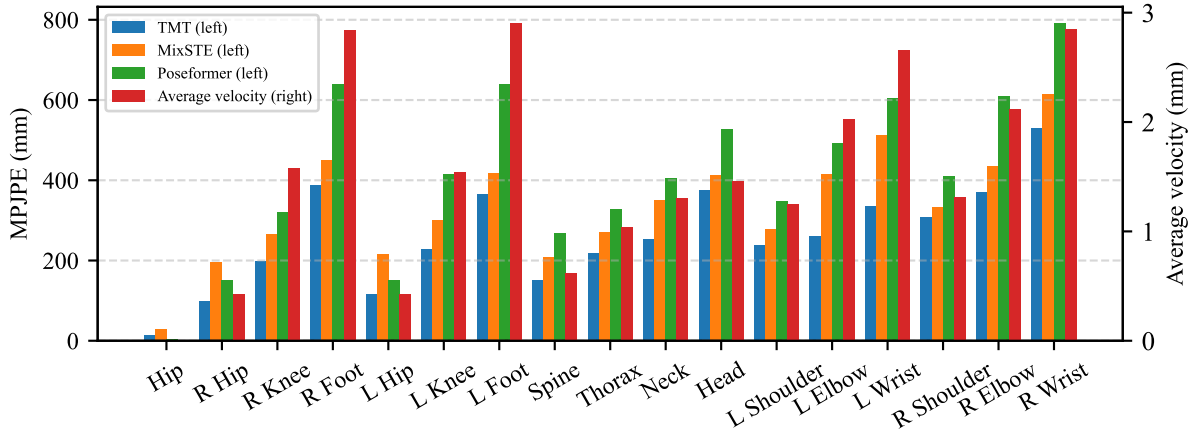
2. Related Work

2.1. 3D human pose estimation

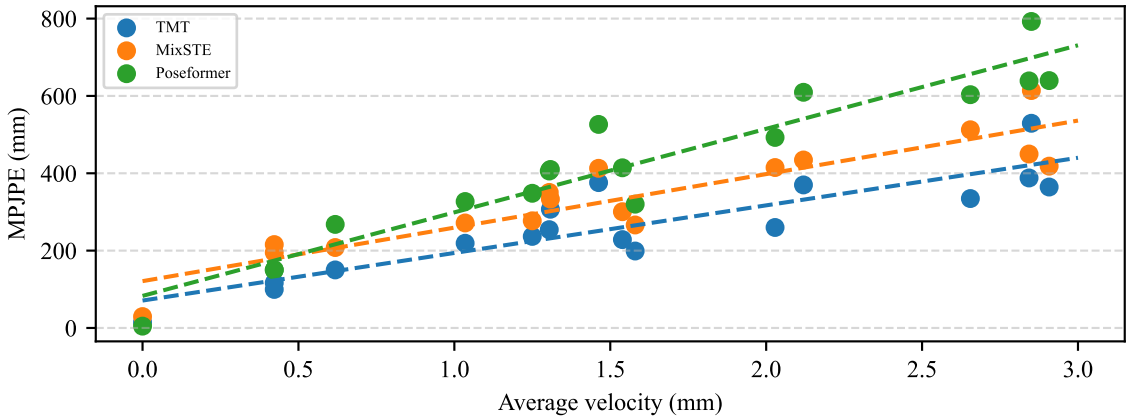
3D human pose estimation can be divided into monocular and multi-view methods based on perspective data. Our work focuses on and summarizes 3D monocular methods. There are two main approaches used in monocular methods: direct estimation (end-to-end manner) and 2D-to-3D lifting. The direct estimation approach [22, 27, 29] estimates the 3D pose coordinates directly from the raw input, without relying on 2D pose coordinates. However, due to the advancements in reliable 2D keypoints detection methods [2, 4, 26], the 2D-to-3D lifting approach [5, 15, 17, 39], which utilizes the 2D pose coordinates to estimate the 3D pose coordinates, has shown better performance. In our proposed strategy, TMT, we also adopt the 2D-to-3D lifting method in the estimation process.

Based on these 2D intermediate representation, lots of works focus on the lifting process. For a single frame, Martinez *et al*. [18] utilize residual network to regress 3D human pose estimation from 2D keypoints. Furthermore, to capture the correlations in temporal domain and improve the accuracy, different network architectures [1, 7, 30] have been proposed. These methods use 2D keypoints sequence from videos to generate 3D human pose estimations. Long-Short-Term-Memory (LSTM) provides another way to explore temporal correlations in the 2D keypoints sequence. Hossain and Little [9] proposed a recurrent network that uses LSTM to explore the correlation of the 2D sequence in the temporal domain. Currently, some works propose to focus on the correlation in the spatial-temporal domains rather than just the temporal domain. In the spatial-temporal domain, features such as bone length [6] and symmetry [13] are considered to improve performance.

For 3D human pose estimation, the motion trajectory plays a crucial role in learning the relationship between joints. The learning of motion trajectory is divided into two parts: the time domain and the space domain. The transformer can efficiently discover the relationship between



(a) Total MPJPE (mm) and average velocity (mm) of each joint on the test set.



(b) The correlation of total MPJPE and average velocity.

Figure 2. The comparison of total MPJPE of each joint and average velocity.

each token through data-driven optimization. Currently, the use of the transformer in 3D human pose estimation can be categorized into two approaches: modeling joints simultaneously in time and space within a continuous frame, and modeling joints within a continuous frame separately in the time domain and the space domain. So far, the second approach generally performs better. Therefore, we have chosen three state-of-the-art models, namely MixSTE [37], PoseFormer [40], and MHFormer [12], for evaluation.

PoseFormer [40] is the first to propose a highly competitive convolution-free transformer network, in contrast to the previous state-of-the-art models that rely on CNNs. In order to effectively capture local joint correlations, PoseFormer utilizes two separate transformers for spatial and temporal information. The input to PoseFormer consists of a 2D sequence of joints, and the output is the 3D position of the central frame.

MixSTE [37] builds upon the work of PoseFormer by introducing seq2seq transformer architectures and making further improvements. In order to capture both spatial dependencies and temporal motion in an input sequence, MixSTE incorporates a spatial transformer block (STB) and a temporal transformer block (TTB), achieving promising results. Initially, the input is projected into a high-dimensional feature space with dimension d_m. To preserve location information in both the spatial and temporal domains, MixSTE utilizes a location embedding matrix. The basic block of MixSTE employs the STB and TTB in an alternating manner to learn spatial and temporal correlations. Finally, a regression head is employed to concatenate the output of the TTB and reduce the dimensionality from d_m to 3, generating the desired output format.

MHFormer [12] effectively models multi-hypothesis dependencies and establishes strong relationships among hypothetical features, enabling it to learn spatio-temporal representations of multiple plausible pose hypotheses. The task can be divided into three parts: (1) generating multiple initial hypothesis representations; (2) facilitating communication and merging among the multiple self-hypotheses to form a unified representation, which is subsequently partitioned into multiple diverging hypotheses; (3) merging multiple hypotheses to obtain the final 3D pose.

2.2. Usage of velocity in 3D human pose estimation

SmoothNet [35] utilizes velocity and acceleration information to refine the output of the 3D estimator and enhance temporal smoothness. The velocity and acceleration values are obtained by calculating the differences between successive 3D position estimations. These values are then used in an additional network to eliminate jitters in the output. In contrast to SmoothNet, our approach employs the velocity derived from the 3D ground truth to directly train the estimator, without the need for an extra network. This eliminates the requirement for additional parameters and reduces computational complexity. As shown in Table 4, our method, TMT, achieves improved accuracy without significant computation and storage overhead.

In the works [1, 9], a derivative loss is employed to enforce temporal smoothness in the estimation of joints' velocity. The 21 joints are divided into three parts: palm root, finger mid, and finger terminal. The derivative loss is then utilized during the backpropagation process to improve the temporal smoothness of the joint velocity estimation.

Based on the analysis and comparison mentioned earlier, it has been observed that the MPJPE is positively correlated with the velocity norm. However, existing methods do not directly consider the velocity vector as an input. While it is possible to derive the 3D velocity vector from ground truth during training, it is not generally available for inference. In order to address this limitation, we propose the TMT method.

3. Approach

Our strategy involves two steps in the training process: training steps 1 and 2. The difference between these two steps lies in the input contents. In step 1, we merge the 2D position estimation from detectors with the 3D zero vector to create a 5D vector, which serves as the input for the Transformer-based 3D estimator. In step 2, we combine the 2D position estimation with the 3D velocity vector, derived from the differentiation of the 3D ground truth, to obtain a 5D vector input.

3.1. Training and inference process

As shown in Fig. 1, the network takes 5D concatenation, which containing 2D key point coordinates $P_2 \in \mathbb{R}^{N\times T\times 2}$ with N joints and T frames, as input. There are two training steps, Training step 1 and 2. First, 2D keypoint coordinates are padded with zero vector $Z_3 \in \mathbb{R}^{N\times T\times 3}$ into $P_{Z3} \in \mathbb{R}^{N\times T\times 5}$. In the training step 2, 2D keypoint coordinates P_2 and 3D joints velocity vector $V_3 \in \mathbb{R}^{N\times T\times 3}$ are merged into $P_{V3} \in \mathbb{R}^{N\times T\times 5}$. P_{Z3} and P_{V3} are the inputs of TMT. 3D prediction sequence $\overline{P_3} \in \mathbb{R}^{N\times T\times 3}$ is the output of Transformer models. The inference process is the same as the training step 1, using the P_{Z3} as input. The calculation formula of V_3 is as follows:

$$V_{3(i,j)} = [P_{3(i,j)} - P_{3(i-1,j)}]/\Delta T \tag{1}$$

$V_{3(i,j)}$ represents the velocity vector of j-th joint in i-th frame , where $i \in [2, T]$ and $j \in [1, N]$. The length of the $V_{3(i,j)}$ is $T-1$. In order to ensure the consistency of the length of the input sequence, $V_{3(1,j)}$ is set to 0. P_3 denotes the ground truth 3D positions.

ΔT is set to one for simplicity. Since ΔT is assumed to be the same over the entire video input sequence, it becomes a redundant scaling factor and can actually be omitted. With the additional 3D velocity information, TMT generates better 3D prediction sequence. Under TMT, The model parameters are updated in step 1 and step 2 alternatively.

There are three reasons for this approach. The first one, as the goal of TMT, is to add the 3D velocity vector into the input as a new characteristic. The second is forcing the model to fit the input data from two distribution with different input structure. The connections for velocity are dropped out when padding zero, preventing from overfitting to velocity vector [25]. The third reason is to follow the inference requirement. Because the 3D velocity vectors are not available for input of lifting process in application, unlike the training process with two steps, the inference process has only step 1, that is, using 2D position estimation and 3D zero vector as input. As justification, choosing zero vector mimics the idea of zero padding, with the fact that the average velocity is typically zero. Also, using zero vector follows the regularization of dropout method, while regularly alternative padding makes them different.

To handle the increased dimensionality, the input dimensions are modified from 2D keypoints to 5D composed vectors. This modification can be achieved by resetting the model parameters of the first layer. By adjusting the input dimensions, the model can effectively handle the increased dimensionality of the composed vectors.

3.2. Synthesize 2D keypoints with various qualities

The quality of the 2D keypoints input has a significant impact on the effectiveness of TMT. To investigate this impact, we utilize CPN as the 2D detector and generate 2D keypoints of varying quality by gradually adjusting the CPN keypoints towards the 2D ground truth. This allows us to explore the influence of 2D keypoints input quality on TMT. The calculation formula is as follows:

$$P_{2(i,j)} = CPN_{(i,j)} + m \times (GT_{(i,j)} - CPN_{(i,j)}) \tag{2}$$

$m \in [0, 1]$ represents the proximity of the final 2D keypoints to the 2D GT. $P_{2(i,j)} \in \mathbb{R}^2$ indicates the final 2D keypoints of j-th joint in i-th frame under m. $CPN_{(i,j)} \in \mathbb{R}^2$ represents the original 2D keypoints of j-th joint in i-th frame from CPN. $GT_{(i,j)} \in \mathbb{R}^2$ represents 2D GT of j-th joint in i-th frame.

Protocol 1 (MPJPE, GT)	Dir.	Disc.	Eat	Greet	Phone	Photo	Pose	Pur.	Sit	SitD.	Smoke	Wait	WalkD.	Walk	WalkT.	Avg.
Hossain & Little [9]	35.2	40.8	37.2	37.4	43.2	44.0	38.9	35.6	42.3	44.6	39.7	39.7	40.2	32.8	35.5	39.2
Liu *et al*. [16] (T=243)	34.5	37.1	33.6	34.2	32.9	37.1	39.6	35.8	40.7	41.4	33.0	33.8	33.0	26.6	26.9	34.7
Wang *et al*. [32] (T=96)	23.0	25.7	22.8	22.6	24.1	30.6	24.9	24.5	31.1	35.0	25.6	24.3	25.1	19.8	18.4	25.6
PoseFormer [40] (T = 81)	30.0	33.6	29.9	31.0	30.2	33.3	34.8	31.4	37.8	38.6	31.7	31.5	29.0	23.3	23.1	31.3
MHFormer [12] (T = 9)	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	36.6
MHFormer [12] (T = 27)	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	34.3
P-STMO [24] (T = 243)	28.5	30.1	28.6	27.9	29.8	33.2	31.3	27.8	36.0	37.4	29.7	29.5	28.1	21.0	21.0	29.3
MixSTE [37] (T=81)	25.6	27.8	24.5	25.7	24.9	29.9	28.6	27.4	29.9	29.0	26.1	25.0	25.2	18.7	19.9	25.9
MixSTE [37] (T=243)	21.6	22.0	20.4	21.0	20.8	24.3	24.7	21.9	26.9	24.9	21.2	21.5	20.8	14.7	15.7	21.6
TMT (T=9,MHFormer-based)	30.5	34.9	33.4	33.1	36.0	41.3	35.5	32.8	41.3	44.3	35.2	35.5	35.6	27.4	28.9	35.0
TMT (T=27,MHFormer-based)	31.6	35.3	33.8	32.3	33.9	36.8	35.3	32.5	39.9	41.2	32.9	33.6	32.6	25.6	26.8	33.6
TMT (T=81,MixSTE-based)	24.6	25.3	22.5	23.3	23.7	28.2	26.5	26.1	26.6	30.0	24.2	23.7	23.8	16.8	18.5	24.2
TMT (T=243,MixSTE-based)	**18.7**	**18.2**	**18.7**	**18.1**	**18.8**	**21.5**	**19.8**	**18.3**	**23.8**	**23.4**	**18.8**	**17.0**	**17.7**	**12.7**	**13.9**	**18.6**

Table 1. The comparison results on Human3.6M under Protocol 1 (no rigid alignment applied) using 2D ground truth as the input. The highlighted numbers in bold are the best results of different motions.

Protocol 1 (MPJPE, 2D Detectors)	Dir.	Disc.	Eat	Greet	Phone	Photo	Pose	Pur.	Sit	SitD.	Smoke	Wait	WalkD.	Walk	WalkT.	Avg.
Pavlakos *et al*. [21]	48.5	54.4	54.4	52.0	59.4	65.3	49.9	52.9	65.8	71.1	56.6	52.9	60.9	44.7	47.8	56.2
Cai *et al*. [1](CPN, T=7)	44.6	47.4	45.6	48.8	50.8	59.0	47.2	43.9	57.9	61.9	49.7	46.6	51.3	37.1	39.4	48.8
Pavllo *et al*. [23]	45.2	46.7	43.3	45.6	48.1	55.1	44.6	44.3	57.3	65.8	47.1	44.0	49.0	32.8	33.9	46.8
Yeh *et al*. [33]	44.8	46.1	43.3	46.4	49.0	55.2	44.6	44.0	58.3	62.7	47.1	43.9	48.6	32.7	33.3	46.7
Lin [14](T=50)	42.5	44.8	42.6	44.2	48.5	57.1	52.6	41.4	56.5	64.5	47.4	43.0	48.1	33.0	35.1	46.6
Liu *et al*. [16] (T=243)	41.8	44.8	41.1	44.9	47.4	54.1	43.4	42.2	56.2	63.6	45.3	43.5	45.3	31.3	32.2	45.1
Wang *et al*. [32] (CPN, T=96)	40.2	42.5	42.6	41.1	46.7	56.7	41.4	42.3	56.2	60.4	46.3	42.2	46.2	31.7	31.0	44.5
Chen *et al*. [3] (CPN, T=243)	41.4	43.5	40.1	42.9	46.6	51.9	41.7	42.3	53.9	60.2	45.4	41.7	46.0	31.5	32.7	44.1
PoseFormer [40] (T=9)	45.4	48.6	45.9	50.1	50.6	59.7	48.3	44.8	59.0	62.9	50.6	48.6	52.8	37.8	41.3	49.9
PoseFormer [40] (CPN, T=81)	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	44.3
MHFormer [12] (CPN, T=9)	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	47.8
MHFormer [12] (CPN, T=27)	42.2	45.0	42.6	43.9	48.6	56.2	43.1	41.3	57.6	64.2	46.8	43.3	46.9	33.0	35.1	45.9
MHFormer [12] (CPN, T=81)	41.1	45.2	41.2	43.1	45.6	52.7	42.2	42.5	54.4	61.3	45.1	42.8	46.9	31.4	33.1	44.5
MixSTE [37] (CPN,T=81)	39.8	43.0	38.6	40.1	43.4	50.6	40.6	41.4	52.2	56.7	43.8	40.8	43.9	29.4	30.3	42.4
MixSTE [37] (CPN,T=243)	37.6	40.9	**37.3**	**39.7**	42.3	**49.9**	40.1	39.8	**51.7**	55.0	42.1	39.8	**41.0**	**27.9**	27.9	40.9
TMT (CPN,T=9,PoseFormer-based)	45.4	48.6	45.9	50.1	50.6	59.7	48.3	44.8	59.0	62.9	50.6	48.6	52.8	37.8	41.3	49.8
TMT (CPN,T=9,MHFormer-based)	43.8	46.6	45.7	46.8	49.5	58.8	44.6	43.6	58.2	66.5	48.5	45.8	50.1	34.5	36.8	48.0
TMT (CPN,T=27,MHFormer-based)	42.0	44.7	42.5	43.7	48.2	55.7	42.9	41.0	57.2	64.0	46.6	43.0	46.6	32.9	34.7	45.7
TMT (CPN,T=81,MHFormer-based)	41.2	45.2	41.2	42.7	45.5	52.8	42.3	42.2	54.3	61.5	45.5	42.7	46.8	31.2	33.0	44.5
TMT (CPN,T=81,MixSTE-based)	39.5	43.1	38.7	39.8	43.1	50.3	40.3	41.8	51.8	56.8	43.4	50.8	44.0	29.1	30.0	42.2
TMT (CPN,T=243,MixSTE-based)	**37.3**	**40.5**	37.6	40.4	**40.6**	50.4	**38.5**	**39.6**	51.8	**53.8**	**41.9**	**38.8**	41.2	28.1	**27.5**	**40.5**
Protocol 2 (P-MPJPE, 2D Detectors)	**Dir.**	**Disc.**	**Eat**	**Greet**	**Phone**	**Photo**	**Pose**	**Pur.**	**Sit**	**SitD.**	**Smoke**	**Wait**	**WalkD.**	**Walk**	**WalkT.**	**Avg.**
Hossain & Little [9]	35.7	39.3	44.6	43.0	47.2	54.0	38.3	37.5	51.6	61.3	46.5	41.4	47.3	34.2	39.4	44.1
Pavlakos *et al*. [21]	34.7	39.8	41.8	38.6	42.5	47.5	38.0	36.6	50.7	56.8	42.6	39.6	43.9	32.1	36.5	41.8
Cai *et al*. [1] (CPN,T=7)	35.7	37.8	36.9	40.7	39.6	45.2	37.4	34.5	46.9	50.1	40.5	36.1	41.0	29.6	32.3	39.0
Liu *et al*. [16] (T=243)	32.3	35.2	33.3	35.8	35.9	41.5	33.2	32.7	44.6	50.9	37.0	32.4	37.0	25.2	27.2	35.6
Wang *et al*. [32] (CPN,T=96)	31.8	34.3	35.4	33.5	35.4	41.7	**31.1**	31.6	44.4	49.0	36.4	32.2	35.0	24.9	23.0	34.5
PoseFormer [40] (T=81)	34.1	36.1	34.4	37.2	36.4	42.2	34.4	33.6	45.0	52.5	37.4	33.8	37.8	25.6	27.3	36.5
Chen *et al*. [3] (CPN,T=243)	32.6	35.1	32.8	35.4	36.3	40.4	32.4	32.3	42.7	49.0	36.8	32.4	36.0	24.9	26.5	35.0
MixSTE [37] (CPN,T=243)	30.8	33.1	30.3	**31.8**	**33.1**	39.1	**31.1**	30.5	42.5	44.5	34.0	30.8	**32.7**	22.1	22.9	32.6
TMT (CPN,T=243,MixSTE-based)	**30.6**	**32.8**	**30.1**	**31.8**	33.2	**38.9**	31.4	**30.3**	**41.8**	**43.7**	**33.9**	**30.5**	33.0	**21.8**	**22.2**	**32.4**
MPJVE (2D Detectors)	**Dir.**	**Disc.**	**Eat**	**Greet**	**Phone**	**Photo**	**Pose**	**Pur.**	**Sit**	**SitD.**	**Smoke**	**Wait**	**WalkD.**	**Walk**	**WalkT.**	**Avg.**
Pavllo *et al*. [23]	3.0	3.1	2.2	3.4	2.3	2.7	2.7	3.1	2.1	2.9	2.3	2.4	3.7	3.1	2.8	2.8
Chen *et al*. [3] (CPN, T=243)	2.7	2.8	2.0	3.1	2.0	2.4	2.4	2.8	1.8	2.4	2.0	2.1	3.4	2.7	2.4	2.5
PoseFormer [40] (T=81)	3.2	3.4	2.6	3.6	2.6	3.0	2.9	3.2	2.6	3.3	2.7	2.7	3.8	3.2	2.9	3.1
MixSTE [37] (CPN,T=243)	2.5	2.7	1.9	**2.8**	**1.9**	2.2	2.3	2.6	1.6	2.2	1.9	2.0	3.1	2.6	2.2	2.3
TMT (CPN,T=243,MixSTE-based)	**2.4**	**2.6**	**1.7**	**2.8**	**1.9**	**2.2**	**2.0**	**2.5**	**1.5**	**2.1**	**1.8**	**1.7**	**3.0**	**2.3**	**2.1**	**2.2**

Table 2. Quantitative comparison (MPJPE,P-MPJPE,MPJVE) under Protocol 1 (no rigid alignment applied) and Protocol 2 (rigid-alignment) on Human3.6M using detected 2D pose estimation as the input. **Top table**:comparison under Protocol 1 (MPJPE); **Middle table**:comparison under Protocol 2 (P-MPJPE); **Bottom table**: comparison of MPJVE. The highlighted numbers in bold are the best results of different motions.

Tab. 8 showcases the influence of 2D keypoints quality. As m approaches one, indicating higher quality 2D keypoints, the accuracy of the trained model grows. It is worth noting that synthesized data is utilized due to the limited availability of input data with varying qualities. Additionally, for better comparison with previous works, which often use CPN and 2D ground truth, we follow this convention in our experiments.

4. Experiments

4.1. Datasets and evaluation

We assess our model's performance on two widely used datasets: Human3.6M and MPI-INF-3DHP.

The **Human3.6M** dataset is considered the primary dataset for indoor 3D human pose estimation. It comprises recordings from four cameras that capture the movements of 11 actors. The dataset includes 17 standard actions, such as walking, talking, smoking, and waiting. For each action, the coordinates of 17 joints on the actor's body, known as keypoints, are recorded. The dataset consists of a total of 3.6 million frames, divided into different sections. All actions are used for both training and testing. Five subjects (S1, S5, S6, S7, S8) are used for training, while two subjects (S9 and S11) are reserved for testing. The evaluation is conducted using Protocol 1 (MPJPE) and Protocol 2 (P-MPJPE), as defined in [23].

MPI-INF-3DHP dataset includes both indoor and outdoor scenes. It consists of recordings from 14 cameras capturing the movements of 8 actors. The dataset includes eight

standard actions. Additionally, MPI-INF-3DHP provides a test set consisting of recordings from 6 subjects, featuring various scenes.

4.2. Implementation details

TMT is a training strategy based on transformers. In the following experiments, we utilized the state-of-the-art model MixSTE as the 2D-3D lifting estimator. Our strategy was implemented using PyTorch. Two RTX 3090ti GPUs were employed for both training and testing. The input for the 2D keypoints consisted of the results from CPN (Cascaded Pyramid Network) [4] and the 2D ground truth. The weight setting in Weighted MPJPE (WMPJPE) was determined based on the joint type.

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

Human 3.6m. The input for TMT and other baseline models consists of the 2D human pose estimation from the CPN network and the ground truth. The specific results for 15 different motions are provided in Tab. 1.. The last column displays the average results across all 15 actions. Notably, the MPJPE (Mean Per Joint Position Error) has been reduced from 21.6 mm to 18.6 mm, which corresponds to a reduction of approximately 13.8%.

For the CPN 2D detector, the test set results of 15 actions (S9 and S11) are presented in Tab. 2. Under Protocol 1, TMT achieves a value of 40.5 mm, while under Protocol 2, it reaches 32.4 mm. The comparison of the total MPJPE for different joints and the 3D target average velocity for each joint on the Human3.6M test set is depicted in Fig. 2a. Additionally, the correlation between the total MPJPE and average velocity is illustrated in Fig. 2b. When comparing limb joints with trunk joints, it is observed that limb joints have a higher average velocity, and the average velocity increases as the joints move farther away from the trunk. Overall, there is a positive relationship between the average velocity of a joint and its MPJPE, indicating a strong correlation between average velocity and MPJPE.

Comparing to GT 2D keypoints, the improvement on CPN input is limited. This observation motivates the exploration on effects of input qualities.

MPI-INF-3DHP. Tab. 3 shows the comparison between TMT and other methods on MPI-INF-3DHP. The ground truth is used as the input. In the Tab. 3, TMT reaches the best results in MPJPE, outperforming the traditional strategy by 12% on average.

4.4. Parameters and FLOPS

In TMT, there is no need for additional model parameters apart from the embedding matrix in the first layer. As a result, the number of parameters in TMT shows almost no increase compared to MixSTE. Table 4 provides a comparison of the parameters and FLOPS (floating point operations per second) between MixSTE and TMT.

Methods (GT)	MPJPE
Pavllo *et al*. [23]	84.8
Lin [14]	79.8
Li *et al*. [11]	99.7
Chen *et al*. [3]	79.1
Wang *et al*. [32]	68.1
Gong *et al*. [8]	73.0
Zheng *et al*. [40]	77.1
MixSTE [37] (T=27)	54.9
TMT (T=27,MixSTE-based)	**48.3**

Table 3. The comparison results of MPJPE on MPI-INF-3DHP with 2D ground truth input. The result in bold is the best result.

T	Parameters (M)		FLOPS (G)	
	MixSTE	TMT	MixSTE	TMT
9	33.66	33.67	10.30	10.30
27	33.67	33.67	30.89	30.90
81	33.70	33.70	92.69	92.70
243	33.78	33.79	278.08	278.10

Table 4. The comparison of parameters and FLOPS between MixSTE and TMT (MixSTE-based).

4.5. Time for convergence

Tab. 5 shows that TMT achieves faster convergence compared to MixSTE when using the same batch size and learning rate. Although TMT takes two training steps in each batch, resulting in approximately double the time for each epoch, the total number of epochs required for convergence in TMT is less than half of that in MixSTE. As a result, the overall time for convergence in TMT is less than that in MixSTE. This indicates that TMT can achieve convergence more efficiently within a shorter training time.

T	Batch	lr	Total time (min)	
			MixSTE	TMT
27	1024	0.00004	2167	1543
81	1024	0.00004	2346	1731
243	1024	0.00004	2205	1652
27	2048	0.00008	1408	1213
81	2048	0.00008	1532	1024
243	2048	0.00008	2101	1557

Table 5. The comparison of time for convergence between MixSTE and TMT (MixSTE-based) under different batchsize (Batch) and learning rate (lr).

4.6. Ablation study

Parameter setting analysis. Tab. 6 shows the effectiveness of different hyper-parameters under protocol 1 with MPJPE on MixSTE. There are three hyper-parameters for the network: the depth of Pose Estimation Block (d_l), the dimension of Pose Estimation Block (d_m), and the input sequence length (T). We divided the settings into three parts.

Each part will decide the best result of one parameter while keeping other parameters fixed. The best settings in the table are in bold font.

d_l	d_m	T	MPJPE
4	64	27	35.9
6	64	27	34.2
8	64	27	33.6
10	64	27	33.7
8	128	27	32.1
8	256	27	31.2
8	**512**	27	30.7
8	600	27	30.9
8	512	64	25.1
8	512	81	24.7
8	512	128	23.0
8	512	**243**	**18.6**
8	512	256	19.5

Table 6. Hyper-parameters setting analysis in d_l, d_m and T with ground truth input on Human3.6M.

Effectiveness of two-step training process. This section discusses the necessity of the two-step training process. Tab. 7 shows the comparison results of TMT, TMT_1 and TMT_2 given the same parameter setting. TMT_1 means that TMT without training step 2, TMT_2 means that TMT without training step 1. TMT reaches the best result.

For TMT_1, due to the lack of velocity feature as input in training step 2, the result performance falls back to the same level with traditional training strategy. And the accuracy is still lower than TMT, probably because the extra zero vectors introduce overfitting problems.

For TMT_2, the inference result is largely degraded compared to the other two, mainly because the data information of training process is inconsistent with that of inference process. Specifically, 2D input is concatenated with 3D velocity when training and then with zero vector for inference, resulting in different input distribution.

d_l	d_m	T	TMT_1	TMT_2	TMT
4	64	27	38.4	69.1	**35.9**
6	64	27	36.7	65.4	**34.2**
8	64	27	34.4	62.3	**33.6**
8	128	27	34.2	58.4	**32.1**

Table 7. The MPJPE (mm) comparison results of TMT, TMT_1 and TMT_2.

Effectiveness of the 2D position input. The quality of the 2D position input plays a significant role in determining the performance of TMT. To investigate the impact of different-quality 2D inputs on the final 3D predictions, we generate inputs of varying qualities by continuously approximating the CPN input to the ground truth (GT). Tab. 8 illustrates the relationship between the input quality of 2D keypoints and the improvement effect of the final 3D prediction at T=243. As the CPN input progressively approaches the GT, the quality of the 2D keypoints improves, resulting in a more substantial improvement effect in the TMT predictions. In other words, the closer the CPN input is to the GT, the better the performance of TMT becomes.

m	MixSTE	TMT	Improvement
0	40.9	40.5	0.98%
0.25	36.45	36.04	1.12%
0.5	29.2	28.8	1.37%
0.75	23.9	23.3	2.51%
0.80	22.7	21.8	3.96%
0.85	22.3	20.9	6.28%
0.90	22.0	20.4	7.27%
0.95	21.8	19.2	11.93%
1	21.6	18.6	13.89%

Table 8. The comparison of MPJPE between different 2D position input (different m).

Effectiveness of different concatenation. In the training step 2, 2D position input can concatenate with 2D velocity vector V_2, and also can concatenate with 3D velocity V_3 and acceleration A_3. The calculation formulas of V_2 and A_3 are described by Eq. (3) and Eq. (4), respectively. In Eq. (3) and Eq. (4), $i \in [2, T]$ and $j \in [1, N]$. P_2 denotes the ground-truth 2D positions. $V_{2(1,j)}$ and $A_{3(1,j)}$ are set to 0 for the same reason as described in Eq. (1).

$$V_{2(i,j)} = [P_{2(i,j)} - P_{2(i-1,j)}]/\Delta T \tag{3}$$

$$A_{3(i,j)} = [V_{3(i,j)} - V_{3(i-1,j)}]/\Delta T \tag{4}$$

TMT_{v2} represents that 2D position input concatenated with V_2 into 4D input. TMT_{a3} means the 2D position input concatenated with A_3 into 5D input. TMT_{va3} means the 2D position input concatenated with V_3 and A_3 into 8D input. Tab. 9 shows that TMT reached the best result. The 2D velocity in TMT_{v2} input lacks 3D information, resulting in a higher value of MPJPE than TMT. Because 3D acceleration has been expressed in the model through 3D velocity information, the 3D acceleration in TMT_{va3} input is redundant, resulting in slightly worse performance.

As expected, the result of TMT_{v2} is worse than TMT_{va3} and TMT, because 2D velocity is the difference of adjacent frames input, providing less useful information. And the accuracy of TMT and TMT_{va3} is almost at the same level, with the same reason that 3D acceleration is just simple linear combination of 3D velocity. It can be observed that the performance of TMT_{va3} is slightly degraded compared to TMT, probably resulting from overfitting problem.

Effectiveness of loss function. The loss function of TMT is exactly the same as the loss function of the selected method. MixSTE is taken as an example to explore the impact of the loss function on the TMT result. Tab. 10 shows the results. Each loss function is used for training process, and MPJPE and MPJVE are recorded after model convergence respectively. WMPJPE apply different weight to

2D position input	T	TMT$_{v2}$	TMT$_{a3}$	TMT$_{va3}$	TMT
CPN	81	42.5	42.6	42.6	**42.2**
CPN	243	41.0	41.1	40.9	**40.5**
GT	81	26.5	25.6	25.1	**24.2**
GT	243	22.1	21.4	19.6	**18.6**

Table 9. The comparison of MPJPE between different concatenation on GT and CPN.

each joint regarding their loss. The MPJPE with WMPJPE training loss is less than that with pure MPJPE loss. However, WMPJPE loss + TCLoss has higher MPJVE value (0.93 mm) than WMPJPE loss + MPJVE loss MPJVE value (0.89 mm).Furthermore, combining WMPJPE, MPJVE and TCLoss shows best performance in both indicator.

Loss function	MPJPE	MPJVE
MPJPE loss	20.5	1.17
WMPJPE loss	20.2	1.03
WMPJPE loss+TCLoss	19.6	0.93
WMPJPE loss+MPJVE loss	19.9	0.89
ours (WMPJPE loss+MPJVE loss+TCLoss)	**18.6**	**0.84**

Table 10. Ablation study for loss functions.

4.7. Qualitative results

In this section, we utilize the S11 dataset from Human3.6M for evaluation purposes. Fig. 3 illustrates the results of our estimation as well as the MixSTE estimation with ground truth input on Human3.6M. It is evident from Fig. 3 that the estimation using the TMT strategy performs better in handling scenes with occlusion.

To visualize the spatial and temporal correlation of the pose estimation, we conducted a visualization of the self-attention weights among joints and sequences. Fig. 4 displays the attention outputs of two heads, allowing us to observe the different correlations among joints and frames. The attention outputs have been normalized to a range of $[0, 1]$. The rows and columns in Fig. 4 represent the queries and predicted outputs, respectively.

5. Conclusion

In this paper, we propose the Two-step Mixed-Training Strategy (TMT), a training strategy for 3D human pose estimation (3DHPE) using a transformer-based model. TMT incorporates the 3D velocity vector as a new input feature to enhance the learning of relevant features in the shallow layers. Additionally, TMT enables models to effectively handle input data from two distributions, resulting in minimal overhead and potentially faster training speeds. Furthermore, TMT introduces a method to obtain 2D keypoints of varying qualities. Experimental results demonstrate that our strategy improves the state-of-the-art results by up to

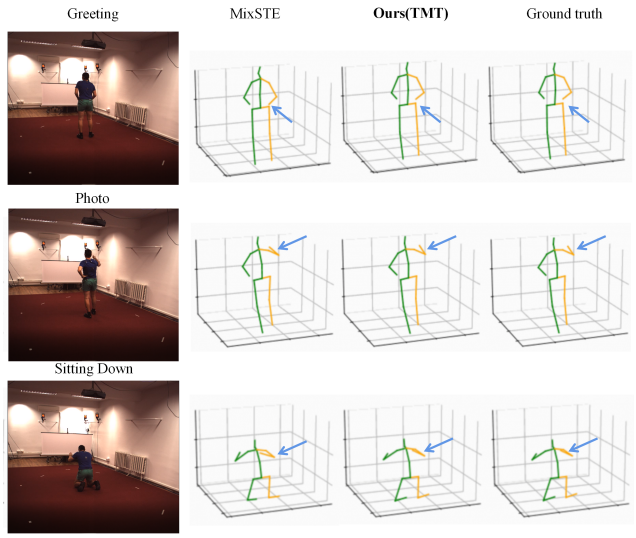


Figure 3. The comparison results of our strategy (TMT) and MixSTE with different actions on Human3.6M. The blue arrows highlight the better result of TMT.

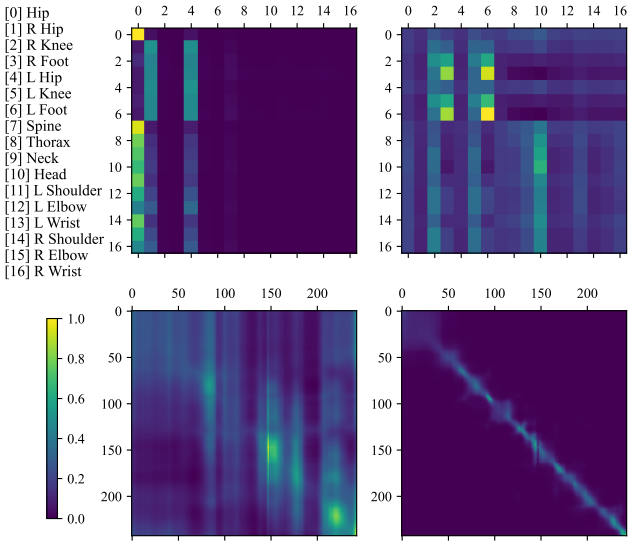


Figure 4. The upper two images show the spatial correlation among joints. The pixel denotes the attention weight $w_{i,j}$ of the j-query for the i-th output where i is the row index and j the column index. The lower two images show the temporal correlation among frames.

13.8% when using ground truth as input. Although we currently concatenate the 3D velocity vector to the input vector in TMT, exploring different methods of including it during the training process is still valuable.

References

[1] Yujun Cai, Liuhao Ge, Jun Liu, Jianfei Cai, Tat-Jen Cham, Junsong Yuan, and Nadia Magnenat Thalmann. Exploiting spatial-temporal relationships for 3d pose estimation via graph convolutional networks. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2272–2281, 2019.

[2] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7291–7299, 2017.

[3] Tianlang Chen, Chen Fang, Xiaohui Shen, Yiheng Zhu, Zhili Chen, and Jiebo Luo. Anatomy-aware 3d human pose estimation with bone-based pose decomposition. *IEEE Transactions on Circuits and Systems for Video Technology*, 32(1):198–209, 2021.

[4] Yilun Chen, Zhicheng Wang, Yuxiang Peng, Zhiqiang Zhang, Gang Yu, and Jian Sun. Cascaded pyramid network for multi-person pose estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7103–7112, 2018.

[5] Hai Ci, Chunyu Wang, Xiaoxuan Ma, and Yizhou Wang. Optimizing network structure for 3d human pose estimation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2262–2271, 2019.

[6] Rishabh Dabral, Anurag Mundhada, Uday Kusupati, Safeer Afaque, Abhishek Sharma, and Arjun Jain. Learning 3d human pose from structure and motion. In *Proceedings of the European conference on computer vision (ECCV)*, pages 668–683, 2018.

[7] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[8] Kehong Gong, Jianfeng Zhang, and Jiashi Feng. Poseaug: A differentiable pose augmentation framework for 3d human pose estimation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8575–8584, 2021.

[9] Mir Rayat Imtiaz Hossain and James J Little. Exploiting temporal information for 3d human pose estimation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 68–84, 2018.

[10] 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.

[11] Shichao Li, Lei Ke, Kevin Pratama, Yu-Wing Tai, Chi-Keung Tang, and Kwang-Ting Cheng. Cascaded deep monocular 3d human pose estimation with evolutionary training data. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6173–6183, 2020.

[12] W Li, H Liu, H Tang, P Wang, and L MHFormer Van Gool. Multi-hypothesis transformer for 3d human pose estimation. arxiv 2021. *arXiv preprint arXiv:2111.12707*.

[13] Zhi Li, Xuan Wang, Fei Wang, and Peilin Jiang. On boosting single-frame 3d human pose estimation via monocular videos. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 2192–2201, 2019.

[14] Jiahao Lin and Gim Hee Lee. Trajectory space factorization for deep video-based 3d human pose estimation. *arXiv preprint arXiv:1908.08289*, 2019.

[15] Kenkun Liu, Rongqi Ding, Zhiming Zou, Le Wang, and Wei Tang. A comprehensive study of weight sharing in graph networks for 3d human pose estimation. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part X 16*, pages 318–334. Springer, 2020.

[16] Ruixu Liu, Ju Shen, He Wang, Chen Chen, Sen-ching Cheung, and Vijayan Asari. Attention mechanism exploits temporal contexts: Real-time 3d human pose reconstruction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5064–5073, 2020.

[17] Xiaoxuan Ma, Jiajun Su, Chunyu Wang, Hai Ci, and Yizhou Wang. Context modeling in 3d human pose estimation: A unified perspective. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6238–6247, 2021.

[18] Julieta Martinez, Rayat Hossain, Javier Romero, and James J Little. A simple yet effective baseline for 3d human pose estimation. In *Proceedings of the IEEE international conference on computer vision*, pages 2640–2649, 2017.

[19] Dushyant Mehta, Helge Rhodin, Dan Casas, Pascal Fua, Oleksandr Sotnychenko, Weipeng Xu, and Christian Theobalt. Monocular 3d human pose estimation in the wild using improved cnn supervision. In *2017 international conference on 3D vision (3DV)*, pages 506–516. IEEE, 2017.

[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 *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part VII 16*, pages 752–768. Springer, 2020.

[21] Georgios Pavlakos, Xiaowei Zhou, and Kostas Daniilidis. Ordinal depth supervision for 3d human pose estimation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7307–7316, 2018.

[22] Georgios Pavlakos, Xiaowei Zhou, Konstantinos G Derpanis, and Kostas Daniilidis. Coarse-to-fine volumetric prediction for single-image 3d human pose. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7025–7034, 2017.

[23] Dario Pavllo, Christoph Feichtenhofer, David Grangier, and Michael Auli. 3d human pose estimation in video with temporal convolutions and semi-supervised training. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7753–7762, 2019.

[24] Wenkang Shan, Zhenhua Liu, Xinfeng Zhang, Shanshe Wang, Siwei Ma, and Wen Gao. P-stmo: Pre-trained spa-

tial temporal many-to-one model for 3d human pose estimation. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part V*, pages 461–478. Springer, 2022.

[25] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958, 2014.

[26] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5693–5703, 2019.

[27] 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.

[28] Mikael Svenstrup, Soren Tranberg, Hans Jorgen Andersen, and Thomas Bak. Pose estimation and adaptive robot behaviour for human-robot interaction. In *2009 IEEE International Conference on Robotics and Automation*, pages 3571–3576. IEEE, 2009.

[29] Bugra Tekin, Artem Rozantsev, Vincent Lepetit, and Pascal Fua. Direct prediction of 3d body poses from motion compensated sequences. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 991–1000, 2016.

[30] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International conference on machine learning*, pages 10347–10357. PMLR, 2021.

[31] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[32] Jingbo Wang, Sijie Yan, Yuanjun Xiong, and Dahua Lin. Motion guided 3d pose estimation from videos. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIII 16*, pages 764–780. Springer, 2020.

[33] Raymond Yeh, Yuan-Ting Hu, and Alexander Schwing. Chirality nets for human pose regression. *Advances in Neural Information Processing Systems*, 32, 2019.

[34] Ailing Zeng, Xiao Sun, Fuyang Huang, Minhao Liu, Qiang Xu, and Stephen Lin. Srnet: Improving generalization in 3d human pose estimation with a split-and-recombine approach. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XIV 16*, pages 507–523. Springer, 2020.

[35] Ailing Zeng, Lei Yang, Xuan Ju, Jiefeng Li, Jianyi Wang, and Qiang Xu. Smoothnet: a plug-and-play network for refining human poses in videos. In *Computer Vision–ECCV 2022: 17th European Conference, Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part V*, pages 625–642. Springer, 2022.

[36] Can Zhang, Tianyu Yang, Junwu Weng, Meng Cao, Jue Wang, and Yuexian Zou. Unsupervised pre-training for temporal action localization tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 14031–14041, 2022.

[37] Jinlu Zhang, Zhigang Tu, Jianyu Yang, Yujin Chen, and Junsong Yuan. Mixste: Seq2seq mixed spatio-temporal encoder for 3d human pose estimation in video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13232–13242, 2022.

[38] Jiaxu Zhang, Gaoxiang Ye, Zhigang Tu, Yongtao Qin, Qianqing Qin, Jinlu Zhang, and Jun Liu. A spatial attentive and temporal dilated (satd) gcn for skeleton-based action recognition. *CAAI Transactions on Intelligence Technology*, 7(1):46–55, 2022.

[39] Long Zhao, Xi Peng, Yu Tian, Mubbasir Kapadia, and Dimitris N Metaxas. Semantic graph convolutional networks for 3d human pose regression. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3425–3435, 2019.

[40] Ce Zheng, Sijie Zhu, Matias Mendieta, Taojiannan Yang, Chen Chen, and Zhengming Ding. 3d human pose estimation with spatial and temporal transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11656–11665, 2021.