Hourglass Tokenizer for Efficient Transformer-Based 3D Human Pose Estimation –Supplemental Material–

Wenhao Li[1] Mengyuan Liu[1*] Hong Liu[1] Pichao Wang[2†] Jialun Cai[1] Nicu Sebe[3]
[1]National Key Laboratory of General Artificial Intelligence, Peking University, Shenzhen Graduate School
[2]Amazon Prime Video [3]University of Trento
{wenhaoli,liumengyuan,hongliu}@pku.edu.cn pichaowang@gmail.com
cjl@stu.pku.edu.cn niculae.sebe@unitn.it

This supplementary material covers the following details:

- A brief description of video pose transformers (Sec. A).
- Computation complexity of transformers (Sec. B).
- Additional implementation details (Sec. C).
- Additional quantitative results (Sec. D).
- Additional ablation studies (Sec. E)
- Additional visualization results (Sec. F).

A. Video Pose Transformers

Recent studies of video pose transformers (VPTs) [2, 4, 6–9] are mainly designed to estimate 3D poses from 2D pose sequences. These VPTs share a similar architecture, which includes a pose embedding module (often containing only a linear layer) to embed spatial and temporal information of pose sequences, a stack of transformer blocks to learn global spatio-temporal correlations, and a regression module to predict 3D human poses. We summarize the architecture in Figure A. There are two types of pipelines based on their outputs: the *seq2frame* pipeline outputs the 3D poses of all frames, while the *seq2seq* pipeline outputs the 3D pose of the center frame.

B. Computation Complexity

Each transformer block consists of a multi-head self-attention (MSA) layer and a feed-forward network (FFN) layer. Let N be the number of tokens, D be the dimension, and $2D$ be the expanding dimension in the FFN (the expanding ratio in VPTs is typically 2). The calculational costs of MSA and FFN are $\mathcal{O}\left(4ND^2 + 2N^2D\right)$, and $\mathcal{O}\left(4ND^2\right)$, respectively. Thus, the total computational complexity is $\mathcal{O}\left(8ND^2 + 2N^2D\right)$, which makes VPTs computationally expensive. Since the dimension D is important to determine the modeling ability and most recent VPTs employ a D of 512 or 256, we follow their hyperparameter settings and propose to prune pose tokens of video frames (*i.e.*, reducing N) to reduce the computational cost of VPTs.

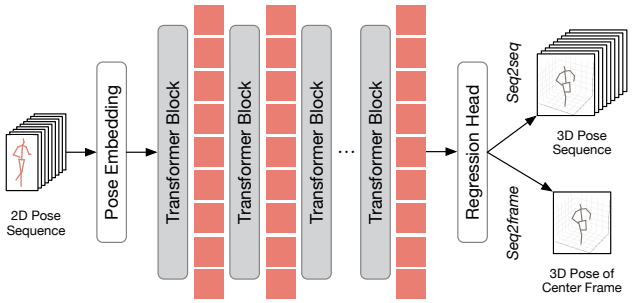


Figure A. Summary of VPT architectures. Existing VPTs typically contain a pose embedding module, a stack of transformer blocks, and a regression head module. The outputs of the regression head can be either the 3D poses of all frames for the *seq2seq* pipeline or the 3D pose of the center frame for the *seq2frame* pipeline.

Table A. Implementation details of our method on MHFormer [3], MixSTE [7], and MotionBERT [9]. (L) - number of transformer blocks, (C) - dimension, (LR) - initial learning rate, (Flip) - horizontal flip augmentation, (CPN) - Cascaded Pyramid Network [1], (SH) - Stack Hourglass [5].

Config	MHFormer [3]	MixSTE [7]	MotionBERT [9]
L	3	8	5
C	512	512	256
Training Epoch	20	160	120
Batch Size	210	4	4
LR	1×10^{-3}	4×10^{-5}	5×10^{-4}
Optimizer	Amsgrad	Adam	Adam
Augmentation	Flip	Flip	Flip
2D Detector	CPN	CPN	SH

C. Additional Implementation Details

Our method is built upon three very recent VPTs: MHFormer [3], MixSTE [7], and MotionBERT [9]. These VPTs achieve state-of-the-art performance but are computationally expen-

*Corresponding Author.
†The work does not relate to author's position at Amazon.

Table B. Comparison of GPU memory cost (G) and training time (min/epoch) on a single GeForce RTX 3090 GPU.

Method	GPU Memory	Training Time	MPJPE ↓
MHFormer [3]	24.1	223.2	43.0
TPC w. MHFormer	13.8 (↓ **42.7%**)	131.0 (↓ **39.7%**)	43.0
MixSTE [7]	11.4	17.0	40.9
HoT w. MixSTE	7.6 (↓ **33.3%**)	10.5 (↓ **38.2%**)	41.0
TPC w. MixSTE	7.3 (↓ **36.0%**)	10.1 (↓ **40.6%**)	40.4
MotionBERT [9]	10.7	17.4	39.8
HoT w. MotionBERT	6.1 (↓ **43.0%**)	8.9 (↓ **47.5%**)	39.8
TPC w. MotionBERT	5.7 (↓ **46.7%**)	8.4 (↓ **51.7%**)	**39.2**

Table C. Comparison with MixSTE.

Method	F	f	Param (M)	FLOPs (G)	MPJPE ↓
MixSTE [7]	81	81	33.70	92.42	42.7
MixSTE [7]	147	147	33.73	167.72	41.8
MixSTE [7]	243	243	33.78	277.25	40.9
HoT w. MixSTE	243	81	35.00	167.52	41.0
TPC w. MixSTE	243	81	33.78	161.73	**40.4**

sive compared to previous methods (see Table 6 in the main paper). We choose these VPTs as baselines to evaluate our method, which focuses on preserving the ability to model spatio-temporal dependencies while reducing computational costs. We adopt most of the optimal hyperparameters and training strategies used in [3, 7, 9], as shown in Table A. We also use the same loss functions for training, such as MPJPE loss for MHFormer, and weighted MPJPE loss, temporal consistency loss (TCLoss), and mean per-joint velocity error (MPJVE) for MixSTE.

Since our TRA is designed for *seq2seq* pipeline, it is unnecessary to add it to the model which is designed for *seq2frame* pipeline (*e.g.*, MHFormer). To provide a comprehensive analysis of our method, we report results with TPC and with both TPC and TRA. We denote the resulting models as follows:

- HoT w. MixSTE (MixSTE + TPC + TRA),
- HoT w. MotionBERT (MotionBERT + TPC + TRA),

which are designed for *seq2seq* pipeline, and:

- TPC w. MHFormer (MHFormer + TPC),
- TPC w. MixSTE (MixSTE + TPC),
- TPC w. MotionBERT (MotionBERT + TPC),

which are designed for *seq2frame* pipeline.

D. Additional Quantitative Results

Training Memory Cost and Training Time. To demonstrate the superiority of deploying our boosted VPTs on resource-limited devices, we report the training GPU memory cost and training time per epoch in Table B (directly using their training settings). Besides, we report the results of our method using the default settings, *i.e.*, $\{F{=}351, n{=}1, f{=}117\}$ for MHFormer, $\{F{=}243, n{=}3, f{=}81\}$ for MixSTE, and $\{F{=}243, n{=}1, f{=}81\}$ for MotionBERT. The results show that our method significantly reduces the GPU memory cost and training time while achieving superior results. For instance, by equipping with HoT, MotionBERT achieves a memory cost reduction of 43.0% and a training time reduction of 47.5% while maintaining the same performance.

Table D. Ablation study on the number of recovered tokens (f') under *seq2frame* pipeline.

Method	Param (M)	FLOPs (G)	MPJPE ↓
MixSTE [7]	33.78	277.25	40.9
HoT w. MixSTE ($f'{=}9$)	34.88	163.33	40.9
HoT w. MixSTE ($f'{=}27$)	34.89	163.66	40.7
HoT w. MixSTE ($f'{=}81$)	34.92	164.62	40.9
HoT w. MixSTE ($f'{=}243$)	35.00	167.52	40.9

Computation Complexity and Accuracy. In our main paper, we mainly report the results to show that our method can reduce FLOPs while achieving highly competitive or even better results (Tables 1, 2, 3, and 6 of the main paper). Here, we compare our method with MixSTE using the same number of representative tokens and approximately the same number of FLOPs. To achieve this, we set the input frame number of the original MixSTE to $F{=}81$ and $F{=}147$, respectively. The results in Table C show that our method obtains better results under both settings, further demonstrating the importance of large receptive fields and the effectiveness of our method.

E. Additional Ablation Study

Number of Recovered Tokens. In Table D, we conduct the ablation study on the number of recovered tokens (f') under *seq2frame* pipeline. Since f' differs from the input frames, we evaluate the performance under the *seq2frame* pipeline, which selects the 3D pose of the center frame as the final estimation. The results show that reducing f' slightly decreases the number of parameters, but the performance remains almost unchanged. Therefore, we choose $f'{=}243$, which is more efficient and can be evaluated under the *seq2seq* pipeline.

Hyperparameters (n and f). In Tables 2 and 3 of our main paper, we conduct ablation studies on the block index of representative tokens (n) under the *seq2frame* pipeline and on the number of representative tokens (f) under the *seq2seq* pipeline, respectively. To systematically explore the hyperparameters, we further conduct the ablation studies on n under the *seq2seq* pipeline (Table E) and on f under the *seq2frame* pipeline (Table F). It shows that we can flexibly adjust the values of n and f to achieve a speed-accuracy trade-off that meets the specific demands of real-world applications.

Table E. Ablation study on the block index of representative tokens (n) under *seq2seq* pipeline.

Method	Param (M)	FLOPs (G)	FPS	GPU Memory (G)	Training Time	MPJPE ↓
MixSTE [7]	33.78	277.25	10432	11.4	17.0	40.9
HoT w. MixSTE, n=1	35.00	121.31 (↓ **56.3%**)	20374 (↑ **95.3%**)	6.0 (↓ **47.4%**)	7.8 (↓ **54.1%**)	41.8
HoT w. MixSTE, n=2	35.00	144.42 (↓ **47.9%**)	17724 (↑ **69.9%**)	6.8 (↓ **40.4%**)	9.2 (↓ **45.9%**)	41.6
HoT w. MixSTE, n=3	35.00	167.52 (↓ **39.6%**)	15770 (↑ **51.2%**)	7.6 (↓ **33.3%**)	10.5 (↓ **38.2%**)	**41.0**
HoT w. MixSTE, n=4	35.00	190.62 (↓ **31.2%**)	14094 (↑ **35.1%**)	8.5 (↓ **25.4%**)	12.0 (↓ **29.4%**)	41.4
HoT w. MixSTE, n=5	35.00	213.72 (↓ **22.9%**)	12801 (↑ **22.7%**)	9.3 (↓ **18.4%**)	13.2 (↓ **22.4%**)	41.7
HoT w. MixSTE, n=6	35.00	236.82 (↓ **14.6%**)	11673 (↑ **11.9%**)	10.0 (↓ **12.3%**)	14.7 (↓ **13.5%**)	41.6
HoT w. MixSTE, n=7	35.00	259.93 (↓ **06.3%**)	10791 (↑ **03.4%**)	10.9 (↓ **04.4%**)	16.0 (↓ **05.9%**)	41.5

Table F. Ablation study on the number of representative tokens (f) under *seq2frame* pipeline. Here, * denotes the result without re-training.

Method	Param (M)	FLOPs (G)	FPS	GPU Memory (G)	Training Time	MPJPE*	MPJPE ↓
MixSTE [7]	33.78	277.25	43	11.4	17.0	40.7	40.7
TPC w. MixSTE, f=9	33.78	110.39 (↓ **60.2%**)	89 (↑ **107.0%**)	5.8 (↓ **49.1%**)	7.2 (↓ **57.6%**)	44.1	41.5
TPC w. MixSTE, f=16	33.78	115.38 (↓ **58.4%**)	88 (↑ **104.7%**)	5.8 (↓ **49.1%**)	7.5 (↓ **55.9%**)	42.7	41.0
TPC w. MixSTE, f=27	33.78	123.23 (↓ **55.6%**)	84 (↑ **95.3%**)	6.1 (↓ **46.5%**)	7.9 (↓ **53.5%**)	41.8	40.5
TPC w. MixSTE, f=61	33.78	147.47 (↓ **46.8%**)	75 (↑ **74.4%**)	7.0 (↓ **38.6%**)	9.3 (↓ **45.3%**)	41.2	40.5
TPC w. MixSTE, f=81	33.78	161.73 (↓ **41.7%**)	68 (↑ **58.1%**)	7.3 (↓ **36.0%**)	10.1 (↓ **40.6%**)	41.1	40.4
TPC w. MixSTE, f=121	33.78	190.26 (↓ **31.4%**)	62 (↑ **44.2%**)	8.3 (↓ **27.2%**)	11.7 (↓ **31.2%**)	40.9	40.4
TPC w. MixSTE, f=135	33.78	200.24 (↓ **27.8%**)	58 (↑ **34.9%**)	8.5 (↓ **25.4%**)	12.4 (↓ **27.1%**)	**40.9**	**40.2**

F. Additional Visualization Results

Selected Tokens. In Figure 7 of our main paper, we provide statistics visualization of selected tokens by taking some samples of consecutive video frames as input with a temporal interval of 1 between neighboring samples. For more comprehensive observation, we further statistically visualize selected tokens of different token pruning strategies using random samples (temporal interval is set to 243), *i.e.*, the neighboring samples have no overlapping frames. The frame indexes and frequency count of frame indexes of selected tokens are shown at the top and bottom of Figure B. The visualization figure of motion pruning (Figure B (c)) shows the most significant changes compared to Figure 7 of our main paper. The reason for this is that the random samples do not contain consecutive motion information. Interestingly, the visualization figures of frame indexes between TPC and motion pruning are somewhat similar but our TPC selects more tokens for the center frame. Besides, the performance of motion pruning is much worse than our TPC due to noise frames (see Table 4 of main paper).

Cluster Groups. The visualization in Figure C depicts cluster groups corresponding to varying numbers of representative tokens (f). We observe that the cluster primarily groups neighboring tokens into the same group, as the nearby poses are similar. Moreover, it also groups some tokens that are relatively distant from each other into the same group based on their feature similarity.

3D Pose Reconstruction. Figure D presents the qualitative comparison among the proposed HoT w. MixSTE and TPC w. MixSTE, and MixSTE [7] on Human3.6M dataset. Furthermore, Figure E shows the qualitative results on challenging in-the-wild videos. These results confirm the ability of our method to produce accurate 3D pose estimations. However, in challenging scenarios, there are some failure cases where our method cannot accurately estimate 3D human poses due to factors such as partial body visibility, rare poses, and significant errors in the 2D detector (Figure F). We also provide visualizations of recovering 3D human poses in Figure G, which illustrate that our method can predict realistic 3D human poses of the entire sequence, thereby further demonstrating the effectiveness of the proposed TRA.

References

[1] 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 *CVPR*, pages 6173–6183, 2020. 1

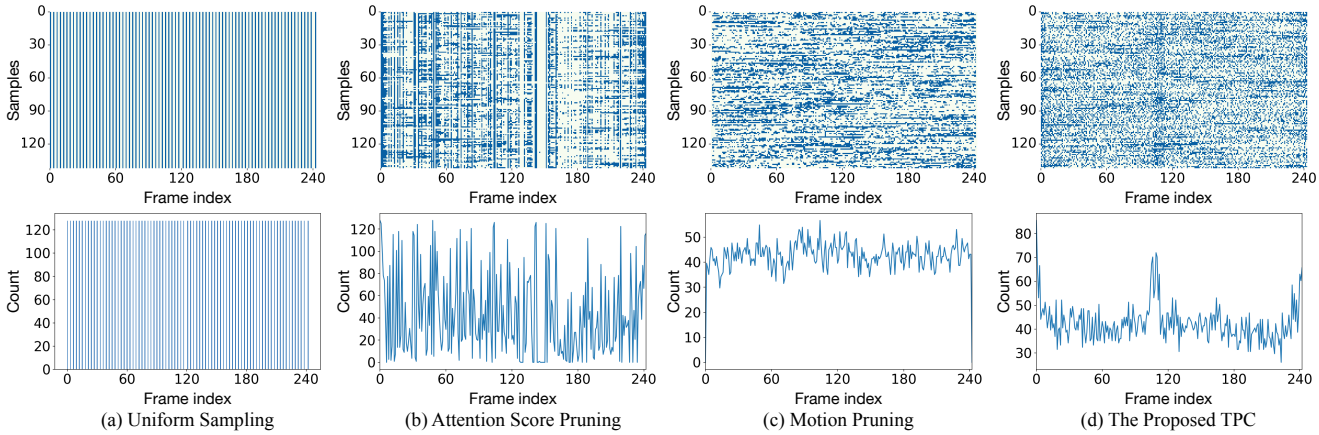
[2] Wenhao Li, Hong Liu, Runwei Ding, Mengyuan Liu, Pichao Wang, and Wenming Yang. Exploiting temporal contexts with strided transformer for 3D human pose estimation. *IEEE TMM*, 25:1282–1293, 2022. 1

[3] Wenhao Li, Hong Liu, Hao Tang, Pichao Wang, and Luc Van Gool. MHFormer: Multi-hypothesis transformer for 3D human pose estimation. In *CVPR*, pages 13147–13156, 2022. 1, 2

[4] Wenhao Li, Hong Liu, Hao Tang, and Pichao Wang. Multi-hypothesis representation learning for transformer-based 3D human pose estimation. *PR*, 141:109631, 2023. 1

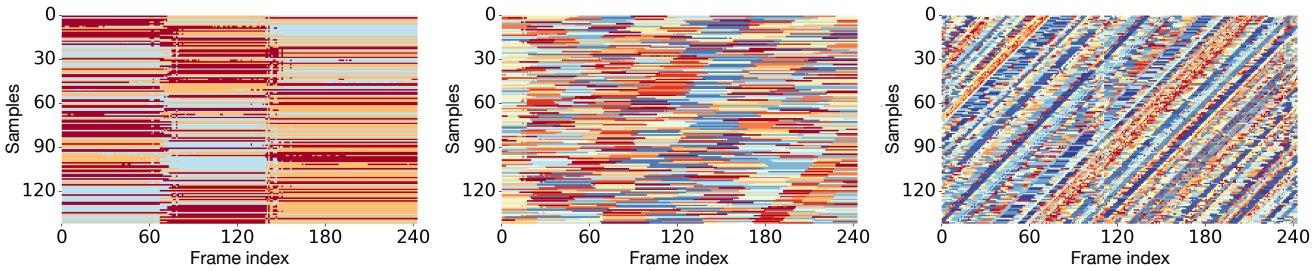
[5] Alejandro Newell, Kaiyu Yang, and Jia Deng. Stacked hourglass networks for human pose estimation. In *ECCV*, pages 483–499, 2016. 1

[6] Wenkang Shan, Zhenhua Liu, Xinfeng Zhang, Shanshe Wang, Siwei Ma, and Wen Gao. P-STMO: Pre-trained spatial tem-



(a) Uniform Sampling (b) Attention Score Pruning (c) Motion Pruning (d) The Proposed TPC

Figure B. Statistics visualization of selected tokens for different token pruning strategies. **Top**: Frame indexes of selected tokens for some samples (140 samples) of video sequences (243 frames). Blue points represent selected tokens and white points represent pruned tokens. **Bottom**: Frequency count of frame indexes of selected tokens for these samples.



(a) $f = 3$ (b) $f = 9$ (c) $f = 81$

Figure C. Visualization of cluster groups for the different numbers of representative tokens f. In each row, points of the same color represent the same cluster group.

poral many-to-one model for 3D human pose estimation. In *ECCV*, 2022. 1

[7] 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 *CVPR*, pages 13232–13242, 2022. 1, 2, 3, 5

[8] Ce Zheng, Sijie Zhu, Matias Mendieta, Taojiannan Yang, Chen Chen, and Zhengming Ding. 3D human pose estimation with spatial and temporal transformers. In *ICCV*, pages 11656–11665, 2021.

[9] Wentao Zhu, Xiaoxuan Ma, Zhaoyang Liu, Libin Liu, Wayne Wu, and Yizhou Wang. MotionBERT: A unified perspective on learning human motion representations. In *ICCV*, pages 15085–15099, 2023. 1, 2

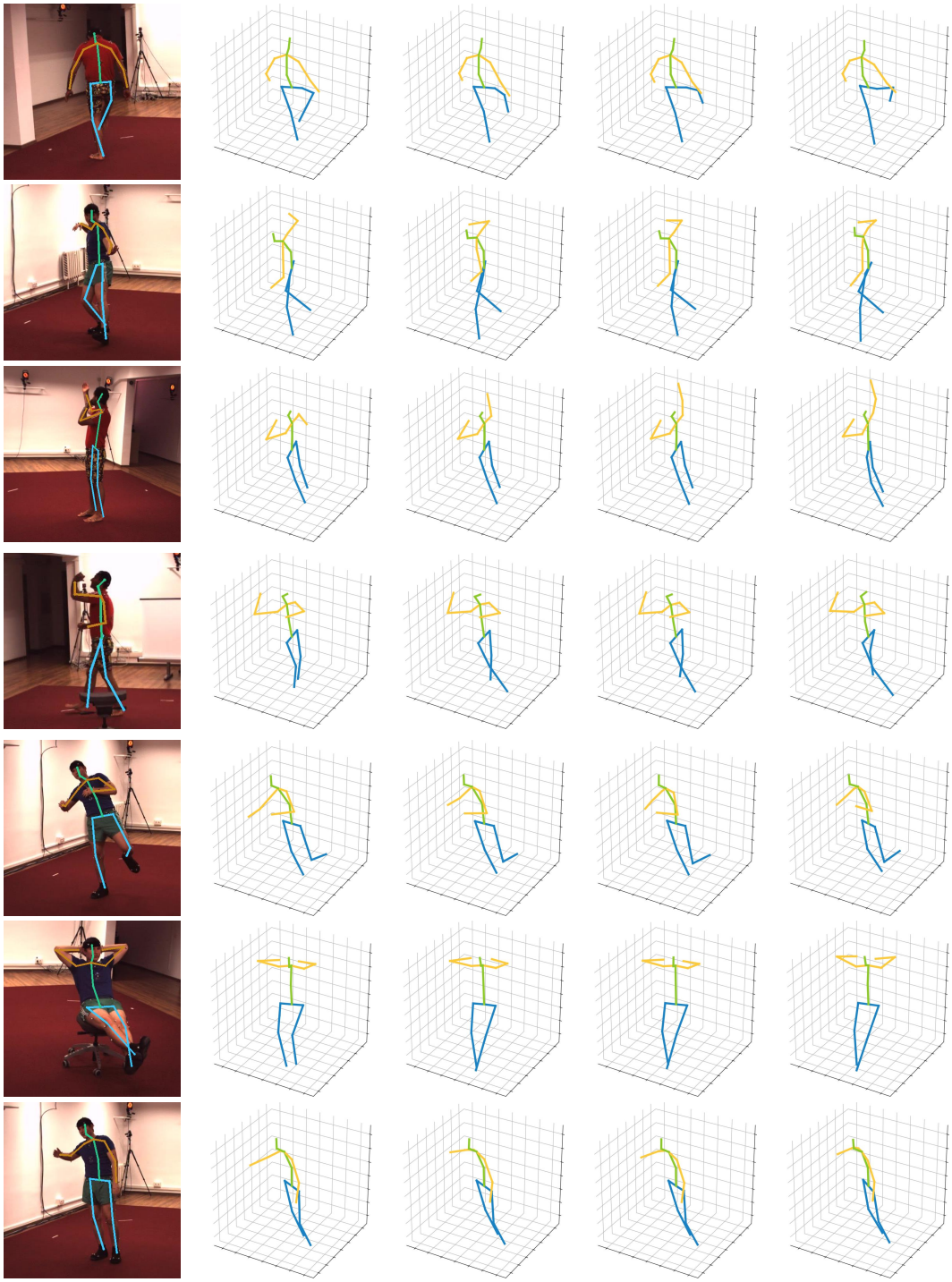


Figure D. Qualitative comparison among the previous state-of-the-art method (MixSTE [7]), our HoT w. MixSTE, and our TPC w. MixSTE on Human3.6M dataset.

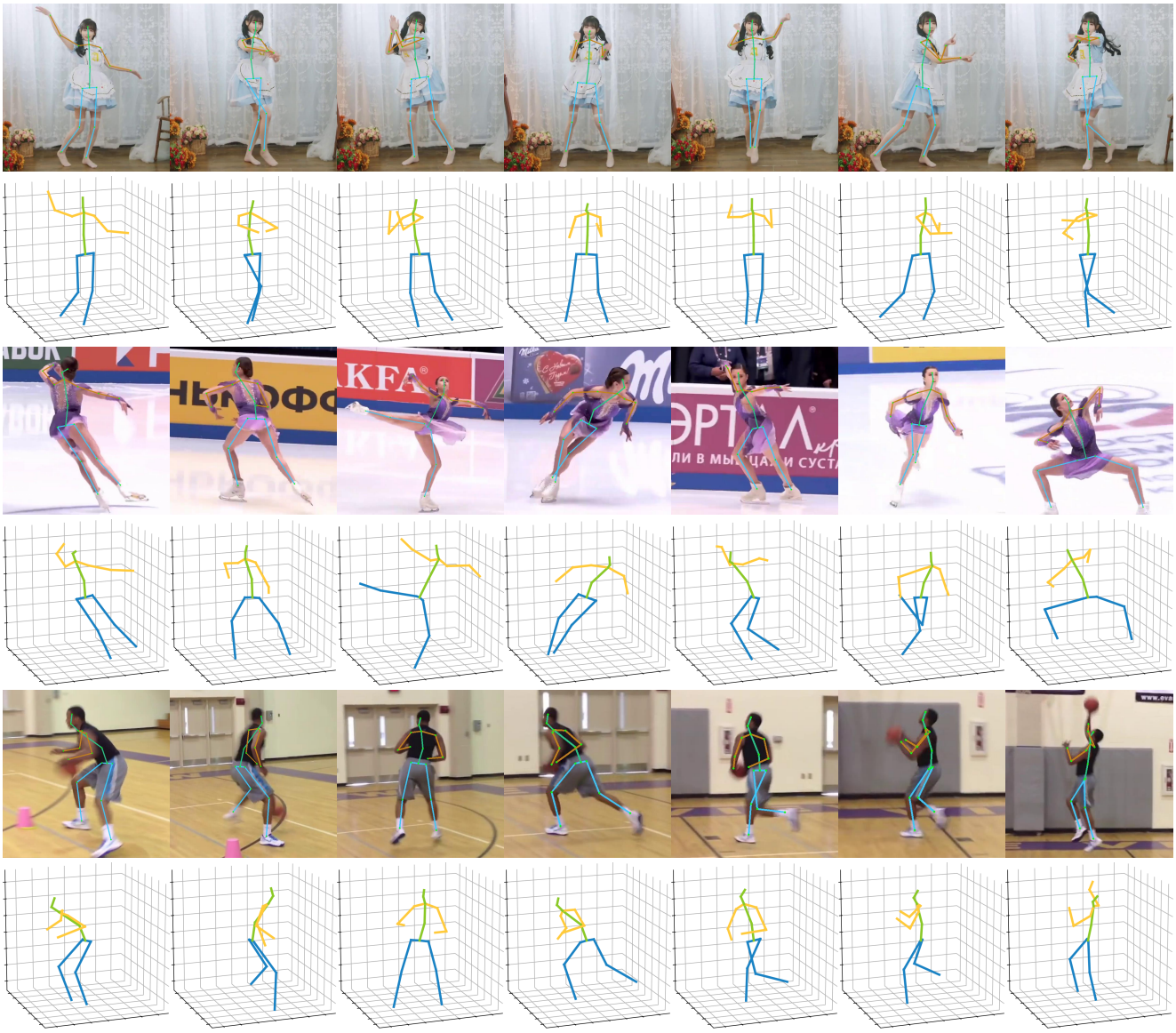
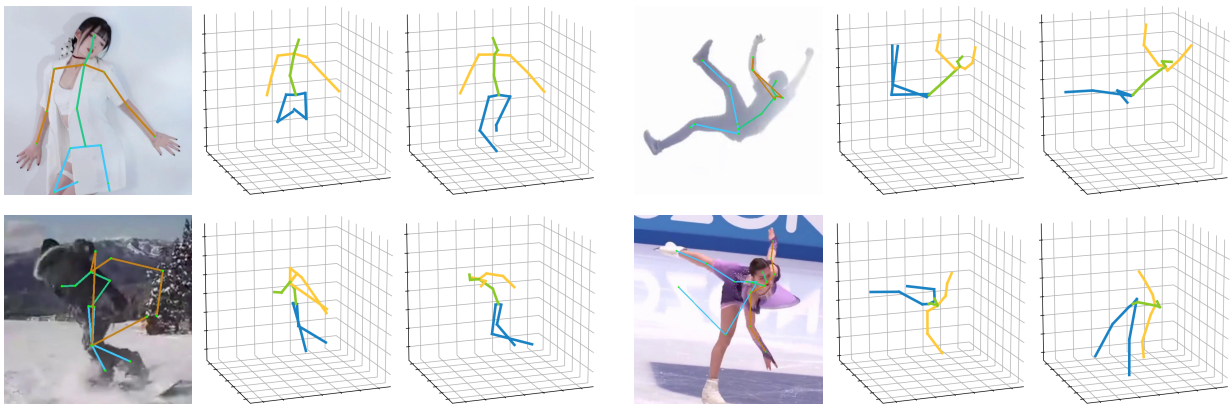


Figure E. Qualitative results of our method on challenging in-the-wild videos.



Input MixSTE **HoT w. MixSTE** Input MixSTE **HoT w. MixSTE**

Figure F. Failure cases in challenging scenarios.

Figure G. Visualization of input images, estimated 3D poses (cyan), and ground truth 3D poses (black) from three video sequences. The 2D poses of selected frames are colored in red, and the 2D poses of pruned frames are colored in gray. The 3D poses of selected frames are highlighted with red rectangular boxes.