—— Supplementary Material ——
MixSynthFormer: A Transformer Encoder-like Structure with Mixed Synthetic Self-attention for Efficient Human Pose Estimation

1. Computation of Attention Matrix Synthesis

This section presents a detailed computation calculation of the *SynthAttenOP*. We use the same notation in the main paper, and the input has dimension $\mathbb{R}^{N\times D}$. In practice, r_{se} is set to 4 or 8, making the cost from *SELayer* very small. We ignore the computation from *SELayer* in the following computation calculation.

The operations in the attention block can be separated into two steps: attention calculation, which computes the attention weights, and feature fusion, which multiplies the attention weights with values. Table 1 compares the computation in the standard transformer with *SynthAttenOP* attention generation.

In a standard transformer, the pairwise dot-product attention costs D^2. However, with our synthetic attention weights generation, the cost is reduced to DN, a reduction of $\frac{D}{N}$ times. Typically, the number of tokens N is smaller than embedding dimension D. By introducing the reduction factor r, where $d = \lfloor N/r \rfloor$, the generation cost is reduced to $Dd + Nd$ and the fusion with value reduced from N^2 to d^2. In our experiments, r is set to 4 or 8, which saves a considerable amount of computation.

Table 1: Computation in attention matrix generation

Method	Atten Calculation	Fusion
Vanilla	D^2	N^2
SynthAttenOp	DN	N^2
SynthAttenOp with r	$Dd + Nd$	d^2

2. Dataset and Pose Estimator Descriptions

We use different datasets for different pose estimation tasks. For 2D pose estimation, we use Sub-JHMDB [4] with SimplePose [16] as the estimator. For 3D pose estimation, we use Human3.6M [3] with FCN [12] as the estimator.For 3D body recovery (SMPL-based [10]), we use 3DPW [15] with SPIN [8], EFT [5] and PARE [7] as estimators, and AIST++ [9] estimated by SPIN [8]. Below we present the descriptions of these datasets and estimators.

2.1. Datasets

We utilize the following datasets for pose estimation. Human3.6M is also used in motion prediction.

Sub-JHMDB. JHMDB [4] is a dataset for 2D human pose estimations containing 316 short video clips with an average of 35 frames. We conduct experiments on its subset Sub-JHMDB. Poses are annotated by 15 keypoints. The bounding box is calculated from the puppet mask provided by [11]. We combine the original splitting schemes during experiments, consistent with previous works [18, 1, 17].

Human3.6M. Human3.6M [3] is a large-scale indoor dataset with 15 actions from four camera viewpoints. It comprises 3.6 million frames with 17 annotated joints in each frame. We follow previous works [12, 17], and train with the subjects S1, S5, S6, S7, S8, while the subjects S9 and S11 are used for testing.

3DPW. 3DPW [15] is the first in-the-wild dataset containing videos captured from moving phone cameras. It consists of 60 video sequences with accurate pose annotations and is usually used as the testing set for body recovery methods.

AIST++. AIST++ [9] is a dancing dataset constructed from the AIST Dance Video Database [14]. It contains diverse and fast-moving poses. It includes 3D keypoint annotations and SMPL [10] parameters for 10.1 million images, covering 30 actors in nine views. We adopt the same split setting as [17] in experiments.

2.2. Pose Estimators

To be comparable with [17], we use the following single-frame pose estimators for the detection of keyframe poses. However, in practice, more lightweight pose estimators can be used to further speed up the inference.

SimplePose. SimplePose [16] is a baseline model for 2D pose estimation and pose tracking. It uses ResNet [2] as backbone and incorporates deconvolutional layers.

FCN. FCN [12] is an MLP-based 2D-to-3D lifting model that operates along the spatial dimension. The model estimates 3D poses from 2D joint detections, making it a simple yet effective option.

SPIN. SPIN [8] combines SMPL [10] optimization in the training process. It collaborates optimization and regression techniques to better handle human body recovery tasks.

EFT. EFT [5] is trained on augmented 2D datasets with high-quality 3D pose fits and has better generalization ability than SPIN.

PARE. PARE [7] is an occlusion-robust human pose and shape estimator that can handle partial occlusion with the usage of the part-guided attention mechanism.

3. Implementation and Training Details

We set different Q for different datasets. Sub-JHMDB contains shorts video clips, and the Q is set to 1. For long videos, Q is set to 10 by default. Specially, 3DPW are in-the-wide videos which may contain many occluded and shaking cases, so we set the Q to be 5. Table 2 shows all the parameters.

In addition, inside FFN of *MixSynthEncoder*, the expansion ratio is set to 2, and a dropout layer with a rate of 0.1 is added to prevent overfitting. r_{se} in *SELayer* is set to 4 for all datasets. We use Adam [6] as the optimizer with an initial learning rate of 0.001, decayed by 0.97 after each epoch. All models are trained for 60 epochs.

Table 2: Training parameter settings for different datasets. C, Q, L represent the embedding dimension, the number of keyframes, and the number of encoder blocks respectively. r_t and r_s are the reduction factors used in temporal and spatial attention matrix synthesis. Batch stands for the training batch size. Interp means the preliminary recovery method, which can be a traditional interpolator or a learned interpolator. Linear and NN means the interpoaltion is done by a linear interpolator and a linear layer respectively.

Dataset	C	Q	L	r_t	r_s	Batch	Interp
Sub-JHMDB	128	1	5	1	8	16	Linear
H36M	64	10	4	4	1	256	NN
PW3D	32	5	4	1	1	256	Linear
AIST++	128	10	5	4	8	512	NN

4. Experiments

4.1. Inference Time

We evaluate the computation costs and inference time of different models on various datasets, using the settings presented in Table 2. For CPU inference, we use an 8-core CPU Apple M1 Pro chip. For GPU inference, we use NVIDIA GeForce GTX 1080 Ti. Results are presented in Table 3.

As shown in the table, models using linear interpolation for the preliminary recovery have similar inference speeds on both CPU and GPU. In contrast, models using NN interpolation run much faster on GPU than on CPU. Models using a linear layer for interpolation runs less than 0.15ms per frame on a CPU, which suggests that our model can be integrated with real-time applications on resource-constrained devices.

Table 3: Parameters, computation and inference time on GPU and CPU per frame

Dataset	#param	FLOPs	T_{GPU}	T_{CPU}
Sub-JHMDB	0.48M	0.45M	0.51ms	0.65ms
H36M	0.21M	0.12M	0.05ms	0.09ms
PW3D	0.07M	0.03M	0.09ms	0.09ms
AIST++	0.58M	0.48M	0.06ms	0.15ms

4.2. Ablation Study

Different Sampling Strategies. In the main paper, we use uniform sampling in all experiments. Here we extend our analysis by examining the performance of the model trained on uniformly sampled data using three additional sampling strategies: (i) random sampling (Random): randomly select keyframes; (ii) uniform-random (U-R): divide the whole sequence into equal-length intervals and randomly select one frame in each interval; (iii) random with first and last frame (R-FL): select the first and last frames of the entire sequence and randomly select middle keyframes. Table 4 shows the results.

Despite being trained on uniformly sampled data, *MixSynthFormer* demonstrates robust performance with both U-R and R-FL sampling. This highlights its ability to recover and refine poses even with varying short intervals between keyframes. However, randomly sampled sequences may suffer from long invisible periods that can negatively impact performance.

Table 4: Different sampling strategies on 3DPW estimated by PARE (MPJPE 78.9/ Accel 25.7).

Strategy	MPJPE	Accel
Uniform	76.5	**6.7**
Random	80.5	11.0
U-R	**75.9**	8.9
R-FL	76.9	7.2

Different Interpolation Methods. The performance of different interpolators can vary based on the dataset being used. We employ the single-frame pose estimator SPIN to evaluate the effect of interpolators on 3DPW and AIST++. Table 5 presents the result.

The results indicate that different interpolators may be more suitable for specific datasets. *MixSynthFormer* performs better on 3DPW with a linear interpolator, while a learned interpolator produces better results on AIST++. The choice of interpolator may depend on the types of actions

performed in each dataset. Since 3DPW involves daily life actions, a linear interpolator suffices for initial recovery. Conversely, AIST++ involves complex dancing motions, which can be better recovered by a learned layer that can better learn the motion prior.

Table 5: Different Interpolation methods on 3DPW and AIST++ estimated by SPIN. NN means the interpolation is done by a linear layer.

Dataset	Interp	MPJPE	Accel
3DPW	NN	92.0	8.1
	Linear	**91.2**	**6.8**
AIST++	NN	**71.2**	**4.7**
	Linear	71.9	5.5

4.3. Generalization Ability

We conducted experiments to evaluate the generalization ability of *MixSynthFormer* on different datasets, using various interpolators and pose estimators. The testing model is trained on 3DPW with keyframe poses estimated by PARE. Linear interpolation is used during training.

In the cross-interpolator tests, as Table 6 shows, we find that our model can learn the patterns of human motions to refine coarsely-recovered sequences, and changing the interpolator does not significantly affect the performance. Surprisingly, we obtained a smaller acceleration error than in training when using quadratic interpolation.

Table 6: Cross-interpolator results

Interpolation	MPJPE	Accel
Linear	**76.5**	6.7
Quadratic	**76.5**	**6.2**
Cubic-spline	76.7	6.3

Table 7 shows the results of cross-dataset and cross-backbone tests. *MixSynthFormer* can reduce the acceleration error by around 80% for all tested datasets. With the exception of a slight increase in MPJPE in 3DPW estimated by SPIN, it can reduce the localization error in other datasets. Therefore, *MixSynthFormer* has the potential to serve as a highly-efficient smoothing tool for new data without compromising accuracy.

4.4. Motion Prediction

Table 8 presents the average MPJPE for different actions in the short-term motion prediction task. Four testing intervals (80, 160, 320 and 400 ms) are used.

Our method outperforms both baselines for all actions in the extremely short intervals (80 and 160 ms). The near future actions are rather predictable, and *MixSynthFormer* can make relatively accurate predictions through the recover-refine mechanism. It also performs well on complex actions containing self-occlusion, such as sitting and taking photos. However, as the prediction interval enlarges, our model fails to predict high-frequency actions, such as posing and discussion, due to the overly-smooth coarse prediction. Nevertheless, *MixSynthFormer* is still competitive with state-of-the-art motion prediction models.

Table 7: Cross-dataset and cross-backbone results

Dataset (Estimator)	MPJPE	Accel
3DPW (SPIN)	**96.9**	34.7
	97.7	**6.8**
3DPW (EFT)	90.3	32.8
	87.1	**6.9**
AIST++ (SPIN)	107.7	33.8
	100.5	**5.9**

5. Limitations and Future Work

The limitations and potential improvements of *MixSynthFormer* can be analyzed from both the refinement model design and application perspective. Limitations in the model design are primarily related to synthetic attention operations (*SynthAttenOp*):

▷ The number of parameters and computational complexity of the model are dependent on the input window size due to the usage of linear layers.

▷ Attention weights are synthesized by a linear layer which has a fixed number of weights. When presented with a sequence of different sizes from training, attention matrices need to be truncated, similar to the learned position embeddings in transformers.

▷ The current framework generates only one attention matrix for each branch, equivalent to one-head in standard self-attention. To enhance model robustness, it is possible to generate multiple attention matrices and combine the results, as in multi-head attention.

▷ Current framework does not manually emphasize the importance of keyframes, and the inter-frame relation is learned by the network. Future research could explore keyframe-based operations to enhance the influence of keyframes and improve performance.

Our framework incorporates an existing single-frame pose estimator for keyframe pose estimation. From an application perspective, our framework has the following shortcomings that can be improved upon:

▷ The performance of the current framework partially relies on the quality of estimated poses. Even though *MixSynthFormer* has the ability to correct some incorrectly detected joints. It fails to produce correct joints if the error exceeds its ability of refining. How to ensure the quality of the

Table 8: Average MPJPE on short-term motion prediction in Human3.6M

Action	Walking				Eating				Smoking				Discussion			
	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400
STSGCN [13]	10.7	16.8	29.1	38.2	6.7	11.3	22.6	31.6	7.1	11.6	22.3	30.6	9.7	16.7	33.4	45.0
STGAGCN [19]	10.3	16.1	28.8	32.4	6.4	11.5	21.7	25.2	7.1	11.8	21.7	24.3	9.7	17.1	31.4	**38.9**
Ours	**8.1**	**14.7**	**25.3**	**29.3**	**5.3**	**10.1**	**19.4**	**23.3**	**5.6**	**10.4**	**20.0**	**24.1**	**8.0**	**15.7**	**31.5**	39.2
Action	Directions				Greeting				Phoning				Posing			
	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400
STSGCN [13]	7.4	13.5	29.2	40.9	12.4	21.7	42.1	54.5	8.2	13.7	26.8	36.6	9.9	18.0	38.2	52.6
STGAGCN [19]	7.3	12.8	30.3	34.5	11.8	20.1	40.5	48.4	8.8	13.5	25.5	**28.7**	10.1	17.0	**35.5**	**45.1**
Ours	**5.7**	**12.1**	**27.5**	**33.5**	**11.2**	**20.9**	**40.0**	**48.2**	**6.8**	**12.7**	**24.6**	30.3	**11.1**	**16.5**	35.7	46.5
Action	Purchasing				Sitting				Sitting Down				Taking Photo			
	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400
STSGCN [13]	11.9	21.3	41.9	54.8	9.1	15.1	29.8	39.8	14.4	23.7	41.9	53.8	8.1	14.1	29.7	41.9
STGAGCN [19]	11.9	20.7	41.8	**47.6**	9.3	14.4	29.6	38.5	14.1	24.8	40.0	**47.4**	8.5	13.9	28.8	35.1
Ours	**10.3**	**20.6**	**40.2**	49.8	**7.8**	**14.2**	**27.5**	**33.7**	**13.2**	**22.4**	**39.4**	**47.4**	**6.5**	**13.0**	**26.0**	**34.6**
Action	Waiting				Walking Dog				Walking Together				Average			
	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400
STSGCN [13]	8.6	14.7	29.6	40.7	17.6	29.3	52.6	66.4	8.6	14.3	26.5	35.1	10.1	17.1	33.1	38.3
STGAGCN [19]	8.5	14.1	29.8	33.8	17.0	28.8	50.1	**59.4**	-	-	-	-	10.1	16.9	32.5	38.5
Ours	**6.8**	**13.1**	**27.0**	**32.9**	**15.7**	**28.6**	**49.5**	**59.4**	**6.8**	**12.7**	**23.6**	**27.5**	**8.4**	**15.8**	**30.5**	**37.3**

keyframes is a key problem. One possible solution is to use accurate single-frame pose estimators or to select "good" frames in the sequence.

▷ In this work, we demonstrate the adaptability of the model on motion prediction tasks. future research could explore how to combine multiple more motion synthesis tasks into one.

6. Qualitative Results

6.1. Visualization of Synthetic Attention

We present visualizations of the general and reduced synthetic attention matrices in Figure 1 and 2 respectively. The first row of each figure shows the synthetic temporal attention weights, and the second row shows the synthetic spatial attention weights.

From Figure 1, we can see the temporal attention weights from the first two layers focus more on the first half of the sequence and the attention from layer 3 focus on the latter half. The attention in the last layer focuses on the whole sequence. The reduced temporal synthetic attention weights in Figure 2 are more concentrated. This is because important features are more likely to be forwarded incorporating the reduction factor. Both synthetic spatial attention matrices are sparse, only with a few highlights, as shown in the second row of figures.

Layer 1 to 4 from left to right

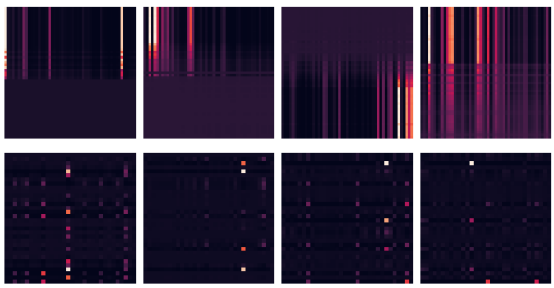

Figure 1: Visualization of synthetic attention weights on 3DPW. The first row shows temporal attention and the second row shows spatial attention.

Layer 1 to 4 from left to right

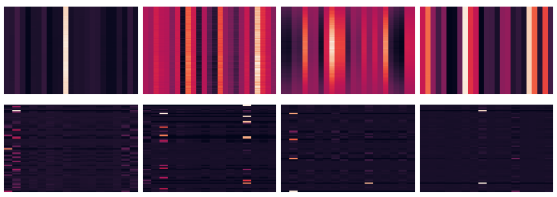

Figure 2: Visualization of reduced synthetic attention weights on Human3.6M. The first row shows reduced temporal attention and the second row shows reduced spatial attention.

6.2. Visualization of Pose Estimation

2D Pose Estimation. Figure 3 and 4 show the visualization results of the ground truth poses, poses detected by SimplePose and poses recovered and refined by *MixSynthFormer* respectively. The keyframes are highlighted with red boxes. From the results, there are two cases: (i) correct keyframe poses, where the proposed method leverages

Figure 3: Sub-JHMDB visualization - Golf. Poses in the red boxes are used for recovery. Keyframe poses are correctly estimated. *MixSynthFormer* can accurately recover the entire sequence by exploiting motion continuity.

Figure 4: Sub-JHMDB visualization - Push. Poses in the red boxes are used for recovery. Keyframe poses are noisy. *MixSynthFormer* can refine the noisy keyframe poses and recover the whole sequence effectively at the same time.

the motion continuity and generates accurate poses for the remaining frames; (ii) incorrect keyframe poses, where the proposed method can refine these poses, as demonstrated in Figure 4. These findings suggest that the proposed method has the ability to correct keyframe poses and enhance the overall performance of pose estimation.

3D Pose Estimation. The visualization results for the photo action in the Human3.6M dataset are presented in Figure 5. The poses recovered and refined by the proposed method exhibit a significantly lower acceleration error and slightly lower mean per joint position error (MPJPE) compared to FCN. Using FCN estimates poses frame by frame can cause jitter, which can be addressed through smoothing techniques such as interpolation. *MixSynthFormer* is based on interpolation and are effective in smoothing the sequence as linear interpolation. Additionally, *MixSynthFormer* has the ability to learn motion patterns during training, resulting in slightly better performance than linear interpolation.

3D Body Recovery. Figure 6, 7 and 8 show the visualization of different dance motions in the AIST++ dataset. The blue body indicates the estimated keyframe body shape and pose. Despite some incorrect detections in the keyframe poses (first row in figures), *MixSynthFormer* is able to refine the incorrect estimations from the arms and legs. Even though the single-frame estimator SPIN generates incorrect results between the two keyframes, our model estimates poses only utilizing keyframes in a recover-refine manner and is effective in generating natural-looking poses sequences with smooth transitions.

Smilarly, *MixSynthFormer* can use relatively accurate keyframe poses and recover missing poses by leveraging temporal redundancy on 3DPW dataset. In Figure 9, our model generates an intermediate hugging pose from the former and latter keyframes. Figure 10 demonstrates its ability of refining keyframe poses. The estimator wrongly estimates the hand positions, one hand going through the palm of the other, which is not plausible in real life. *MixSynthFormer* can refine errors in such cases.

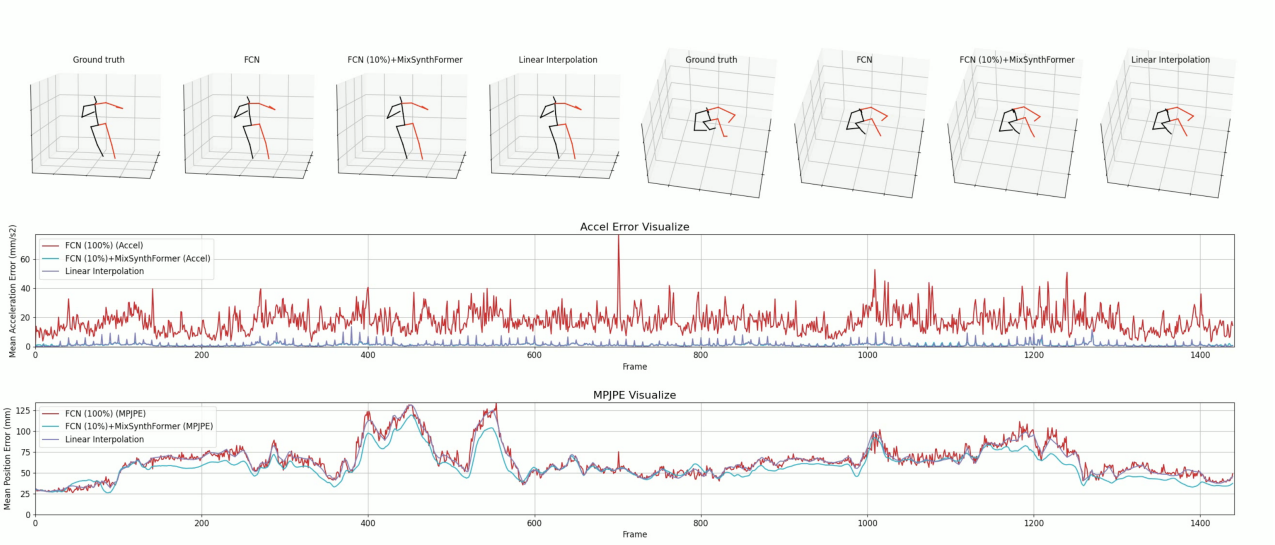

Figure 5: Human3.6M visualization - Photo. *MixSynthFormer* can produce smooth motions like interpolation but with lower MPJPE.

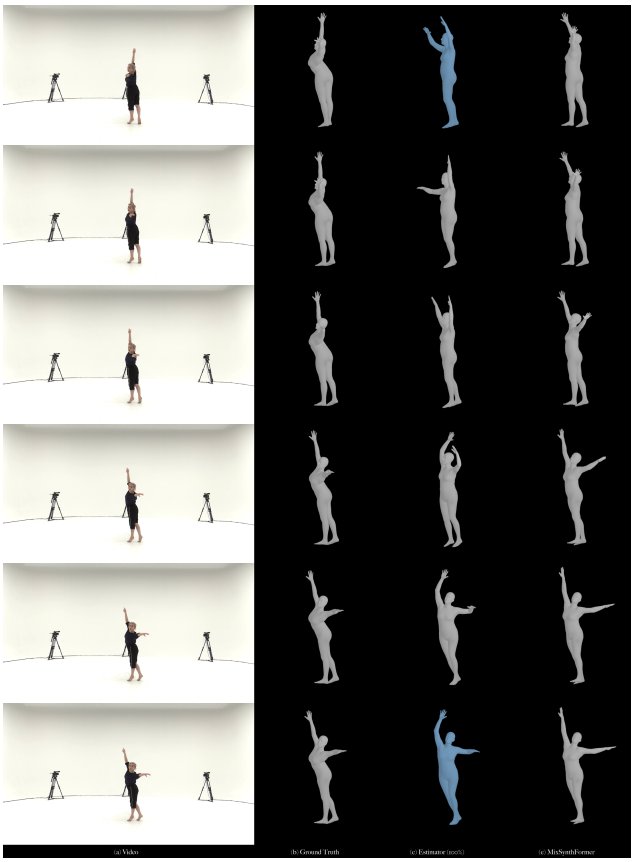

Figure 6: AIST++ visualization - Ballet Jazz. Body in blue are used for recovery. Three body mesh columns are from the ground truth data, SPIN estimation and our results respectively.

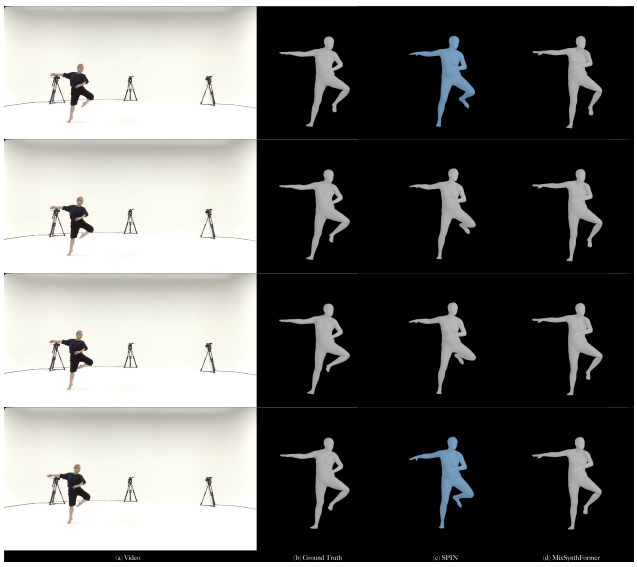

Figure 7: AIST++ visualization - Ballet Jazz. The left leg from SPIN estimation does not match the ground truth, which should be at a lower position.

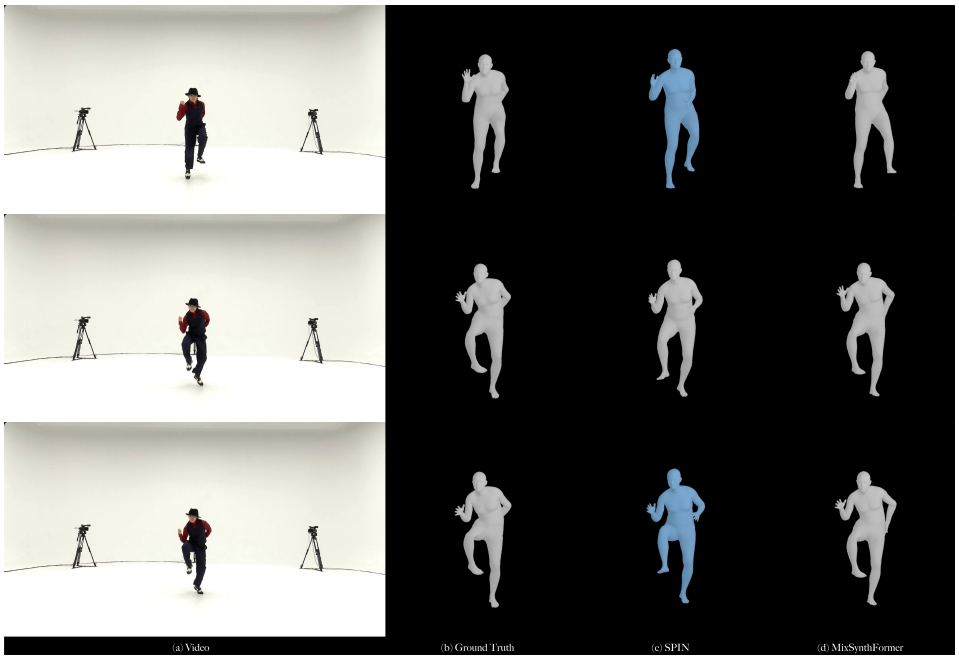

Figure 8: AIST++ visualization - Lock. The left hand from SPIN estimation is in front of the body. *MixSynthFormer* corrects it to the back of the body, as in ground truth.

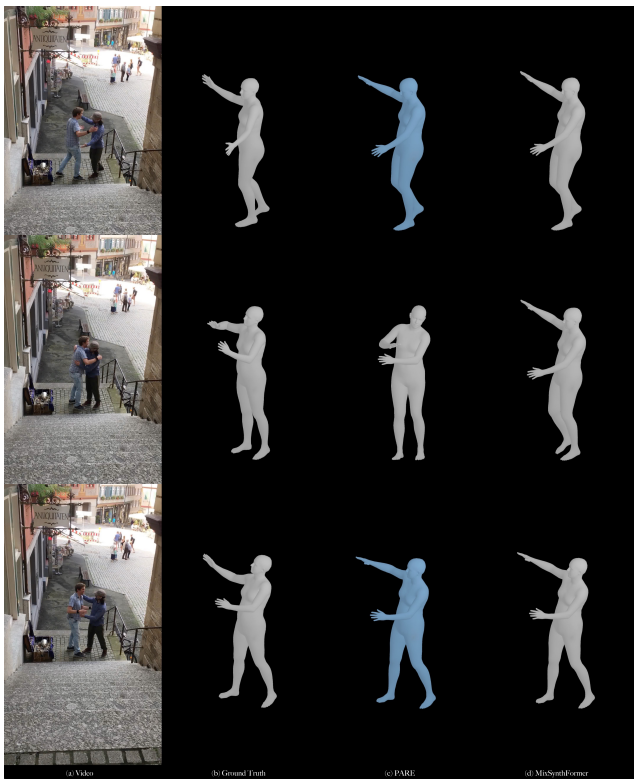

Figure 9: 3DPW visualization - Warm Welcome. Body in blue are used for recovery. Three body mesh columns are from the ground truth data, PARE estimation and our results respectively. PARE gives a wrong estimation of hugging.

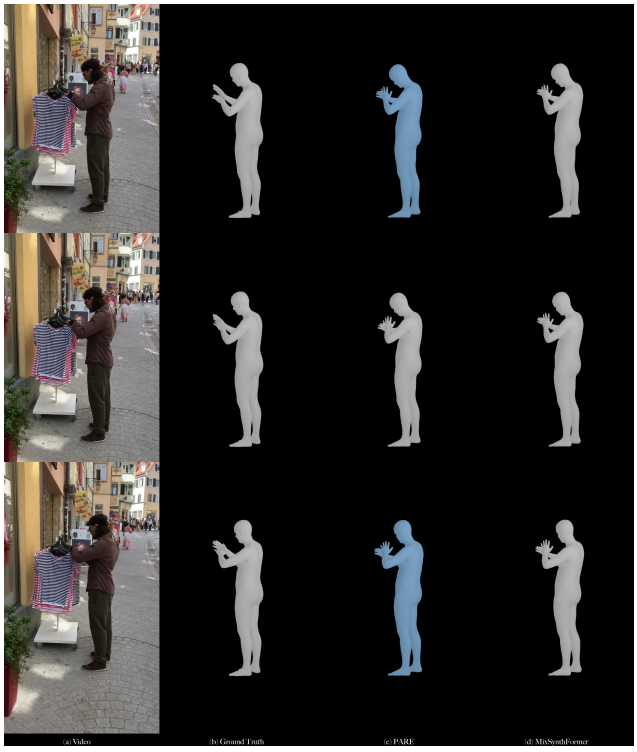

Figure 10: 3DPW++ visualization - Enter Shop. Hands in PARE estimation are intersected.

Figure 11: Failure case due to large keyframe pose estimation errors on AIST++

Figure 12: Failure case due to high-frequency motions on Sub-JHMDB

Failure Cases. While *MixSynthFormer* exhibits promising performance in our experiments, there were also instances of failure cases. For example, if the errors in the keyframes are too large, as shown in Figure 11, it may fail to generate correct pose sequences. In this specific case shown in Figure 11, the person is facing backward in the ground truth data, but the estimated keyframe poses are facing forward. Although *MixSynthFormer* has some ability to correct for errors, it was unable to generate the correct pose sequence in such cases with serve wrongly estimated results. Consequently, the generated poses are facing the side, indicating such cases exceed the model's ability to correct.

Another failure case we encountered in our experiments is related to the keyframe selection. Specifically, we sampled one frame out of every ten frames to save computation time. However, this lower sampling ratio also introduced more uncertainty in unsampled frames, as shown in Figure 12. In this case, the person in the video is running, but *MixSynthFormer* only generates the transition between the two given keyframes. Although the generated feet are still moving, they do not move as fast as in running. To address this issue, we suggest increasing the sampling ratio or adding more keyframes when necessary.

These failure cases highlight the need for developing more robust and accurate single-frame pose estimation models, particularly for sequences containing complex and rapid movements or uncommon dance actions.

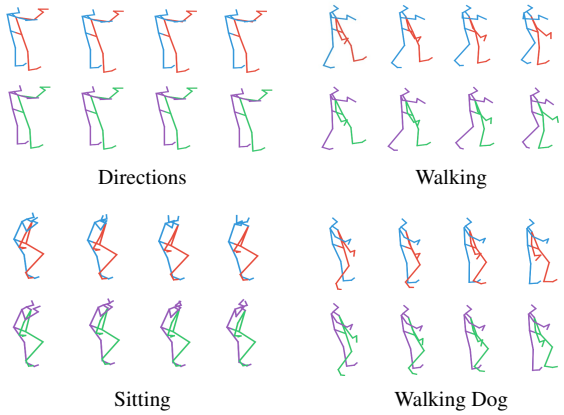

Figure 13: Visualization of four actions on the short-term motion prediction. Predictions are drawn in green and purple and ground truth poses are drawn in red and blue.

6.3. Visualization of Motion Prediction

The short-term motion prediction results, along with the corresponding ground truth poses, are presented in Figure 13. *MixSynthFormer* has the ability to learn motion patterns from the historical sequences and can generate smooth and plausible predictions. It is worth noting that short-term motion predictions are relatively predictable because the motion pattern can be easily captured in the given sequences. Our model has excellent performance in periodic motions such as walking and complex actions like sitting. Overall, the visualized results demonstrate the versatility and effectiveness of *MixSynthFormer*.

References

[1] Zhipeng Fan, Jun Liu, and Yao Wang. Motion adaptive pose estimation from compressed videos. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11699–11708, 2021. 1

[2] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. 1

[3] 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. 1

[4] Hueihan Jhuang, Juergen Gall, Silvia Zuffi, Cordelia Schmid, and Michael J Black. Towards understanding action recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3192–3199, 2013. 1

[5] Hanbyul Joo, Natalia Neverova, and Andrea Vedaldi. Exemplar fine-tuning for 3d human model fitting towards in-the-wild 3d human pose estimation. In *International Conference on 3D Vision (3DV)*, pages 42–52. IEEE, 2021. 1, 2

[6] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. 12 2014. 2

[7] Muhammed Kocabas, Chun-Hao P Huang, Otmar Hilliges, and Michael J Black. Pare: Part attention regressor for 3d human body estimation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11127–11137, 2021. 1, 2

[8] Nikos Kolotouros, Georgios Pavlakos, Michael J Black, and Kostas Daniilidis. Learning to reconstruct 3d human pose and shape via model-fitting in the loop. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2252–2261, 2019. 1, 2

[9] Ruilong Li, Shan Yang, David A. Ross, and Angjoo Kanazawa. Ai choreographer: Music conditioned 3d dance generation with aist++. In *Proceedings of the IEEE International Conference on Computer Vision*, 2021. 1

[10] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A skinned multi-person linear model. *ACM Trans. Graphics (Proc. SIGGRAPH Asia)*, 34(6):248:1–248:16, Oct. 2015. 1, 2

[11] Yue Luo, Jimmy Ren, Zhouxia Wang, Wenxiu Sun, Jinshan Pan, Jianbo Liu, Jiahao Pang, and Liang Lin. Lstm pose machines. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5207–5215, 2018. 1

[12] 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. 1

[13] Theodoros Sofianos, Alessio Sampieri, Luca Franco, and Fabio Galasso. Space-time-separable graph convolutional network for pose forecasting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 11209–11218, 2021. 4

[14] Shuhei Tsuchida, Satoru Fukayama, Masahiro Hamasaki, and Masataka Goto. Aist dance video database: Multi-genre, multi-dancer, and multi-camera database for dance information processing. In *Proceedings of the 20th International Society for Music Information Retrieval Conference, ISMIR 2019*, pages 501–510, Delft, Netherlands, Nov. 2019. 1

[15] Timo von Marcard, Roberto Henschel, Michael J. Black, Bodo Rosenhahn, and Gerard Pons-Moll. Recovering accurate 3d human pose in the wild using imus and a moving camera. In *Proceedings of the European Conference on Computer Vision*, September 2018. 1

[16] Bin Xiao, Haiping Wu, and Yichen Wei. Simple baselines for human pose estimation and tracking. In *Proceedings of the European Conference on Computer Vision*, pages 466–481, 2018. 1

[17] Ailing Zeng, Xuan Ju, Lei Yang, Ruiyuan Gao, Xizhou Zhu, Bo Dai, and Qiang Xu. Deciwatch: A simple baseline for 10x efficient 2d and 3d pose estimation. In *Proceedings of the European Conference on Computer Vision*. Springer, 2022. 1

[18] Yuexi Zhang, Yin Wang, Octavia Camps, and Mario Sznaier. Key Frame Proposal Network for Efficient Pose Estimation in Videos. In *Proceedings of the European Conference on Computer Vision*, August 2020. 1

[19] Chongyang Zhong, Lei Hu, Zihao Zhang, Yongjing Ye, and Shihong Xia. Spatio-temporal gating-adjacency gcn for human motion prediction. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6447–6456, 2022. 4