3D Human Pose Estimation with Spatial and Temporal Transformers

Ce Zheng[1], Sijie Zhu[1], Matias Mendieta[1], Taojiannan Yang[1], Chen Chen[1], Zhengming Ding[2]
[1]Department of Electrical and Computer Engineering, University of North Carolina at Charlotte
[2]Department of Computer Science, Tulane University
{czheng6, szhu3, mmendiet, tyang30, chen.chen}@uncc.edu; zding1@tulane.edu

Abstract

Transformer architectures have become the model of choice in natural language processing and are now being introduced into computer vision tasks such as image classification, object detection, and semantic segmentation. However, in the field of human pose estimation, convolutional architectures still remain dominant. In this work, we present ***PoseFormer****, a purely transformer-based approach for 3D human pose estimation in videos without convolutional architectures involved. Inspired by recent developments in vision transformers, we design a spatial-temporal transformer structure to comprehensively model the human joint relations within each frame as well as the temporal correlations across frames, then output an accurate 3D human pose of the center frame. We quantitatively and qualitatively evaluate our method on two popular and standard benchmark datasets: Human3.6M and MPI-INF-3DHP. Extensive experiments show that PoseFormer achieves state-of-the-art performance on both datasets.* ***Code is available at*** https://github.com/zczcwh/PoseFormer

1. Introduction

Human pose estimation (HPE) aims to localize joints and build a body representation (*e.g.* skeleton position) from input data such as images and videos. HPE provides geometric and motion information of the human body and can be applied to a wide range of applications (*e.g.* human-computer interaction, motion analysis, healthcare). Current works generally can be divided into two classes: (1) direct estimation approaches, and (2) 2D-to-3D lifting approaches. Direct estimation methods [31, 29] infer a 3D human pose from 2D images or video frames without intermediately estimating the 2D pose representation. 2D-to-3D lifting approaches [25, 5, 43, 38] infer 3D human pose from an intermediately estimated 2D pose. Benefiting from the excellent performance of state-of-the-art 2D pose detectors, 2D-to-3D lifting approaches generally outperform direct estimation methods. However, the mapping of these 2D poses to 3D is non-trivial; various potential 3D poses could be generated from the same 2D pose due to depth ambiguity and occlusion. To alleviate some of these issues and preserve natural coherence, many recent works have integrated temporal information from videos into their approaches. For example, [25, 5] utilize temporal convolutional neural networks (CNNs) to capture global dependencies from adjacent frames, and [33] uses recurrent architectures to similar effect. However, the temporal correlation window is limited for both of these architectures. CNN-based approaches typically rely on dilation techniques, which inherently have limited temporal connectivity, and recurrent networks are mainly constrained to simply sequential correlation.

Recently, the transformer [37] has become the *de facto* model for natural language processing (NLP) due to its efficiency, scalability and strong modeling capabilities. Thanks to the self-attention mechanism of the transformer, global correlations across long input sequences can be distinctly captured. This makes it a particularly fitting architecture for sequence data problems, and therefore naturally extendable to 3D HPE. With its comprehensive connectivity and expression, the transformer provides an opportunity to learn stronger temporal representations across frames. However, recent works [12, 36] show that transformers require specific designs to achieve comparable performance with CNN counterparts for vision tasks. Specifically, they often require either extremely large scale training datasets [12], or enhanced data augmentation and regularization [36] if applied to smaller datasets. Moreover, existing vision transformers have been limited primarily to image classification [12, 36], object detection [4, 50], and segmentation [41, 47], but how to harness the power of transformers for 3D HPE remains unclear.

To begin answering this question, we first directly apply the transformer on 2D-to-3D lifting HPE. In this case, we view the entire 2D pose for each frame in a given sequence as a token (Fig. 1(a)). While this baseline approach is functional to an extent, it ignores the natural distinction of spatial relations (joint-to-joint), leaving potential improvements on the table. A natural extension to this baseline is

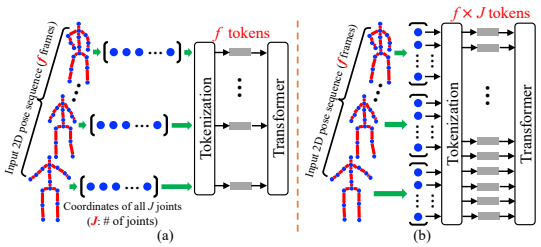

Figure 1. Two baseline approaches.

to instead view each 2D joint coordinate as a token, and provide an input formed with these joints from across all frames of the sequence (Fig. 1(b)). However, in this case, the number of tokens becomes increasingly large when long frame sequences are used (up to 243 frames and 17 joints per frame is common in 3D HPE, the number of tokens would be $243 \times 17 = 4131$). Since the transformer computes direct attention with each token to another, the memory requirement of the model approaches an unreasonable level.

Therefore, as an effective solution to these challenges, we propose **PoseFormer**, the first pure transformer network for 2D-to-3D lifting HPE in videos. PoseFormer directly models the spatial and temporal aspects with distinct transformer modules for both dimensions. Not only does PoseFormer produce strong representations across the spatial and temporal elements, it does so without inducing enormous token counts for long input sequences. On a high level, PoseFormer simply takes a sequence of detected 2D poses from an off-the-shelf 2D pose estimator, and outputs the 3D pose for the center frame. More specifically, we build a spatial transformer module to encode local relationships between the 2D joints in each frame. The spatial self-attention layers consider the position information of 2D joints and return a latent feature representation for that frame. Next, our temporal transformer module analyzes global dependencies between each spatial feature representation, and generates an accurate 3D pose estimation.

Experimental evaluations on two popular 3D HPE benchmarks, Human3.6M [16] and MPI-INF-3DHP [27], show that PoseFormer achieves state-of-the-art performance on both datasets. We visualize our estimated 3D pose compared with the state-of-the-art approach, and find that PoseFormer produces smoother and more reliable results. Also, visualizations and analyses of PoseFormer's attention maps are provided in the ablation study to understand the internal workings of our model and demonstrate its effectiveness. Our **contributions** are three-fold:

- We propose the first pure transformer-based model, PoseFormer, for 3D HPE under the category of 2D-to-3D lifting.
- We design an effective Spatial-Temporal Transformer model, where the spatial transformer module encodes local relationships between human body joints, and the temporal transformer module captures the global dependencies across frames in the entire sequence.
- Without bells and whistles, our PoseFormer model achieves state-of-the-art results on both Human3.6M and MPI-INF-3DHP datasets.

2. Related Works

Here we specifically summarize 3D single-person-single-view HPE methods. Direct estimation approaches infer 3D human pose from 2D images without intermediately estimating 2D pose representation. 2D-to-3D lifting approaches utilize the 2D pose as input to generate the corresponding 3D pose, which is more popular among state-of-the-art methods in this domain. Any off-the-shelf 2D pose estimator can be effectively compatible with these methods. Our proposed method, PoseFormer, also follows the 2D-to-3D lifting pipeline, and therefore we will focus mainly on such methods in this section.

2D-to-3D Lifting HPE. 2D-to-3D lifting approaches leverage 2D poses estimated from input images or video frames. OpenPose [3], CPN [6], AlphaPose [13], and HR-Net [35] have been extensively used as the 2D pose detectors. Based on this intermediate representation, the 3D pose can be generated with a variety of methods. Martinez *et al*. [26] proposed a simple and effective fully connected residual network to regress 3D joint locations based on the 2D joint locations from just a single frame. However, instead of estimating 3D human pose from monocular images, videos can provide temporal information to improve accuracy and robustness [49, 10, 32, 8, 2, 44, 38]. Hossain and Little [33] proposed a recurrent neural network using Long Short-Term Memory (LSTM) cells to exploit temporal information in the input sequence. Several works [10, 2, 21] utilized spatial-temporal relationships and constraints such as bone-length and left-right symmetry to improve performance. Pavllo *et al*. [32] introduced a temporal convolution network to estimate 3D pose over 2D keypoints from consecutive 2D sequences. Based on [32], Chen *et al*. [5] added a bone direction module and bone length module to ensure temporal consistency across video frames, and Liu *et al*. [25] utilized an attention mechanism to recognize significant frames. However, the previous state-of-the-art methods (*e.g*. [25, 5]) rely on dilated temporal convolutions to capture global dependencies, which are inherently limited in temporal connectivity. Additionally, the majority of these works [25, 5, 33, 32] project the joint coordinates to a latent space using simple operations, without considering the kinematic correlations of human joints.

GNNs in 3D HPE. Naturally, a human pose can be represented as a graph where the joints are the nodes and the bones are the edges. Graph Neural Networks (GNNs) have also been applied to the 2D-to-3D pose lifting problem and provided promising performance [9, 45, 24]. Ci *et al*. [9] proposed a framework, named Locally Connected Net-

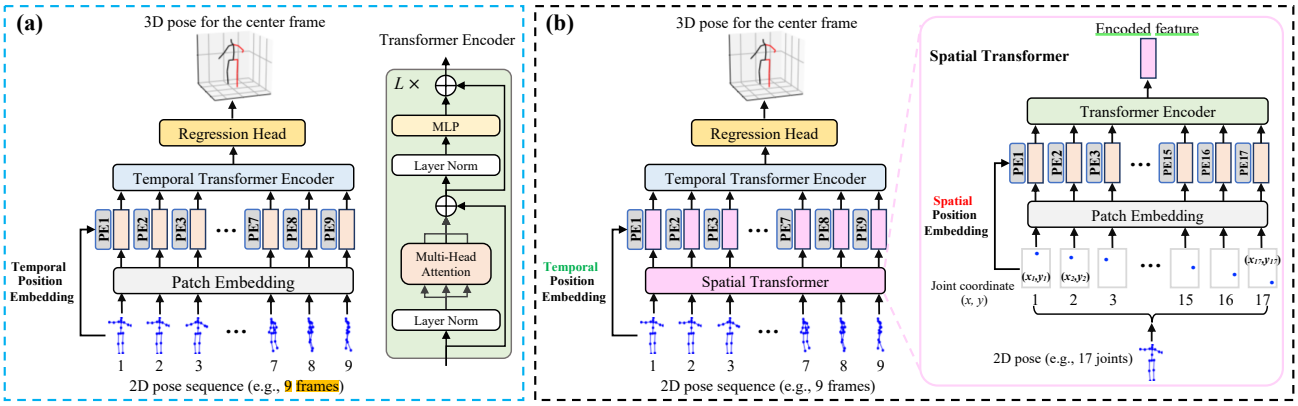

Figure 2. (a) Temporal transformer baseline. (b) Spatial-temporal transformer (PoseFormer) architecture, which consists of three modules. A spatial transformer module for extracting features with considering joints correlations of each individual skeleton. A temporal transformer module for learning global dependencies of entire sequence. A regression head module regresses the final 3D pose of the center frame. The illustration of the transformer encoder is followed by ViT [12].

works (LCNs), which leverages both fully connected networks and GNN operations to encode the relationship between local joint neighborhoods. Zhao *et al*. [45] tackled a limitation of Graph Convolutional Network [19] (GCN) operations, specifically how the weight matrix is shared across nodes. The semantic graph convolution operation was introduced to learn channel-wise weights for edges.

For our PoseFormer, the transformer can be viewed as a type of graph neural network with a unique, and often advantageous, graph operation. Specifically, a transformer encoder module essentially forms a fully-connected graph, where the edge weights are computed using input-conditioned, multi-headed self-attention. The operation also includes the normalization of node features, a feed-forward aggregator across attention head outputs, and residual connections which enable it to scale effectively with stacked layers. Such an operation can be advantageous in comparison to other graph operations. For example, the strength of the connection between nodes is determined by the self-attention mechanism of the transformer, rather than predefined through an adjacency matrix as with the typical GCN-based formulations employed in this task. This allows the model flexibility to adapt the relative importance of joints to each other with each input pose. Additionally, the comprehensive scaling and normalization components of the transformer are likely advantageous in mitigating the over-smoothing effect that troubles many GNN operation variants when numerous layers are stacked together [48].

Vision Transformers. Recently, there is an emerging interest in applying transformers to vision tasks [17, 14]. Carion *et al*. [4] presented a DEtection TRansformer (DETR) for object detection and panoptic segmentation. Dosovitskiy *et al*. [12] proposed a pure transformer architecture, Vision Transformer (ViT), which achieves state-of-the-art performance on image classification. However, ViT was trained on large-scale datasets ImageNet-21k and JFT-300M that requires huge computation resources. Then, a data-efficient image transformer (DeiT) [36] was proposed which builds upon the ViT with knowledge distillation. For regression problems such as HPE, Yang *et al*. [40] proposed a transformer network, Transpose, which only estimates 2D pose from images. Lin *et al*. [23] combined CNNs with transformer networks in their method METRO (MEsh TRansfOrmer) to reconstruct the 3D pose and mesh vertices from a single image. In contrast to our approach, METRO falls under the category of direct estimation. Also, temporal consistency is neglected in METRO, which limits the robustness of its estimations. Our spatial-temporal transformer architecture exploits keypoint correlation in each frame and preserves natural temporal coherence in videos.

3. Method

We follow the same 2D-to-3D lifting pipeline for 3D HPE in videos as [26, 32, 25, 5]. The 2D pose of each frame is obtained by an off-the-shelf 2D pose detector, then 2D pose sequences of consecutive frames are used as input for estimating the 3D pose of the center frame. Compared to the previous state-of-the-art models, which are based on CNNs, we produce a highly competitive **convolution-free** transformer network.

3.1. Temporal Transformer Baseline

As a baseline application of a transformer in 2D-to-3D lifting, we treat each 2D pose as an input token and employ a transformer to capture global dependencies among the inputs as illustrated in Fig. 2(a). We will refer to each input token as a patch, similar in terminology to ViT [12]. For the input sequence $X \in \mathbb{R}^{f\times(J\cdot 2)}$, f is the number of frames of the input sequence, J is the number of joints of each 2D pose, and 2 indicates joint's coordinate in 2D space. $\{\mathbf{x}^i \in \mathbb{R}^{1\times(J\cdot 2)} | i = 1, 2, \dots f\}$ indicates the input vector of each frame. The patch embedding is a trainable linear

projection layer to embed each patch to a high dimensional feature. The transformer network utilizes positional embeddings to retain positional information of the sequence. The procedure can be formulated as:

$$Z_0 = [\mathbf{x}^1 E;\ \mathbf{x}^2 E;\ \ldots;\ \mathbf{x}^f E] + E_{pos}. \quad (1)$$

After embedding through a linear projection matrix $E \in \mathbb{R}^{(J\cdot 2)\times C}$ and summing with the positional embedding $E_{pos} \in \mathbb{R}^{f\times C}$, the input sequence $X \in \mathbb{R}^{f\times(J\cdot 2)}$ becomes $Z_0 \in \mathbb{R}^{f\times C}$, where C is the embedding dimension. Z_0 is sent to the Temporal Transformer Encoder.

As the core function of the transformer, self-attention is designed to relate different positions of the input sequence with embedded features. Our transformer encoder is composed of Multi-head Self Attention blocks with multilayer perceptron (MLP) blocks as in [12]. LayerNorm is applied before every block and residual connections are applied after every block [39, 1].

Scaled Dot-Product Attention can be described as a mapping function that maps a query matrix Q, key matrix K and value matrix V to an output attention matrix. $Q, K, V \in \mathbb{R}^{N\times d}$, where N is the number of vectors in the sequence and d is the dimension. A scaling factor of $\frac{1}{\sqrt{d}}$ is utilized within this attention operation for appropriate normalization, preventing extremely small gradients when large values of d lead dot products to grow large in magnitude. Thus the output of the scaled dot-product attention can be expressed as:

$$\text{Attention}(Q, K, V) = \text{Softmax}(QK^\top/\sqrt{d})V. \quad (2)$$

In our temporal transformer, $d = C$ and $N = f$. The Q, K and V are computed from the embedded feature $Z \in \mathbb{R}^{f\times C}$ by linear transformations W_Q, W_K and $W_V \in \mathbb{R}^{C\times C}$:

$$Q = ZW_Q, \quad K = ZW_K, \quad V = ZW_V. \quad (3)$$

Multi-head Self Attention Layer (MSA) utilizes multiple heads to model the information jointly from various representation subspaces with different positions. Each head applies scaled dot-product attention in parallel. The final MSA output will be the concatenation of h attention head outputs.

$$\text{MSA}(Q, K, V) = \text{Concat}(H_1, H_2, \ldots, H_h)W_{out} \quad (4)$$

$$\text{where} \quad H_i = \text{Attention}(Q_i, K_i, V_i), i \in [1, ..., h] \quad (5)$$

The Temporal Transformer Encoder structure of L layers given our embedded feature $Z_0 \in \mathbb{R}^{f\times C}$ can be represented as follows:

$$Z'_\ell = \text{MSA}(\text{LN}(Z_{\ell-1})) + Z_{\ell-1}, \quad \ell = 1, 2 \ldots L \quad (6)$$

$$Z_\ell = \text{MLP}(\text{LN}(Z_\ell)) + Z'_\ell, \quad \ell = 1, 2, \ldots L \quad (7)$$

$$Y = \text{LN}(Z_L), \quad (8)$$

where $\text{LN}(\cdot)$ denotes the layer normalization operator (same as in ViT). The temporal transformer encoder consists of L identical layers and the encoder output $Y \in \mathbb{R}^{f\times C}$ keeps the same size as encoder input $Z_0 \in \mathbb{R}^{f\times C}$.

In order to predict the 3D pose of center frame, the encoder output $Y \in \mathbb{R}^{f\times C}$ is shrunk to a vector $\mathbf{y} \in \mathbb{R}^{1\times C}$ by taking the average in the frame dimension. Finally, an MLP block will regress the output to $\mathbf{y} \in \mathbb{R}^{1\times(J\cdot 3)}$, which is the 3D pose of the center frame.

3.2. PoseFormer: Spatial-Temporal Transformer

We observe that the temporal transformer baseline mainly focuses on global dependencies between frames in the input sequence. The patch embedding, a linear transformation, is utilized to project joint coordinates to a hidden dimension. However, the kinematic information between local joint coordinates is not strongly represented in the temporal transformer baseline because *a simple linear projection layer* is not able to learn attention information. One potential workaround is to view each joint coordinate as an individual patch and feeding the joints from all frames as input to the transformer (see Fig. 1(b)). However, the number of patches would increase rapidly (frames f multiplied by the number of joint J), resulting in a model computational complexity of $O((f \cdot J)^2)$. For example, if we use 81 frames and 17 joints for each 2D pose, the number of patches would be 1377 (ViT model uses 576 patches (input size = 384×384, patch size = 16×16)).

In order to learn local joint correlations efficiently, we employ two separated transformers for spatial and temporal information, respectively. As shown in Fig. 2(b), the proposed PoseFormer consists of three modules: spatial transformer module, temporal transformer module, and regression head module.

Spatial Transformer Module. The spatial transformer module is designed to extract a high dimensional feature embedding from a single frame. Given a 2D pose with J joints, we consider each joint (*i.e.* two coordinates) as a patch and follow the general vision transformer pipeline to perform the feature extraction among all patches. First, we map the coordinate of each joint to a high dimension with a trainable linear projection, which is referred to as the spatial patch embedding. We sum this with the learnable spatial positional embedding $E_{SPos} \in \mathbb{R}^{J\times c}$, and therefore the input $\mathbf{x}_i \in \mathbb{R}^{1\times(J\cdot 2)}$ of the i-th frame becomes $z_0^i \in \mathbb{R}^{J\times c}$, where 2 indicates 2D coordinate in each frame and c is the spatial embedding dimension. The resulting joint sequence of features z_0^i are then fed into the spatial transformer encoder which applies the self-attention mechanism to integrate information across all joints. For the i-th frame, the output of spatial transformer encoder with L layers will be $z_L^i \in \mathbb{R}^{J\times c}$.

Temporal Transformer Module. Since the spatial transformer module encodes high dimensional features for

each individual frame, the goal for the temporal transformer module is to model dependencies across the sequence of frames. For the i-th frame, the output of the spatial transformer $z_L^i \in \mathbb{R}^{J\times c}$ is flattened as a vector $\mathbf{z}^i \in \mathbb{R}^{1\times(J\cdot c)}$. We then concatenate these vectors $\{\mathbf{z}^1, \mathbf{z}^2, \ldots, \mathbf{z}^f\}$ from the f input frames as $Z_0 \in \mathbb{R}^{f\times(J\cdot c)}$. Before the temporal transformer module, we add the learnable temporal positional embedding $E_{TPos} \in \mathbb{R}^{f\times(J\cdot c)}$ to retain frame position information. For the temporal transformer encoder, we apply the same architecture as the spatial transformer encoder, which consists of multihead self-attention blocks and MLP blocks. The output of the temporal transformer module is $Y \in \mathbb{R}^{f\times(J\cdot c)}$.

Regression Head. Since we use a sequence of frames to predict the 3D pose of the center frame, the output of the temporal transformer module $Y \in \mathbb{R}^{f\times(J\cdot c)}$ needs to be reduced to $\mathbf{y} \in \mathbb{R}^{1\times(J\cdot c)}$. We apply a weighted mean operation (with learned weights) on the frame dimension to achieve this. Finally, a simple MLP block with Layer norm and one linear layer returns output $\mathbf{y} \in \mathbb{R}^{1\times(J\cdot 3)}$ which is the predicted 3D pose of the center frame.

Loss Function. To train our spatial-temporal transformer model, we apply the standard MPJPE (Mean Per Joint Position Error) loss to minimize the error between the predicted and ground truth pose as

$$\mathcal{L} = \frac{1}{J}\Sigma_{k=1}^{J}\|p_k - \hat{p}_k\|_2, \tag{9}$$

where p_k and $\hat{p}_k$ are the ground truth and estimated 3D joint locations of the k-th joint, respectively.

4. Experiments

4.1. Datasets and Evaluation Metrics

We evaluate our model on two commonly used 3D HPE datasets, Human3.6M [16] and MPI-INF-3DHP [27].

Human3.6M [16] is the most widely used indoor dataset for 3D single person HPE. There are 11 professional actors performing 17 actions such as sitting, walking, and talking on the phone. Videos of each subject were recorded from 4 different views in an indoor environment. This dataset contains 3.6 million video frames with 3D ground truth annotation captured by an accurate marker-based motion capture system. Following previous works [32, 25, 5], we adopt the same experiment setting: all 15 actions are used for training and testing, the model is trained on five sections (S1, S5, S6, S7, S8) and tested on two subjects (S9 and S11).

MPI-INF-3DHP [27] is a more challenging 3D pose dataset. It contains both constrained indoor scenes and complex outdoor scenes. There are 8 actors performing 8 actions from 14 camera views which cover a greater diversity of poses. MPI-INF-3DHP provides a test set of 6 subjects with different scenes. We follow the setting in [22, 5, 38].

For the Human3.6M dataset, we use the most common evaluation metrics (MPJPE and P-MPJPE) [46] to evaluate the performance of our estimation with the ground truth 3D pose. MPJPE (Mean Per Joint Position Error) is computed as the mean Euclidean distance between the estimated joints and the ground truth in millimeters; we refer to MPJPE as Protocol 1. P-MPJPE is the MPJPE after rigid alignment by post-processing between the estimated 3D pose and the ground truth and it is more robust to individual joint prediction failure. We refer to P-MPJPE as Protocol 2.

For the MPI-INF-3DHP dataset, we use MPJPE, Percentage of Correct Keypoint (PCK) within the 150mm range [22, 5, 38], and Area Under Curve (AUC).

4.2. Implementation Details

实验细节写的比较详细，可以参照

We implemented our proposed method with Pytorch [30]. Two NVIDIA RTX 3090 GPUs were used for training and testing. We chose three different frame sequence lengths when conducting our experiments, *i.e.* $f = 9$, $f = 27$, $f = 81$. The details about number of frames with results are discussed in the ablation studies (Sec. 4.4). We apply pose flipping horizontally as data augmentation both in training and testing following [32, 25, 5]. We train our model using the Adam [18] optimizer for 130 epochs with weight decay of 0.1. We adopt an exponential learning rate decay schedule with the initial learning rate of 2e-4 and decay factor of 0.98 of each epoch. We set the batch size to 1024 and employ stochastic depth [15] with a rate of 0.1 for transformer encoder layers. For the 2D pose detector, we use the cascaded pyramid network (CPN) [7] on Human3.6M following [32, 25, 5], and we use the ground truth 2D pose as input for MPI-INF-3DHP following [28, 22, 38].

4.3. Comparison with State-of-the-Art

Human3.6M. We report all 15 action results of the test set (S9 and S11) in Table 1. The last column provides the average performance on all of the test set. Following the 2D-to-3D lifting approach, we use the CPN network as the 2D pose detector, then use the detected 2D pose as input for training and testing. PoseFormer outperforms our baseline (*i.e.* temporal transformer baseline in Sec. 3.1) by a large margin (6.1% and 6.4%) under protocol 1 and protocol 2, respectively. This clearly demonstrates the advantage of using spatial transformer to expressively model the correlations between joints in each frame. PoseFormer yields the lowest average MPJPE of 44.3mm under protocol 1 as shown in Table 1 (top). Comparing with the transformer-based method METRO [23], which ignores the temporal consistency since the 3D pose is estimated by a single image, PoseFormer reduces the MPJPE by approximately 18%. For Protocol 2, we also obtain the best overall result as shown in Table 1 (bottom). Moreover, PoseFormer achieves more accurate pose predictions on difficult actions such as *Photo*, *SittingDown*, *WalkDog*, and *Smoke*. Unlike

Table 1. Quantitative comparison of Mean Per Joint Position Error between the estimated 3D pose and the ground truth 3D pose on Human3.6M under Protocols 1&2 using the detected 2D pose as input. Top-table: results under Protocol 1 (MPJPE). Bottom-table: results under Protocol 2 (P-MPJPE). f denotes the number of input frames used in each method, $*$ indicates that the input 2D pose is detected by the cascaded pyramid network (CPN), and † denotes a Transformer-based model. (Red: best; Blue: second best)

Protocol 1		Dir.	Disc.	Eat.	Greet	Phone	Photo	Pose	Purch.	Sit	SitD.	Somke	Wait	WalkD.	Walk	WalkT.	Average
Dabral *et al.* [11]	ECCV'18	44.8	50.4	44.7	49.0	52.9	61.4	43.5	45.5	63.1	87.3	51.7	48.5	52.2	37.6	41.9	52.1
Cai *et al.* [2] ($f = 7$)	ICCV'19	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.* [32] ($f = 243$)*	CVPR'19	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
Lin *et al.* [22] ($f = 50$)	BMVC'19	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
Yeh *et al.* [42]	NIPS'19	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
Liu *et al.* [25] ($f = 243$)*	CVPR'20	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
SRNet [43] *	ECCV'20	46.6	47.1	43.9	41.6	45.8	49.6	46.5	40.0	53.4	61.1	46.1	42.6	43.1	31.5	32.6	44.8
UGCN [38] ($f = 96$)	ECCV'20	41.3	43.9	44.0	42.2	48.0	57.1	42.2	43.2	57.3	61.3	47.0	43.5	47.0	32.6	31.8	45.6
Chen *et al.* [5] ($f = 81$)*	TCSVT'21	42.1	43.8	41.0	43.8	46.1	53.5	42.4	43.1	53.9	60.5	45.7	42.1	46.2	32.2	33.8	44.6
METRO [23] ($f = 1$) †	CVPR'21	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	54.0
Baseline ($f = 81$)*†		43.8	47.9	43.8	45.5	49.7	55.7	44.3	45.8	57.7	66.3	47.4	45.4	48.6	32.5	33.8	47.2
PoseFormer ($f = 81$)*†		41.5	44.8	39.8	42.5	46.5	51.6	42.1	42.0	53.3	60.7	45.5	43.3	46.1	31.8	32.2	44.3
Protocol 2		Dir.	Disc.	Eat.	Greet	Phone	Photo	Pose	Purch.	Sit	SitD.	Somke	Wait	WalkD.	Walk	WalkT.	Average
Pavlakos *et al.* [31]	CVPR'18	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
Hossain *et al.* [33]	ECCV'18	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
Cai *et al.* [2] ($f = 7$)	ICCV'19	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
Lin *et al.* [22] ($f = 50$)	BMVC'19	32.5	35.3	34.3	36.2	37.8	43.0	33.0	32.2	45.7	51.8	38.4	32.8	37.5	25.8	28.9	36.8
Pavllo *et al.* [32] ($f = 243$)*	CVPR'19	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
Liu *et al.* [25] ($f = 243$)*	CVPR'20	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
UGCN [38] ($f = 96$)	ECCV'20	32.9	35.2	35.6	34.4	36.4	42.7	31.2	32.5	45.6	50.2	37.3	32.8	36.3	26.0	23.9	35.5
Chen *et al.* [5] ($f = 243$)*	TCSVT'21	33.1	35.3	33.4	35.9	36.1	41.7	32.8	33.3	42.6	49.4	37.0	32.7	36.5	25.5	27.9	35.6
Baseline ($f = 81$)*†		33.6	37.1	35.4	36.7	37.8	42.2	33.9	34.7	47.0	53.4	38.2	34.3	37.6	25.3	27.8	37.0
PoseFormer ($f = 81$)*†		32.5	34.8	32.6	34.6	35.3	39.5	32.1	32.0	42.8	48.5	34.8	32.4	35.3	24.5	26.0	34.6

other simple actions, poses in these actions change more quickly and some long-distance frames have strong correlations. In this case, global dependencies play an important role, and the attention mechanisms of the transformer are particularly advantageous.

To further investigate the lower bound of our method, we directly use the ground truth 2D pose as input to alleviate error caused by noisy 2D pose data. The results are shown in Table 2. The MPJPE is reduced from 44.5mm to 31.3mm, about 29.7% by using the clean 2D pose data. PoseFormer achieves the best score in 9 actions and the second best score in 6 actions. The average score is improved by approximately 2% compared with SRNet [43].

In Fig. 3, we also compare the MPJPE for some of the individual joints which have the largest errors on Human3.6M test set S11 with action *Photo*. PoseFormer achieves better performance on these difficult joints than [32, 5].

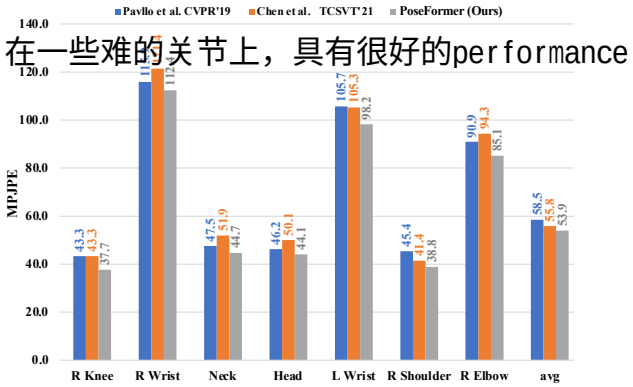

Figure 3. The average joint error comparison across all the frames of the Human3.6M test set S11 with the *Photo* action.

MPI-INF-3DHP. Table 3 reports the quantitative results of PoseFormer with other methods on MPI-INF-3DHP. This dataset contains fewer training samples compared to Human3.6M, and some of the samples are taken from outdoor scenes. We use 2D poses of 9 frames as our model input due to the typically shorter sequence lengths of this dataset. Our method again achieves the best performances on all three evaluation metrics (PCK, AUC and MPJPE).

Qualitative Results. We also provide a visual comparison between the 3D estimated pose and the ground truth. We evaluate PoseFormer on the Human3.6M test set S11 with the *Photo* action, which is one of the most challenging actions (all methods have a high MPJPE). Compared with state-of-the-art method [5], our PoseFormer achieves more accurate predictions as shown in Fig. 4.

4.4. Ablation Study

To verify the contribution of the individual components of PoseFormer and the impact of hyperparameters on performance, we conduct extensive ablation experiments with the Human3.6M dataset under protocol 1.

The Design of PoseFormer. We investigate the impact of the spatial transformer, as well as the positional embeddings of the spatial and temporal transformers in Table 4. We input 9 frames of CPN-detected 2D poses ($J = 17$) to predict the 3D pose. All the architecture parameters are fixed for fairly comparing the impact of each module; the spatial transformer embedding dimension is $17 \times 32 = 544$ and the number of spatial transformer encoder layers is 4. For the temporal transformer, the embedding dimension is consistent with the spatial transformer (that is 544) and we also apply 4 temporal transformer layers. To verify the impact of our spatial-temporal design, we first compare with the transformer baseline we started with in Sec. 3.1. The results in Table 4 demonstrate that our spatial-temporal transformer makes a significant impact (from 52.5 to 49.9 MPJPE), as the joint-wise correlations are more strongly modeled. This is also consistent with the results (Baseline

Table 2. Quantitative comparison of Mean Per Joint Position Error between the estimated 3D pose and the ground truth 3D pose on Human3.6M dataset under Protocol 1 (MPJPE) using the **ground truth** 2D pose as input. (Red: best; Blue: second best)

GT Protocol 1		Dir.	Disc.	Eat.	Greet	Phone	Photo	Pose	Purch.	Sit	SitD.	Somke	Wait	WalkD.	Walk	WalkT.	Average
Hossain *et al.* [33]	ECCV'18	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
Pavllo *et al.* [32] ($f = 243$)	CVPR'19	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	37.2
Liu *et al.* [25] ($f = 243$)	CVPR'20	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
SRNet [43]	ECCV'20	34.8	32.1	28.5	30.7	31.4	36.9	35.6	30.5	38.9	40.5	32.5	31.0	29.9	22.5	24.5	32.0
Chen *et al.* [5] ($f = 243$)	TCSVT'21	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	32.3
PoseFormer ($f = 81$)		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

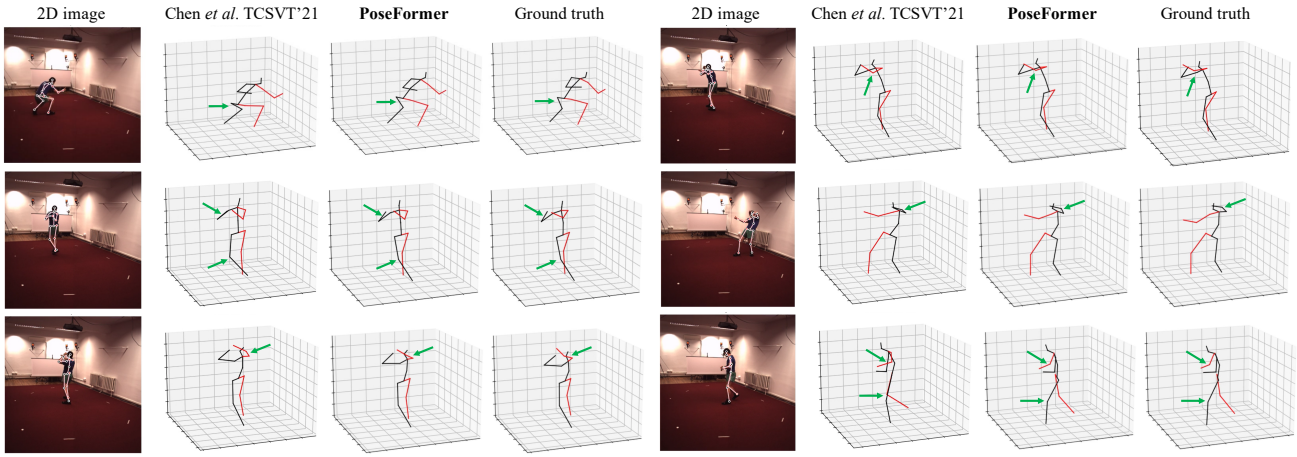

Figure 4. Qualitative comparison between our method (PoseFormer) and the SOTA approach Chen *et al.* [5] on Human3.6M test set S11 with the *Photo* action. The green arrows highlight locations where PoseFormer clearly has better results.

Table 3. Quantitative comparison with previous methods on MPI-INF-3DHP. The best scores are marked in bold.

		PCK ↑	AUC ↑	MPJPE ↓
Mehta *et al.* [27]	3DV'17	75.7	39.3	117.6
Mehta *et al.* [28]	ACM ToG'17	76.6	40.4	124.7
Pavllo *et al.* [32] (81 frames)	CVPR'19	86.0	51.9	84.0
Pavllo *et al.* [32] (243 frames)	CVPR'19	85.5	51.5	84.8
Lin *et al.* [22] (25 frames)	BMVC'19	83.6	51.4	79.8
Li *et al.* [20]	CVPR'20	81.2	46.1	99.7
Chen *et al.* [5] (81 frames)	TCSVT'21	87.9	54.0	78.8
PoseFormer (9 frames)		**88.6**	**56.4**	**77.1**

Table 4. Ablation study on different components in PoseFormer. The evaluation is performed on Human3.6M (Protocol 1) using detected 2D pose as input. (T: Temporal only; S-T: Spatial-temporal)

Input length (f)	T	S-T	Spatial Pos Emb	Temporal Pos Emb	MPJPE
9	✓	✗	✗	✓	52.5
9	✗	✓	✗	✗	51.6
9	✗	✓	✓	✗	50.7
9	✗	✓	✗	✓	50.5
9	✗	✓	✓	✓	**49.9**

vs. PoseFormer) in Table 1 when $f = 81$. Next, we evaluate the impact of the positional embeddings. We explore the four possible combinations: without positional embeddings, only apply the spatial positional embedding, only apply the temporal positional embedding, and apply both spatial and temporal positional embeddings. Comparing the results of these combinations, it is obvious that positional embeddings improve the performance. By applying these on both the spatial and temporal modules, the best overall result is achieved.

Architecture Parameter Analysis. We explore the various parameter combinations to find the optimal network architecture in Table 5. c represents the embedded feature dimension in the spatial transformer and L indicates how many layers are used in the transformer encoder. In PoseFormer, the output of the spatial transformer is flattened and added with the temporal positional embedding to form the input of the temporal transformer encoder. Thus the embedding feature dimension in the temporal transformer encoder is $c \times J$. The optimal parameters for our model are $c = 32$, $L_S = 4$, and $L_T = 4$.

Table 5. Ablation study on different architecture parameters in PoseFormer. The evaluation is performed on Human3.6M (Protocol 1) using detected 2D pose as input. c is the spatial transformer patch embedding dimension. L_S and L_T indicate the number of layers in the spatial and temporal transformers, respectively.

c	16	16	16	32	32	32	48	48	48
L_S	2	4	6	2	4	6	2	4	6
L_T	2	4	6	2	4	6	2	4	6
MPJPE	52.8	51.7	50.4	52.4	**49.9**	50.3	51.7	50.4	50.5

Table 6. Comparison on computational complexity, MPJPE, and inference speed (frame per second (FPS)). The evaluation is performed on Human3.6M under Protocol 1 using detected 2D pose as input. FPS is based on a single GeForce GTX 2080 Ti GPU.

	f	Parameters (M)	FLOPs (M)	MPJPE	FPS
Hossain and Little [33]	-	16.95	33.88	58.3	-
Pavllo *et al.* [32]	27	8.56	17.09	48.8	1492
Pavllo *et al.* [32]	81	12.79	25.48	47.7	1121
Pavllo *et al.* [32]	243	16.95	33.87	46.8	863
Chen *et al.* [5]	27	31.88	61.7	45.3	410
Chen *et al.* [5]	81	45.53	88.9	44.6	315
Chen *et al.* [5]	243	59.18	116	44.1	264
PoseFormer	9	9.58	11.2	49.9	320
PoseFormer	27	9.59	33.9	47.0	297
PoseFormer	81	9.60	101	44.5	269

Computational Complexity Analysis. We report the model performance, total number of parameters, and estimated floating-point operations (FLOPs) per frame, and the number of output frames-per-second (FPS) with various input sequence lengths (f) in Table 6. Our model achieves better accuracy when the sequence length is increased, and the total number of parameters does not increase much. This is because the number of frames only affects the temporal positional embedding layer, which does not require many parameters. Compared with other models, our model requires fewer total parameters with competitive performance. We report the inference FPS of different models on a single GeForce RTX 2080 Ti GPU, following the same settings in [5]. Although our model's inference speed is not the absolute fastest, the speed is still acceptable for real-time inference. For complete 3D HPE processing, the 2D pose is first detected by the 2D pose detector, then the 3D pose is estimated by our method. The FPS for the common 2D pose detector is usually below 80, which means the inference speed of our model will not be the bottleneck.

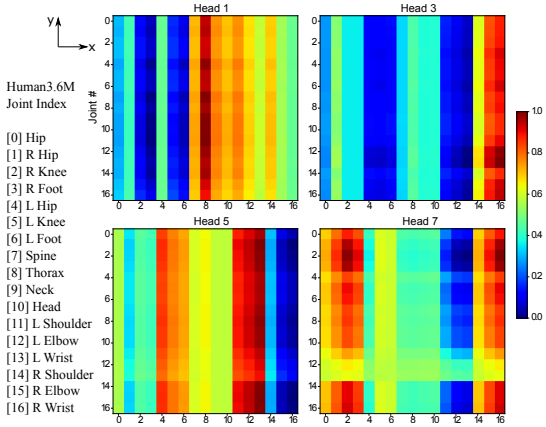

Figure 5. Visualization of self-attentions in the spatial transformer. The x-axis (horizontal) and y-axis (vertical) correspond to the queries and the predicted outputs, respectively. The pixel $w_{i,j}$ (i: row, j: column) denotes the attention weight of the j-th query for the i-th output. Red indicates stronger attention. The attention output is normalized from 0 to 1.

Attention Visualization. In order to illustrate the attention mechanism through multi-head self attention blocks, we evaluate our model on Human3.6M test set S11 for a particular action (*SittingDown*) and visualize the self-attention maps from the spatial and temporal transformers separately as shown in Fig. 5 and Fig. 6. For the spatial self-attention maps, the x-axis corresponds to the query of 17 joints and the y-axis indicates the attention output. As shown in Fig. 5, the attention heads return different attention intensities which represent the various local relations learned among the input joints. We discover that Head 3 focuses on joints 15 and 16, which are the right elbow and right wrist. Head 5 builds the connection of the left leg to the left arm (joints

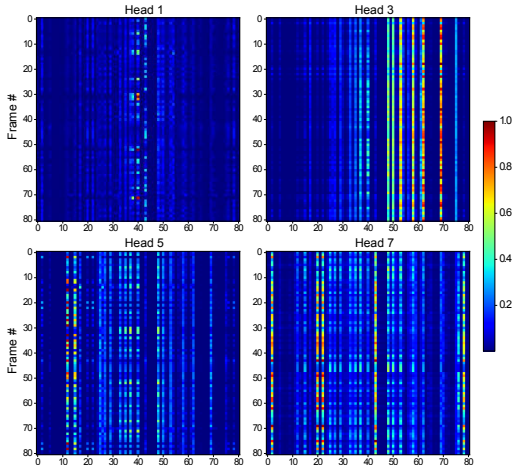

Figure 6. Visualization of self-attentions in temporal transformer. The x-axis (horizontal) and y-axis (vertical) correspond to the queries and the predicted outputs, respectively. The pixel $w_{i,j}$ (i: row, j: column) denotes the attention weight of the j-th query for the i-th output. Red indicates stronger attention. The attention output is normalized from 0 to 1.

4, 5, 6 and joints 11, 12, 13). These joints can be grouped as the left portion of the body, while Head 7 concentrates on the right side (joint 1, 2, 3 with joint 12, 13, 14).

For the temporal self-attention maps in Fig. 6, the x-axis corresponds to the query of 81 frames and the y-axis indicates the attention output. Long term global dependencies are learned by different attention heads. The attention of Head 3 highly correlates to some frames (*e.g.* frame 58, 62, and 69) on the right side of the center frame. Head 7 captures the dependencies of frame 1, 20, 22, 42, 78 despite their long distances. The spatial and temporal attention maps demonstrate that PoseFormer successfully models local relationships between joints, as well as captures long term global dependencies of the entire input sequence.

Table 7. MPJPE evaluation on HumanEva test set. FT indicates using pre-trained model on Human3.6M for fine tuning.

	walk			jog		
	S1	S2	S3	S1	S2	S3
PoseFormer ($f = 43$)	16.3	11.0	47.1	25.0	15.2	15.1
PoseFormer ($f = 43$) FT	**14.4**	**10.2**	**46.6**	**22.7**	**13.4**	**13.4**

Generalization to Small Datasets. Prior work such as [12] concluded that transformers do not generalize well when trained on insufficient amounts of data. We conduct an experiment with our model to investigate the transformer learning ability on a small dataset – HumanEva [34]. It is a much smaller dataset (<50K frames) compared with Human3.6M (>1M frames). Table 7 shows the results of training from scratch as well as fine tuning using the pre-trained model on Human3.6M. We find that the performance can be improved by a large margin when fine tuning, which follows previous observations [12, 36] that transformers can perform well when pre-trained on a large-scale dataset.

5. Conclusion

In this paper, we present PoseFormer, a pure transformer-based approach for 3D pose estimation from 2D videos. The spatial transformer module encodes the local relationships between the 2D joints and the temporal transformer module captures global dependencies across the arbitrary frames regardless of the distance. Extensive experiments show that our model achieves state-of-the-art performance on two popular 3D pose datasets.

References

[1] Alexei Baevski and Michael Auli. Adaptive input representations for neural language modeling. *arXiv preprint arXiv:1809.10853*, 2018.

[2] Y. Cai, L. Ge, J. Liu, J. Cai, T. Cham, J. Yuan, and N. M. Thalmann. Exploiting spatial-temporal relationships for 3d pose estimation via graph convolutional networks. In *ICCV*, 2019.

[3] Zhe Cao, Tomas Simon, Shih-En Wei, and Yaser Sheikh. Realtime multi-person 2d pose estimation using part affinity fields. In *CVPR*, 2017.

[4] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European Conference on Computer Vision*, pages 213–229. Springer, 2020.

[5] 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*, 2021.

[6] Yilun Chen, Zhicheng Wang, Yuxiang Peng, Zhiqiang Zhang, Gang Yu, and Jian Sun. Cascaded pyramid network for multi-person pose estimation. In *CVPR*, 2018.

[7] 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.

[8] Y. Cheng, B. Yang, B. Wang, Y. Wending, and R. Tan. Occlusion-aware networks for 3d human pose estimation in video. In *ICCV*, 2019.

[9] H. Ci, C. Wang, X. Ma, and Y. Wang. Optimizing network structure for 3d human pose estimation. In *ICCV*, 2019.

[10] Rishabh Dabral, Anurag Mundhada, Uday Kusupati, Safeer Afaque, Abhishek Sharma, and Arjun Jain. Learning 3d human pose from structure and motion. In *ECCV*, 2018.

[11] 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.

[12] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *ICLR*, 2021.

[13] Hao-Shu Fang, Shuqin Xie, Yu-Wing Tai, and Cewu Lu. Rmpe: Regional multi-person pose estimation. In *ICCV*, 2017.

[14] Kai Han, Yunhe Wang, Hanting Chen, Xinghao Chen, Jianyuan Guo, Zhenhua Liu, Yehui Tang, An Xiao, Chunjing Xu, Yixing Xu, et al. A survey on visual transformer. *arXiv preprint arXiv:2012.12556*, 2020.

[15] Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *European conference on computer vision*, pages 646–661. Springer, 2016.

[16] C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu. Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE TPAMI*, 2014.

[17] Salman Khan, Muzammal Naseer, Munawar Hayat, Syed Waqas Zamir, Fahad Shahbaz Khan, and Mubarak Shah. Transformers in vision: A survey. *arXiv preprint arXiv:2101.01169*, 2021.

[18] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[19] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017.

[20] 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 (CVPR)*, June 2020.

[21] Z. Li, X. Wang, F. Wang, and P. Jiang. On boosting single-frame 3d human pose estimation via monocular videos. In *ICCV*, 2019.

[22] Jiahao Lin and Gim Hee Lee. Trajectory space factorization for deep video-based 3d human pose estimation. In *BMVC*, 2019.

[23] Kevin Lin, Lijuan Wang, and Zicheng Liu. End-to-end human pose and mesh reconstruction with transformers. *arXiv preprint arXiv:2012.09760*, 2020.

[24] 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 *ECCV*, 2020.

[25] 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 *CVPR*, 2020.

[26] Julieta Martinez, Rayat Hossain, Javier Romero, and James J. Little. A simple yet effective baseline for 3d human pose estimation. In *ICCV*, 2017.

[27] 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.

[28] Dushyant Mehta, Srinath Sridhar, Oleksandr Sotnychenko, Helge Rhodin, Mohammad Shafiei, Hans-Peter Seidel, Weipeng Xu, Dan Casas, and Christian Theobalt. Vnect:

Real-time 3d human pose estimation with a single rgb camera. *ACM Transactions on Graphics (TOG)*, 36(4):1–14, 2017.

[29] 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*, 2020.

[30] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[31] Georgios Pavlakos, Xiaowei Zhou, and Kostas Daniilidis. Ordinal depth supervision for 3D human pose estimation. In *CVPR*, 2018.

[32] Dario Pavllo, Christoph Feichtenhofer, David Grangier, and Michael Auli. 3d human pose estimation in video with temporal convolutions and semi-supervised training. In *CVPR*, 2019.

[33] Mir Rayat Imtiaz Hossain and James J. Little. Exploiting temporal information for 3d human pose estimation. In *The European Conference on Computer Vision (ECCV)*, 2018.

[34] L. Sigal, A. Balan, and M. J. Black. HumanEva: Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion. *IJCV*, 2010.

[35] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. Deep high-resolution representation learning for human pose estimation. In *CVPR*, 2019.

[36] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. *arXiv preprint arXiv:2012.12877*, 2020.

[37] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. 2017.

[38] Jingbo Wang, Sijie Yan, Yuanjun Xiong, and Dahua Lin. Motion guided 3d pose estimation from videos. In *European Conference on Computer Vision*, pages 764–780. Springer, 2020.

[39] Qiang Wang, Bei Li, Tong Xiao, Jingbo Zhu, Changliang Li, Derek F Wong, and Lidia S Chao. Learning deep transformer models for machine translation. *arXiv preprint arXiv:1906.01787*, 2019.

[40] Sen Yang, Zhibin Quan, Mu Nie, and Wankou Yang. Transpose: Towards explainable human pose estimation by transformer. *arXiv preprint arXiv:2012.14214*, 2020.

[41] Linwei Ye, Mrigank Rochan, Zhi Liu, and Yang Wang. Cross-modal self-attention network for referring image segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10502–10511, 2019.

[42] Raymond A Yeh, Yuan-Ting Hu, and Alexander G Schwing. Chirality nets for human pose regression. 2019.

[43] 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 *ECCV*, 2020.

[44] Tianshu Zhang, Buzhen Huang, and Yangang Wang. Object-occluded human shape and pose estimation from a single color image. In *CVPR*, 2020.

[45] Long Zhao, Xi Peng, Yu Tian, Mubbasir Kapadia, and Dimitris N. Metaxas. Semantic graph convolutional networks for 3d human pose regression. In *CVPR*, 2019.

[46] Ce Zheng, Wenhan Wu, Taojiannan Yang, Sijie Zhu, Chen Chen, Ruixu Liu, Ju Shen, Nasser Kehtarnavaz, and Mubarak Shah. Deep learning-based human pose estimation: A survey, 2020.

[47] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip HS Torr, et al. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. *arXiv preprint arXiv:2012.15840*, 2020.

[48] Kaixiong Zhou, Xiao Huang, Yuening Li, Daochen Zha, Rui Chen, and Xia Hu. Towards deeper graph neural networks with differentiable group normalization. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 4917–4928. Curran Associates, Inc., 2020.

[49] Xingyi Zhou, Qixing Huang, Xiao Sun, Xiangyang Xue, and Yichen Wei. Towards 3d human pose estimation in the wild: A weakly-supervised approach. In *ICCV*, 2017.

[50] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. *arXiv preprint arXiv:2010.04159*, 2020.

Appendix

In this Appendix, we provide the following items:

- Comprehensive visualizations of spatial and temporal attention maps.
- Frame-wise comparison to track the average MPJPE of all the joints across frames.
- More qualitative comparison of estimated 3D poses.
- Estimated 3D poses using the proposed PoseFormer on the in-the-wild videos collected from YouTube.

We also provide demo videos to showcase the 3D human pose estimation results of our proposed PoseFormer. For more details, please visit https://github.com/zczcwh/PoseFormer

A. Attention Visualization

We present more visualization examples of spatial attention maps and temporal attention maps for all 8 heads when evaluating our PoseFormer model on Human3.6M test set S11 with the *SittingDown* action. For the spatial self-attention maps in Fig. 7, the x-axis corresponds to the query of 17 joints and the y-axis indicates the attention output. The attention heads return different attention intensities which represent the various local relations learned among the input joints. For the temporal self-attention maps in Fig. 8, the x-axis corresponds to the query of 81 frames and the y-axis indicates the attention output. Long term global dependencies are captured by different attention heads. The spatial and temporal attention maps have demonstrated that PoseFormer successfully encodes the local relationship between 2D joints as well as models global dependencies cross the arbitrary frames regardless of the distance.

B. Frame-wise Analysis

We perform frame-wise estimation analysis by computing the average MPJPE of all estimated joints in each frame. As shown in Fig. 9, we measure the frame-wise MPJPE through Human3.6M [16] test set S11 with *Eating* and *Photo* actions. Our PoseFormer (red line) yields lower MPJPE in most frames of both actions, compared with our baseline (temporal transformer only) and the state-of-the-art method [5].

C. More Qualitative Results

We provide more visual comparison between the 3D estimated pose and the ground truth. We evaluate PoseFormer on the Human3.6M test set S11 with the *Greeting* and *WalkDog* actions. Compared with the state-of-the-art method [5] and our baseline, PoseFormer achieves more accurate estimations as shown in Fig. 10.

D. Performance on Videos in-the-wild

Our model was trained on the indoor dataset: Human3.6M that the background is static and the camera capture setting is known. Estimating 3D human pose from in-the-wild videos is more challenging due to the dynamic environment and unknown camera setting. There are often high variations in foreground/background objects appearances and severe occlusions in unconstrained environment. We also evaluate the performance of our PoseFormer on some online videos from YouTube as shown in Fig. 11. We first use AlphaPose [13] as 2D pose detector to generate 2D poses from the video frames, then apply PoseFormer for 3D pose estimation. We observe that PoseFormer achieves acceptable performance in most of the frames, but there are still some failure cases (see Fig. 11) due to inaccurate 2D pose detection, occlusion, and fast motion. Since PoseFormer is a 2D-to-3D lifting approach, any incorrect detected 2D poses may lead to inaccurate 3D pose estimation. Occlusion is a key challenge remains in 3D HPE since the information is missing. Moreover, estimation from the extreme fast motion may be affected by the motion blurring of frames.

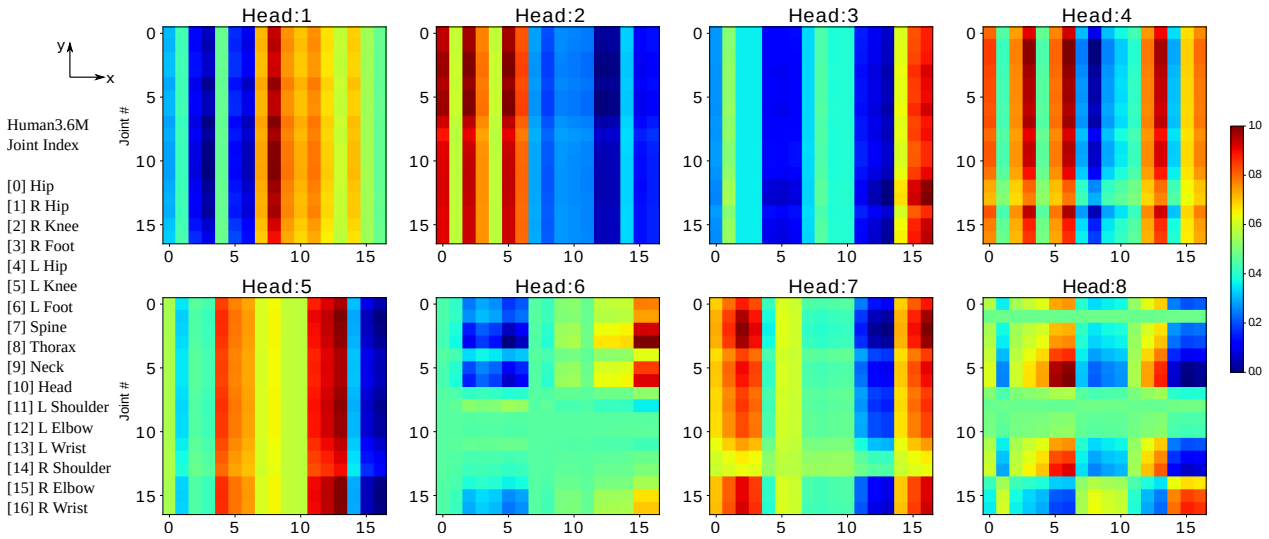

Figure 7. Visualization of self-attentions in the spatial transformer. The x-axis (horizontal) and y-axis (vertical) correspond to the queries and the predicted outputs, respectively. The pixel $w_{i,j}$ (i: row, j: column) denotes the attention weight of the j-th query for the i-th output. Red indicates stronger attention. The attention output is normalized from 0 to 1.

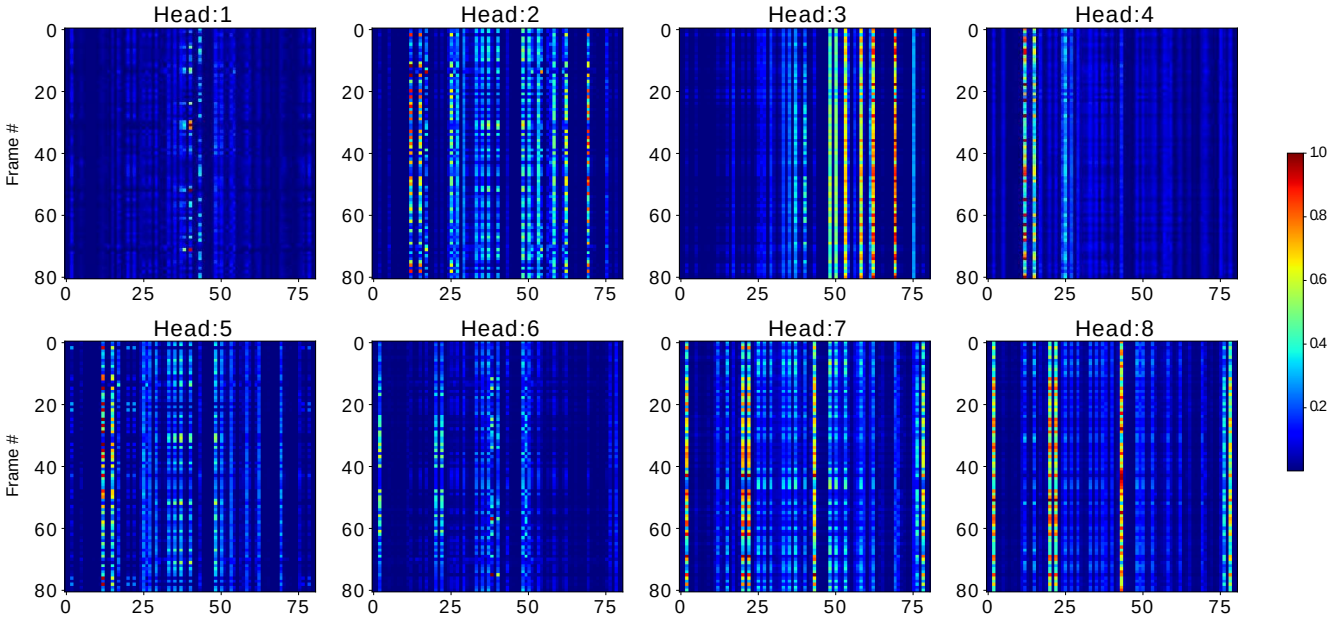

Figure 8. Visualization of self-attentions in temporal transformer. The x-axis (horizontal) and y-axis (vertical) correspond to the queries and the predicted outputs, respectively. The pixel $w_{i,j}$ (i: row, j: column) denotes the attention weight of the j-th query for the i-th output. Red indicates stronger attention. The attention output is normalized from 0 to 1.

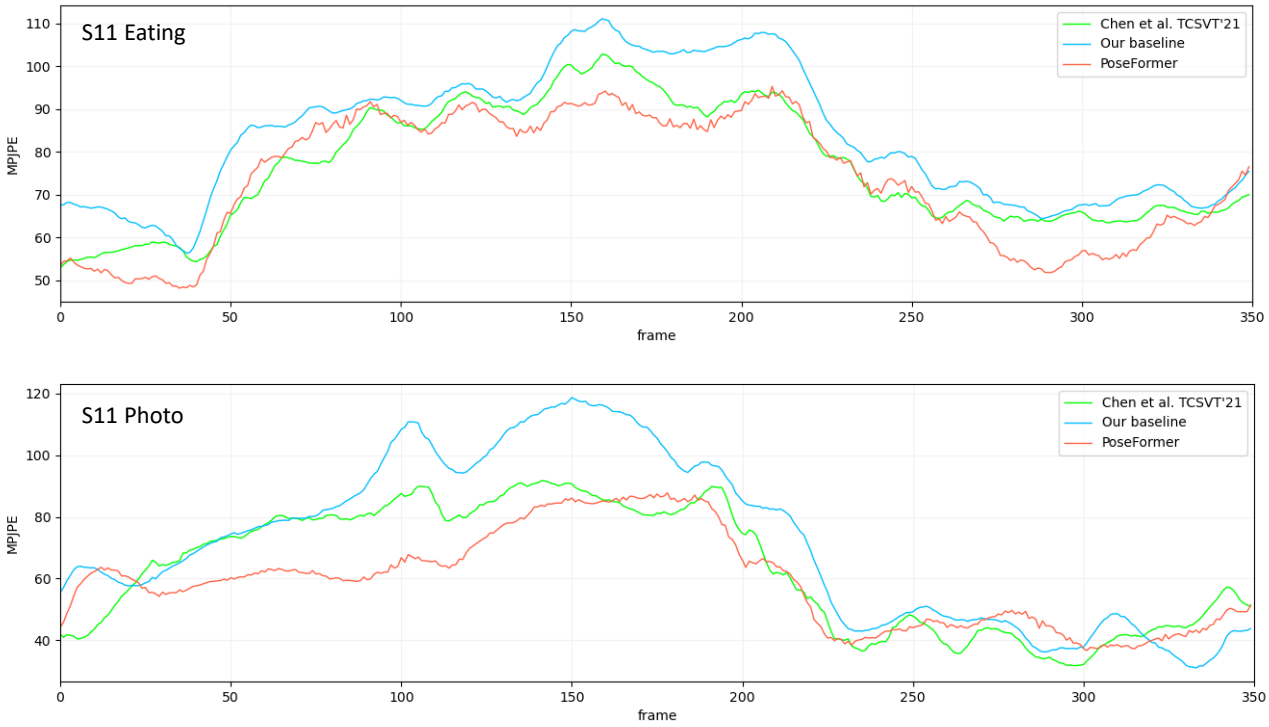

Figure 9. Frame-wise comparison between our method (PoseFormer), our baseline, and the SOTA approach Chen *et al*. [5] on Human3.6M test set. Top-figure: S11 with the *Eating* action. Bottom-figure: S11 with the *Photo* action.

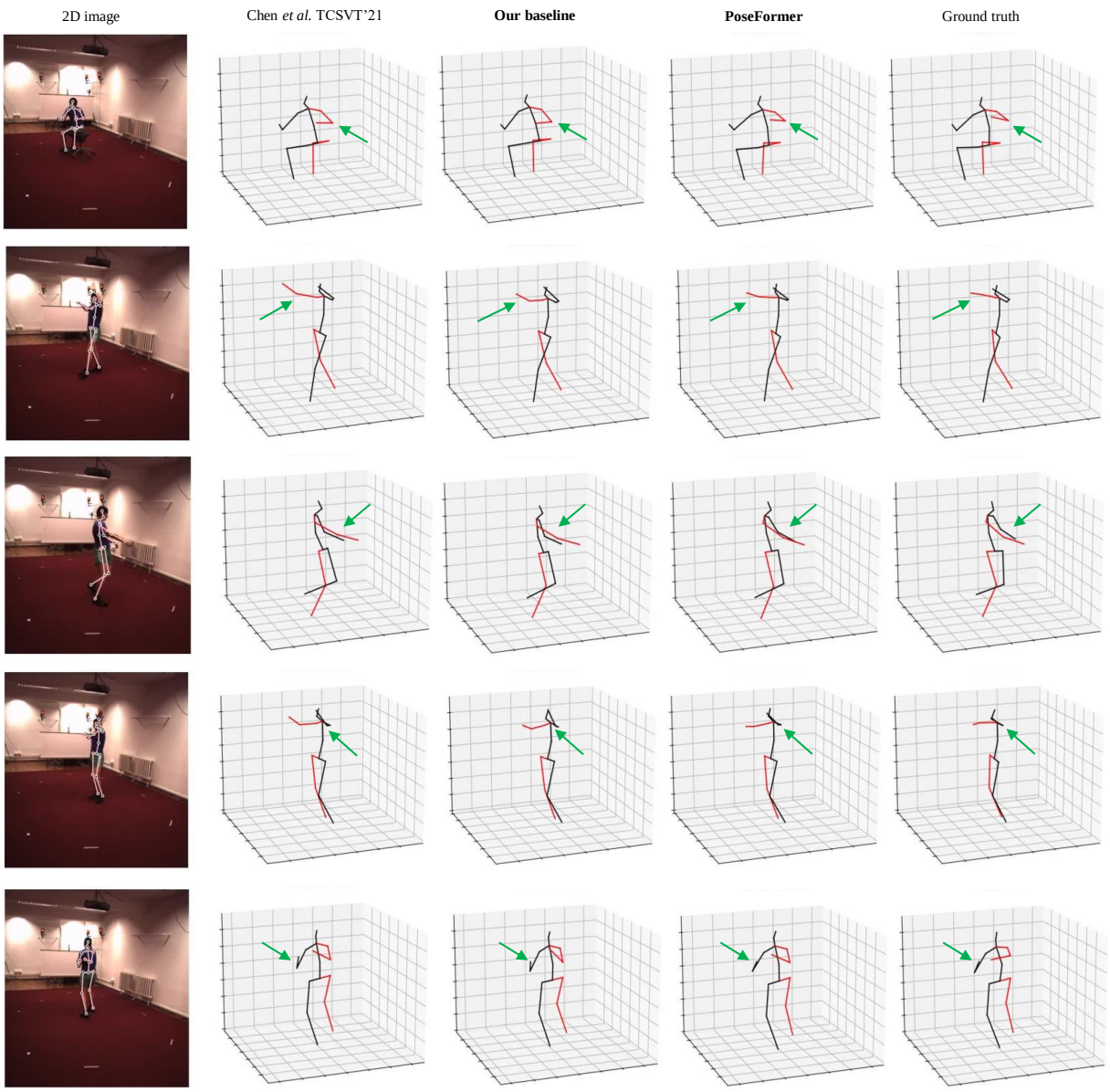

Figure 10. Qualitative comparison between our method (PoseFormer), our baseline, and the SOTA approach Chen *et al*. [5] on Human3.6M test set S11 with the *Greeting* and *WalkDog* actions. The green arrows highlight locations where PoseFormer clearly has better results.

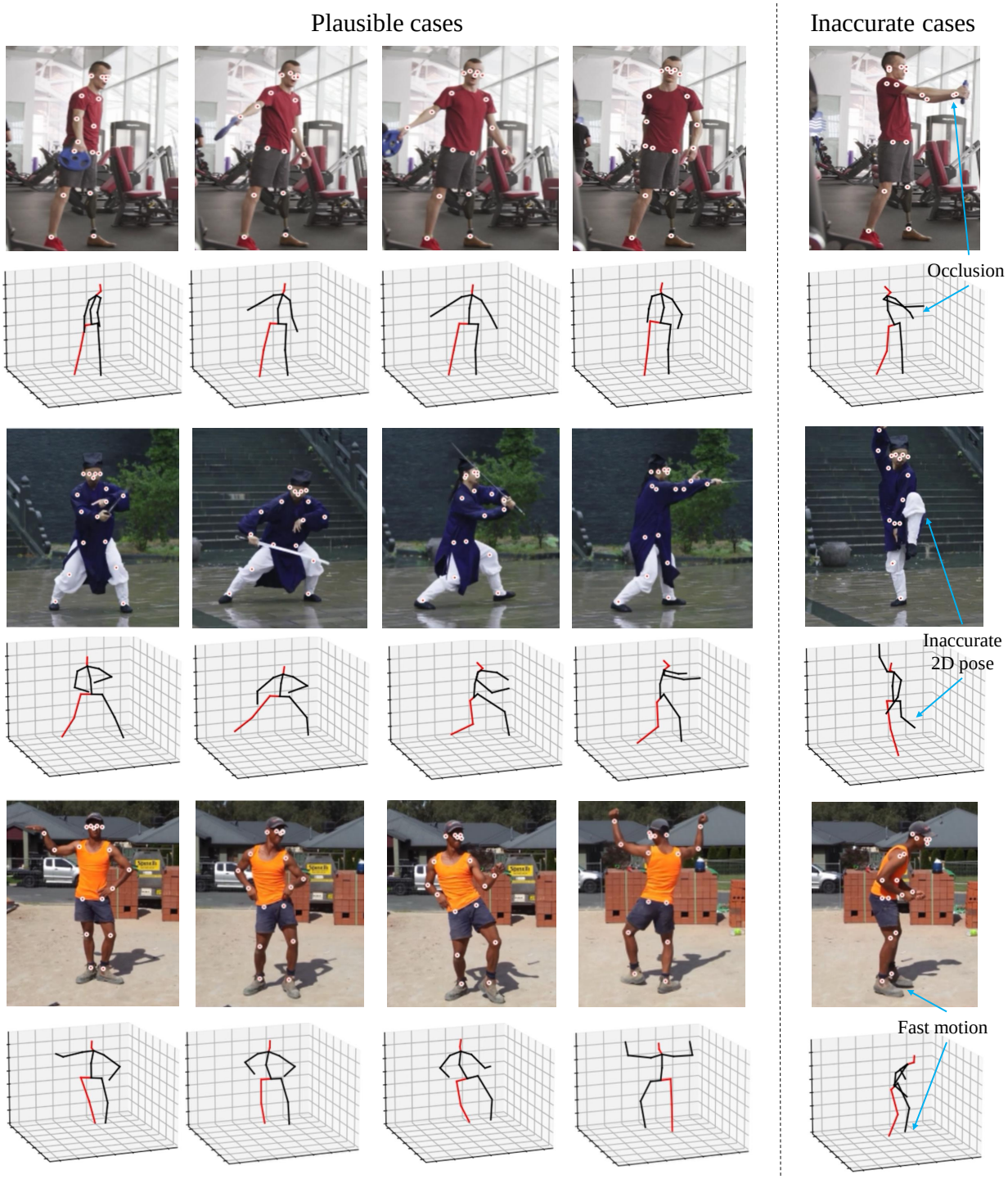

Figure 11. Qualitative results on in-the-wild videos: original frame sequence with detected 2D joints and the recovered 3D poses using PoseFormer.