Back to MLP: A Simple Baseline for Human Motion Prediction

Wen Guo[1]*, Yuming Du[3]*, Xi Shen[4]†, Vincent Lepetit[3], Xavier Alameda-Pineda[1], Francesc Moreno-Noguer[2]
[1] Inria, Univ. Grenoble Alpes, CNRS, Grenoble INP, LJK, 38000 Grenoble, France
[2] Institut de Robòtica i Informàtica Industrial, CSIC-UPC, Barcelona, Spain
[3] LIGM, Ecole des Ponts, Univ Gustave Eiffel, CNRS, France [4] Tencent AI Lab
[1]{wen.guo,xavier.alameda-pineda}@inria.fr, [2]{fmoreno}@iri.upc.edu,
[3]{yuming.du,vincent.lepetit}@enpc.fr, [4]{shenxiluc}@gmail.com

Abstract

This paper tackles the problem of human motion prediction, consisting in forecasting future body poses from historically observed sequences. State-of-the-art approaches provide good results, however, they rely on deep learning architectures of arbitrary complexity, such as Recurrent Neural Networks (RNN), Transformers or Graph Convolutional Networks (GCN), typically requiring multiple training stages and more than 2 million parameters. In this paper, we show that, after combining with a series of standard practices, such as applying Discrete Cosine Transform (DCT), predicting residual displacement of joints and optimizing velocity as an auxiliary loss, a light-weight network based on multi-layer perceptrons (MLPs) with only 0.14 million parameters can surpass the state-of-the-art performance. An exhaustive evaluation on the Human3.6M, AMASS, and 3DPW datasets shows that our method, named siMLPe, *consistently outperforms all other approaches. We hope that our simple method could serve as a strong baseline for the community and allow re-thinking of the human motion prediction problem. The code is publicly available at* https://github.com/dulucas/siMLPe.

1. Introduction

Given a sequence of 3D body poses, the task of human motion prediction aims to predict the follow-up of the pose sequence. Forecasting future human motion is at the core of a number of applications, including preventing accidents in autonomous driving [46], tracking people [17], or human-robot interaction [27].

Due to the spatio-temporal nature of human motion, the common trend in the literature is to design models that are capable of fusing spatial and temporal information. Traditional approaches mainly relied on hidden Markov models [7] or Gaussian process latent variable models [60]. However, while these approaches performed well on simple and periodic motion patterns, they dramatically fail under complex motions [42]. In recent years, with the success of deep learning, various methods have been developed based on different types of neural networks that are able to handle sequential data. For example, some works use Recurrent Neural Networks (RNN) [43] to model the human motion [12, 15, 24, 34, 43], and some more recent

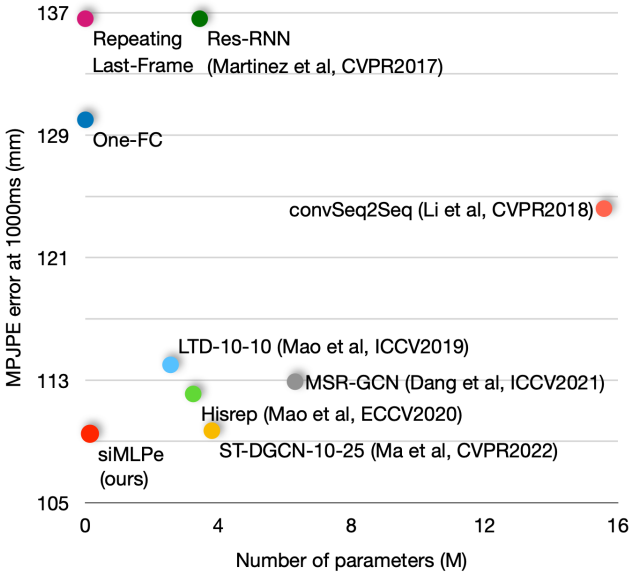

Figure 1. **Comparison of parameter size and performance on the Human3.6M dataset [23].** We report the MPJPE metric in *mm* at 1000 ms as performance on the vertical axis. The closer to bottom-left, the better. Our method (siMLPe, in red) achieves the lowest error with significantly fewer parameters. We also show the performance of two simple methods: 'Repeating Last-Frame' systematically repeats the last input frame as output prediction, and 'One-FC' uses only one single fully connected layer to predict the future motion.

*Equal contribution.
†Corresponding author.

works [13, 19, 32, 33, 38, 41, 42] propose networks based on Graph Convolutional Networks (GCN) [42], or trying with Transformers([2]) based method [2, 10, 41] to fuse the spatial and temporal information of the motion sequence across human joints and time. However, the architectures of these recent methods are usually not simple and some of them require additional priors, which makes their network difficult to analyze and modify. Thus, a question naturally arises: *"Can we tackle the human motion prediction with a simple network?"*

To answer this question, we first tried a naive solution by just repeating the last input pose and using them as the output prediction. As shown in Figure 1, this naive solution could already achieve reasonable results, which means the last input pose is "close" to the future poses. Inspired by this, we further train only one fully connected layer to predict the residual between the future poses and the last input pose and achieves better performance, which shows the potential of a simple network for human motion prediction built on basic layers like the fully connected layer.

Based on the above observations, we go back to the multi-layer perceptrons (MLPs) and build a simple yet effective network named SIMLPE with only three components: fully connected layers, layer normalization [4] and transpose operations. The network architecture is shown in Figure 2. Noticeably, we found that even commonly used activation layers such as ReLU [45] are not needed, which makes our network an entirely linear model except for layer normalization. Despite its simplicity, SIMLPE achieves strong performance when appropriately combined with three simple practices: applying the Discrete Cosine Transform (DCT), predicting residual displacement of joints, and optimizing velocity as an auxiliary loss.

SIMLPE yields state-of-the-art performance on several standard benchmarks, including Human3.6M [23], AMASS [39] and 3DPW [58]. In the meantime, SIMLPE is lightweight and requires $20\times$ to $60\times$ fewer parameters than previous state-of-the-art approaches. A comparison between SIMLPE and previous methods can be found in Figure 1, which shows the Mean Per Joint Position Error (MPJPE) at $1,000$ms on Human3.6M of different networks versus the network complexity. SIMLPE achieves the best performance with high efficiency.

In summary, our contributions are as follows:

- We show that human motion prediction can be modeled in a simple way without explicitly fusing spatial and temporal information. As an extreme example, a single fully connected layer can already achieve reasonable performance.

- We propose SIMLPE, a simple yet effective network for human motion prediction with only three components: fully connected layers, layer normalization, and transpose operation, achieving state-of-the-art performance with far fewer parameters than existing methods on multiple benchmarks such as Human3.6M, AMASS and 3DPW datasets.

2. Related Work

Human motion prediction is formulated as a sequence-to-sequence task, where past observed motion is taken as input to predict the future motion sequence. Traditional methods explore human motion prediction with nonlinear Markov models [30], Gaussian Process dynamical models [59], and Restricted Boltzmann Machine [51]. These approaches have shown to be effective to predict simple motions and eventually struggle with complex and long-term motion prediction [15]. With the deep learning era, human motion prediction has achieved great success with the use of deep networks, including Recurrent Neural Networks (RNNs) [12,15,24,34,43], Graph Convolutional Networks (GCNs) [13, 19, 32, 33, 38, 41, 42] and Transformers [2, 10, 41], which are the main focus of this section.

2.1. RNN-based human motion prediction

Due to the inherent sequential structure of human motion, some works address 3D human motion prediction by recurrent models. Fragkiadaki *et al*. [15] propose an encoder-decoder framework to embed human poses and an LSTM to update the latent space and predict future motion. Jain *et al*. [24] manually encode the semantic similarity between different parts of the body and forwards them via structural RNNs. However, these two methods suffer from discontinuity and they are only trained on action-specific models, *i.e*., a single model is trained for a specific action.

Martinez *et al*. [43] studied multi-actions instead of action-specific models, *i.e*., train one single model for multiple actions, which allows the network to exploit regularities across different actions in large-scale datasets. This is widely adopted by most of the subsequent works. They also introduced a residual connection to model the velocities instead of the absolute value to have more smooth predictions.

Nevertheless, the above-mentioned methods suffer from multiple inherent limitations of RNNs. First, as a sequential model, RNNs are difficult to parallelize during training and inference. Second, the memory constraints prevent RNNs from exploring information from farther frames. Some works alleviate this problem by using RNN variants [12,34], sliding windows [8,9], convolutional models [20,31] or adversarial training [18], as described in the following sections. But their networks are still complicated and have a large number of parameters.

2.2. GCN-based human motion prediction

To better encode the spatial connectivity of human joints, the most recent works usually build the human

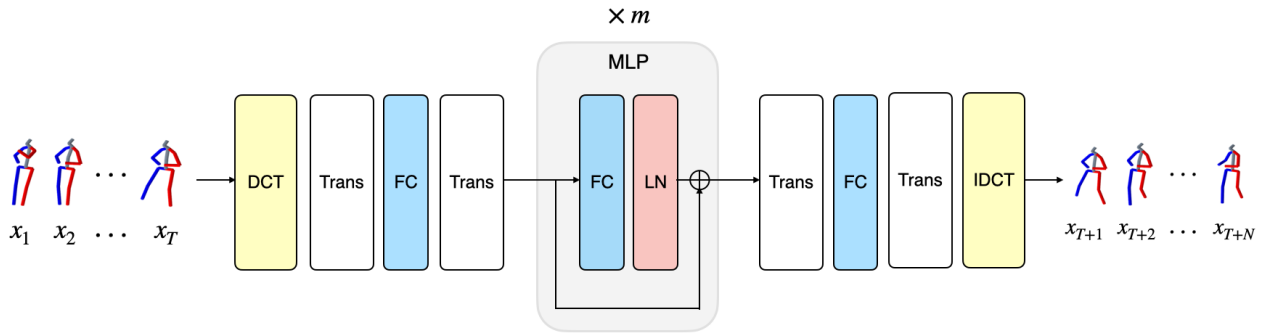

Figure 2. **Overview of our approach siMLPe for human motion prediction.** *FC* denotes a fully connected layer, *LN* denotes layer normalization [4], and *Trans* represents the transpose operation. *DCT* and *IDCT* represent the discrete cosine transformation and inverse discrete cosine transformations respectively. The MLP blocks (in gray), composing FC and LN, are repeated m times.

pose as a graph and adopt Graph Convolutional Networks (GCNs) [26, 50] for human motion prediction.

GCNs were first exploited for human motion prediction in Mao *et al*. [42]. They use a stack of blocks consisting of GCNs, nonlinear activation, and batch normalization to encode the spatial dependencies, and leverage discrete cosine transform (DCT) to encode temporal information. This work inspired most of the GCN-based motion prediction methods in recent years. Based on [42], Mao *et al*. [41] further improved the temporal encoding by cutting the past observations into several sub-sequences and adding an attention mechanism to find similar previous motion subsequences in the past with the current observations. Thus, the future sequence is computed as a weighted sum of observed sub-sequences. Then, a GCN-based predictor, the same as in [42], is used to encode the spatial dependencies.

Instead of using DCT transformation to encode the input sequence, [29] used a multi-scale temporal input embedding, by applying various size convolutional layers for different input sizes to have different receptive fields in the temporal domain. Ma *et al*. [38] proposed two variants of GCNs to extract spatial and temporal features. They built a multi-stage structure where each stage contains an encoder and a decoder, and during the training, the model is trained with intermediate supervision to learn to progressively refine the prediction. [13, 32, 33] extend the graph of human pose to multi-scale version across the abstraction levels of human pose.

2.3. Attention-based human motion prediction

With the development of transformers [57], some works [2, 10, 41] attempted to deal with this task with an attention mechanism. [41] used attention to find temporal relations; [2] also used attention to map not only the temporal dependencies but also the pairwise relation of joints by an architecture combining "spatial attention" and "temporal attention" in parallel. [10] used a transformer-based architecture along with a progressive-decoding strategy to predict the DCT coefficients of the target joints progressively based on the kinematic tree. In order to guide the predictions, they also built a memory-based dictionary to preserve the global motion patterns in training data.

In summary, with the development of human motion prediction in recent years, the RNN/GCN/transformer-based architectures are well explored and the results have been significantly improved. Though these methods provide good results, their architectures are becoming more and more complicated and difficult to train. In this paper, we stick to simple architectures and propose an MLP-based network. Recently, a concurrent and independent work [6] based on [52] also adopts an MLP based network architecture for motion prediction, while our network is much simpler as we do not use the squeeze-and-excitation block [22] nor the activation layers. We hope that our simple method would serve as a baseline and let the community rethink the problem of human motion prediction.

3. Our Approach: siMLPe

In this section, we formulate the problem and present the formulation of the DCT transformation in Section 3.1, details of the network architecture in Section 3.2 and the losses we use for training in Section 3.3.

Given a sequence of 3D human poses in the past, our goal is to predict the future sequence of poses. We denote the observed 3D human poses as $\mathbf{x}_{1:T} = [x_1^\top, .., x_T^\top]^\top \in \mathbb{R}^{T\times C}$, consisting of T consecutive human poses, where the pose at the t-th frame x_t is represented by a C-dimensional vector, *i.e.* $x_t \in \mathbb{R}^C$. In this work, similar to previous works [38, 41–43], x_t is the 3D coordinates of joints at t-th frame and $C = 3 \times K$, where K is the number of joints. Our task is to predict the future N motion frames $\mathbf{x}_{T+1:T+N} = [x_{T+1}^\top, .., x_{T+N}^\top]^\top \in \mathbb{R}^{N\times C}$.

3.1. Discrete Cosine Transform (DCT)

We adopt the DCT transformation to encode temporal information, which is proven to be beneficial for human mo-

Table 1. **Results on Human3.6M** for different prediction time steps (ms). We report the MPJPE error in *mm* and number of parameters (M) for each method. Lower is better. 256 samples are tested for each action. † indicates that the results are taken from the paper [41], ⋆ indicated that the results are taken from the paper [38]. Note that ST-DGCN [38] use two different models to evaluate their short-/long-term performance, here we report their results of a single model which performs better on long-term for fair comparison. We also show results of two simple baselines: 'Repeating Last-Frame' repeats the last input frame 25 times as output, 'One FC' uses only one single fully connected layer for the prediction.

	MPJPE (mm) ↓								# Param.(M) ↓
Time (ms)	80	160	320	400	560	720	880	1000	
Repeating Last-Frame	23.8	44.4	76.1	88.2	107.4	121.6	131.6	136.6	0
One FC	14.0	33.2	68.0	81.5	101.7	115.1	124.8	130.0	0.003
Res-RNN † [43]	25.0	46.2	77.0	88.3	106.3	119.4	130.0	136.6	3.44
convSeq2Seq † [31]	16.6	33.3	61.4	72.7	90.7	104.7	116.7	124.2	15.58
LTD-50-25 † [42]	12.2	25.4	50.7	61.5	79.6	93.6	105.2	112.4	2.56
LTD-10-10 † [42]	11.2	23.4	47.9	58.9	78.3	93.3	106.0	114.0	2.55
Hisrep † [41]	10.4	22.6	47.1	58.3	77.3	91.8	104.1	112.1	3.24
MSR-GCN ⋆ [13]	11.3	24.3	50.8	61.9	80.0	-	-	112.9	6.30
ST-DGCN-10-25 ⋆ [38]	10.6	23.1	47.1	57.9	76.3	90.7	102.4	109.7	3.80
siMLPe (Ours)	**9.6**	**21.7**	**46.3**	**57.3**	**75.7**	**90.1**	**101.8**	**109.4**	**0.14**

tion prediction [38, 41, 42]. More precisely, given an input motion sequence of T frames, the DCT matrix $\mathbf{D} \in \mathbb{R}^{T\times T}$ can be calculated as:

$$\mathbf{D}_{i,j} = \sqrt{\frac{2}{T}} \frac{1}{\sqrt{1+\delta_{i,0}}} \cos\left(\frac{\pi}{2T}(2j+1)i\right) , \quad (1)$$

where $\delta_{i,j}$ denotes the *Kronecker* delta:

$$\delta_{i,j} = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \, . \end{cases} \quad (2)$$

The transformed input is $\mathcal{D}(\mathbf{x}_{1:T}) = \mathbf{D}\mathbf{x}_{1:T}$. We apply the Inverse Discrete Cosine Transform (IDCT) to transform the output of the network back to the original pose representation, denoted as $\mathcal{D}^{-1}$ and the inverse of $\mathbf{D}$.

3.2. Network architecture

Figure 2 shows the architecture of our network. Our network only contains three components: fully connected layers, transpose operation, and layer normalization [4]. For all the fully connected layers, their input dimension is equal to their output dimension.

Formally, given an input sequence of 3D human poses $\mathbf{x}_{1:T} = [x_1^\top, .., x_T^\top]^\top \in \mathbb{R}^{T\times C}$, our network predicts a sequence of future poses $\mathbf{x'}_{T+1:T+N} = [x'^\top_{T+1}, .., x'^\top_{T+N}]^\top \in \mathbb{R}^{N\times C}$:

$$\mathbf{x'}_{T+1:T+N} = \mathcal{D}^{-1}(\mathcal{F}(\mathcal{D}(\mathbf{x}_{1:T}))) , \quad (3)$$

where $\mathcal{F}$ denotes our network.

After the DCT transformation, we apply one fully connected layer to operate only on the spatial dimension of the transformed motion sequence $\mathcal{D}(\mathbf{x}_{1:T}) \in \mathbb{R}^{T\times C}$:

$$\mathbf{z}^0 = \mathcal{D}(\mathbf{x}_{1:T})\mathbf{W}_0 + \mathbf{b}_0 , \quad (4)$$

where $\mathbf{z}^0 \in \mathbb{R}^{T\times C}$ is the output of the fully connected layer, $\mathbf{W}_0 \in \mathbb{R}^{C\times C}$ and $\mathbf{b}_0 \in \mathbb{R}^C$ represent the learnable parameters of the fully connected layer. In practice, this is equivalent to applying a transpose operation with a fully connected layer, and then transposing back the output feature, as shown in Figure 2.

Then, a series of m blocks are introduced to only operate on the temporal dimension, *i.e.*, only to merge information across frames. Each block consists of a fully connected layer followed by layer normalization, formally:

$$\mathbf{z}^i = \mathbf{z}^{i-1} + \text{LN}(\mathbf{W}_i\mathbf{z}^{i-1} + \mathbf{b}_i) , \quad (5)$$

where $\mathbf{z}^i \in \mathbb{R}^{T\times C}, i \in [1, .., m]$ denotes the output of the i-th MLP block, LN denotes the layer normalization operation, and $\mathbf{W}_i \in \mathbb{R}^{T\times T}$ and $\mathbf{b}_i \in \mathbb{R}^T$ are the learnable parameters of the fully connected layer in the i-th MLP block.

Finally, similar to the first fully connected layer, we add another fully connected layer after the MLP blocks to operate only on the spatial dimension of the feature, and then apply IDCT transformation to obtain the prediction:

$$\mathbf{x'}_{T+1:T+N} = \mathcal{D}^{-1}(\mathbf{z'}\mathbf{W}_{m+1} + \mathbf{b}_{m+1}) , \quad (6)$$

where $\mathbf{W}_{m+1}$ and $\mathbf{b}_{m+1}$ are the learnable parameters of the last fully connected layer.

Note that the lengths T and N do not need to be equal. When $T > N$, we only take the N first frames of the prediction, and in the case of $T < N$, we could pad our input sequence to N by repeating the last frame, as done in [41, 42].

3.3. Losses

As mention in the Section 1 and shown in the Figure 1, the last input pose is "close" to the future poses. Inspired

Table 2. **Action-wise results on Human3.6M** for different prediction time steps (ms). Lower is better. 256 samples are tested for each action. † indicates that the results are taken from the paper [41], ⋆ indicates that the results are taken from the paper [38].

Action	walking				eating				smoking				discussion			
Time (ms)	80	400	560	1000	80	400	560	1000	80	400	560	1000	80	400	560	1000
Res-RNN † [43]	23.2	66.1	71.6	79.1	16.8	61.7	74.9	98.0	18.9	65.4	78.1	102.1	25.7	91.3	109.5	131.8
convSeq2Seq † [31]	17.7	63.6	72.2	82.3	11.0	48.4	61.3	87.1	11.6	48.9	60.0	81.7	17.1	77.6	98.1	129.3
LTD-50-25 † [42]	12.3	44.4	50.7	60.3	7.8	38.6	51.5	75.8	8.2	39.5	50.5	72.1	11.9	68.1	88.9	118.5
LTD-10-10 † [42]	11.1	42.9	53.1	70.7	7.0	37.3	51.1	78.6	7.5	37.5	49.4	71.8	10.8	65.8	88.1	121.6
Hisrep † [41]	10.0	39.8	47.4	58.1	6.4	36.2	50.0	75.7	7.0	36.4	47.6	69.5	10.2	65.4	86.6	119.8
MSR-GCN ⋆ [13]	10.8	42.4	53.3	63.7	6.9	36.0	50.8	75.4	7.5	37.5	50.5	72.1	10.4	65.0	87.0	116.8
ST-DGCN-10-25 ⋆ [38]	11.2	42.8	49.6	58.9	6.5	36.8	50.0	74.9	7.3	37.5	48.8	69.9	10.2	64.4	86.1	116.9
siMLPe (Ours)	**9.9**	**39.6**	**46.8**	**55.7**	**5.9**	**36.1**	**49.6**	**74.5**	**6.5**	**36.3**	**47.2**	**69.3**	**9.4**	**64.3**	**85.7**	**116.3**
Action	directions				greeting				phoning				posing			
Time (ms)	80	400	560	1000	80	400	560	1000	80	400	560	1000	80	400	560	1000
Res-RNN † [43]	21.6	84.1	101.1	129.1	31.2	108.8	126.1	153.9	21.1	76.4	94.0	126.4	29.3	114.3	140.3	183.2
convSeq2Seq † [31]	13.5	69.7	86.6	115.8	22.0	96.0	116.9	147.3	13.5	59.9	77.1	114.0	16.9	92.9	122.5	187.4
LTD-50-25 † [42]	8.8	58.0	74.2	105.5	16.2	82.6	104.8	136.8	9.8	50.8	68.8	105.1	12.2	79.9	110.2	174.8
LTD-10-10 † [42]	8.0	54.9	76.1	108.8	14.8	79.7	104.3	140.2	9.3	49.7	68.7	105.1	10.9	75.9	109.9	171.7
Hisrep † [41]	7.4	56.5	73.9	106.5	13.7	78.1	101.9	138.8	8.6	49.2	67.4	105.0	10.2	75.8	107.6	178.2
MSR-GCN ⋆ [13]	7.7	56.2	75.8	105.9	15.1	85.4	106.3	**136.3**	9.1	49.8	67.9	104.7	10.3	75.9	112.5	176.5
ST-DGCN-10-25 ⋆ [38]	7.5	56.0	73.3	**105.9**	14.0	77.3	100.2	136.4	8.7	48.8	66.5	**102.7**	10.2	73.3	102.8	**167.0**
siMLPe (Ours)	**6.5**	**55.8**	**73.1**	106.7	**12.4**	**77.3**	**99.8**	137.5	**8.1**	**48.6**	**66.3**	103.3	**8.8**	**73.8**	**103.4**	168.7
Action	purchases				sitting				sittingdown				takingphoto			
Time (ms)	80	400	560	1000	80	400	560	1000	80	400	560	1000	80	400	560	1000
Res-RNN † [43]	28.7	100.7	122.1	154.0	23.8	91.2	113.7	152.6	31.7	112.0	138.8	187.4	21.9	87.6	110.6	153.9
convSeq2Seq † [31]	20.3	89.9	111.3	151.5	13.5	63.1	82.4	120.7	20.7	82.7	106.5	150.3	12.7	63.6	84.4	128.1
LTD-50-25 † [42]	15.2	78.1	99.2	134.9	10.4	58.3	79.2	118.7	17.1	76.4	100.2	143.8	9.6	54.3	75.3	118.8
LTD-10-10 † [42]	13.9	75.9	99.4	135.9	9.8	55.9	78.5	118.8	15.6	71.7	96.2	142.2	8.9	51.7	72.5	116.3
Hisrep † [41]	13.0	73.9	95.6	134.2	9.3	56.0	76.4	115.9	14.9	72.0	97.0	143.6	8.3	51.5	72.1	115.9
MSR-GCN ⋆ [13]	13.3	77.8	99.2	134.5	9.8	55.5	77.6	115.9	15.4	73.8	102.4	149.4	8.9	54.4	77.7	121.9
ST-DGCN-10-25 ⋆ [38]	13.2	74.0	95.7	**132.1**	9.1	54.6	75.1	114.8	14.7	**70.0**	**94.4**	**139.0**	8.2	**50.2**	**70.5**	112.9
siMLPe (Ours)	**11.7**	**72.4**	**93.8**	132.5	**8.6**	**55.2**	**75.4**	**114.1**	**13.6**	70.8	95.7	142.4	**7.8**	50.8	71.0	**112.8**
Action	waiting				walkingdog				walkingtogether				average			
Time (ms)	80	400	560	1000	80	400	560	1000	80	400	560	1000	80	400	560	1000
Res-RNN † [43]	23.8	87.7	105.4	135.4	36.4	110.6	128.7	164.5	20.4	67.3	80.2	98.2	25.0	88.3	106.3	136.6
convSeq2Seq † [31]	14.6	68.7	87.3	117.7	27.7	103.3	122.4	162.4	15.3	61.2	72.0	87.4	16.6	72.7	90.7	124.2
LTD-50-25 † [42]	10.4	59.2	77.2	108.3	22.8	88.7	107.8	156.4	10.3	46.3	56.0	65.7	12.2	61.5	79.6	112.4
LTD-10-10 † [42]	9.2	54.4	73.4	107.5	20.9	86.6	109.7	150.1	9.6	44.0	55.7	69.8	11.2	58.9	78.3	114.0
Hisrep † [41]	8.7	54.9	74.5	108.2	20.1	86.3	108.2	146.9	8.9	41.9	52.7	64.9	10.4	58.3	77.3	112.1
MSR-GCN ⋆ [13]	10.4	62.4	74.8	105.5	24.9	112.9	107.7	145.7	9.2	43.2	56.2	69.5	11.3	61.9	80.0	112.9
ST-DGCN-10-25 ⋆ [38]	8.7	53.6	71.6	**103.7**	20.4	84.6	105.7	145.9	8.9	43.8	54.4	64.6	10.6	57.9	76.3	109.7
siMLPe (Ours)	**7.8**	**53.2**	**71.6**	104.6	**18.2**	**83.6**	**105.6**	**141.2**	**8.4**	**41.2**	**50.8**	**61.5**	**9.6**	**57.3**	**75.7**	**109.4**

by this observation, instead of predicting the absolute 3D poses from scratch, we let our network predict the residual between the future pose x_{T+t} and the last input pose x_T. As we will show in Section 4.4, this eases the learning and improves performance.

Objective function. Our objective function $\mathcal{L}$ includes two terms $\mathcal{L}_{re}$ and $\mathcal{L}_v$:

$$\mathcal{L} = \mathcal{L}_{re} + \mathcal{L}_v \,. \tag{7}$$

$\mathcal{L}_{re}$ aims to minimize the $\mathcal{L}_2$-norm between the predicted motion $\mathbf{x'}_{T+1:T+N}$ and ground-truth one $\mathbf{x}_{T+1:T+N}$:

$$\mathcal{L}_{re} = \mathcal{L}_2(\mathbf{x'}_{T+1:T+N}, \mathbf{x}_{T+1:T+N}) \,. \tag{8}$$

$\mathcal{L}_v$ aims to minimize the $\mathcal{L}_2$-norm between the velocity of the predicted motion $\mathbf{v'}_{T+1:T+N}$ and the ground truth one $\mathbf{v}_{T+1:T+N}$:

$$\mathcal{L}_v = \mathcal{L}_2(\mathbf{v'}_{T+1:T+N}, \mathbf{v}_{T+1:T+N}) \,, \tag{9}$$

where $\mathbf{v}_{T+1:T+N} = [v_{T+1}^\top, .., v_{T+N}^\top]^\top \in \mathbb{R}^{N \times C}$, v_t represents the velocity at frame t and is computed as the time difference: $v_t = x_{t+1} - x_t$. We provide a full analysis of the loss terms in Section 4.4.

Table 3. **Results on AMASS and 3DPW** for different prediction time steps (ms). We report the MPJPE error in *mm*. Lower is better. The model is trained on the AMASS dataset. The results of the previous methods are taken from [41].

Dataset	AMASS-BMLrub								3DPW							
Time (ms)	80	160	320	400	560	720	880	1000	80	160	320	400	560	720	880	1000
convSeq2Seq [31]	20.6	36.9	59.7	67.6	79.0	87.0	91.5	93.5	18.8	32.9	52.0	58.8	69.4	77.0	83.6	87.8
LTD-10-10 [42]	**10.3**	**19.3**	36.6	44.6	61.5	75.9	86.2	91.2	**12.0**	**22.0**	38.9	46.2	59.1	69.1	76.5	81.1
LTD-10-25 [42]	11.0	20.7	37.8	45.3	57.2	65.7	71.3	75.2	12.6	23.2	39.7	46.6	57.9	65.8	71.5	75.5
Hisrep [41]	11.3	20.7	35.7	42.0	51.7	58.6	63.4	67.2	12.6	23.1	39.0	45.4	56.0	63.6	69.7	73.7
siMLPe (Ours)	10.8	19.6	**34.3**	**40.5**	**50.5**	**57.3**	**62.4**	**65.7**	12.1	22.1	**38.1**	**44.5**	**54.9**	**62.4**	**68.2**	**72.2**

Table 4. **Average results for different prediction time periods on Human3.6M and AMASS.** These results are obtained following the evaluation method of STS-GCN [49] and STG-GCN [61], instead of the standard evaluation protocol adopted in [38, 41, 42].

Dataset	Human3.6M								AMASS-BMLrub							
Time (ms)	80	160	320	400	560	720	880	1000	80	160	320	400	560	720	880	1000
STS-GCN [49]	10.1	17.1	33.1	38.3	50.8	60.1	68.9	75.6	10.0	12.5	21.8	24.5	31.9	38.1	42.7	45.5
STG-GCN [61]	10.1	16.9	32.5	38.5	50.0	-	-	72.9	10.0	11.9	20.1	24.0	30.4	-	-	43.1
siMLPe (Ours)	**4.5**	**9.8**	**22.0**	**28.1**	**39.3**	**49.2**	**57.8**	**63.7**	**6.1**	**10.8**	**19.1**	**22.8**	**29.5**	**35.1**	**39.7**	**42.7**

4. Experiments

In this section, we present our experimental details and results. We introduce the datasets and evaluation metric in Section 4.1, the implementation details in Section 4.2, and the quantitative/qualitative results in Section 4.3. An exhaustive ablation analysis is provided in Section 4.4.

4.1. Datasets and evaluation metric

Human3.6M dataset [23]. Human3.6M contains 7 actors performing 15 actions, and 32 joints are labeled for each pose. We follow the same testing protocols of [41] and use *S5* as the test set, *S11* as the validation set, and the others as the train set. Previous works use different test sampling strategies, including 8 samples per action [42,43], 256 samples per action [41] or all samples in the test set [13]. As 8 samples are too little and taking all testing samples could not balance different actions with different sequence lengths, we thus take 256 samples per action for testing, and evaluate on 22 joints as in [38, 41–43].

AMASS dataset [39]. AMASS is a collection of multiple Mocap datasets [1, 3, 5, 11, 14, 16, 21, 28, 35, 37, 39, 40, 44, 48, 53–56] unified by SMPL parameterization [36]. We follow [41] to use AMASS-BMLrub [53] as the test set and split the rest of the AMASS dataset into training and validation sets. The model is evaluated on 18 joints as in [41].

3DPW dataset [58]. 3DPW is a dataset including indoor and outdoor scenes. A pose is represented by 26 joints, but we follow [41] and evaluate 18 joints using the model trained on AMASS to evaluate generalization.

Evaluation metric. We report the Mean Per Joint Position Error (MPJPE) on 3D joint coordinates, which is the most widely used metric for evaluating 3D pose errors. This metric calculates the average L2-norm across different joints between the prediction and ground-truth. Similar to previous works [13,38,41,42], we ignore the global rotation and translation of the poses and keep the sampling rate as 25 frames per second (FPS) for all datasets.

4.2. Implementation details

In practice, we set the input length $T = 50$, the output length $N = 10$ on Human3.6M dataset and $N = 25$ on AMASS dataset and 3DPW dataset. During testing, we apply our model in an auto-regressive manner to generate motion for longer periods. The feature dimension $C = 3 \times K$, where K is the number of joints, $K = 22$ for Human3.6M and $K = 18$ for AMASS and 3DPW.

To train our network, we set the batch size to 256 and use the Adam optimizer [25]. The memory consumed by our network is about $1.5GB$ during the training. All our experiments are conducted using the Pytorch [47] framework on a single NVIDIA RTX 2080Ti graphics card. We train our network on the Human3.6M dataset for 35k iterations, the learning rate starts from 0.0003 at the beginning and drops to 0.00001 after 30k steps. The training takes $\sim$30 minutes. For AMASS dataset, we train our network for 115k iterations. The learning rate starts from 0.0003 at the beginning and drops to 0.00001 after 100k steps. The training takes $\sim$2 hours. During training, we only use the front-back flip as data augmentation, which randomly inverts the motion sequence during the training.

4.3. Quantitative and qualitative results

In this section, we compare our approach to existing state-of-the-art methods on different datasets. We report

Table 5. **Ablation of the number of MLP blocks** on Human3.6M.

Nb. Blocks	# Param.(M) ↓	MPJPE (mm) ↓							
		80	160	320	400	560	720	880	1000
1	0.012	12.7	28.5	59.7	72.1	93.6	107.0	116.8	123.6
2	0.014	10.9	24.9	52.3	64.0	83.2	97.3	108.4	115.4
6	0.025	10.2	23.1	48.8	60.1	79.0	93.3	105.1	112.6
12	0.041	9.9	22.4	47.2	58.3	77.1	91.5	103.3	110.9
24	0.073	9.7	22.0	46.8	57.7	76.4	90.8	102.6	110.3
48 (Ours)	0.138	**9.6**	**21.7**	**46.3**	**57.3**	**75.7**	**90.1**	**101.8**	**109.4**
64	0.180	9.6	21.8	46.5	57.5	76.0	90.1	101.9	109.7
96	0.266	9.7	21.9	46.7	57.8	76.3	90.5	102.1	109.8

Table 6. **Ablation of different components of our network** on Human3.6M. *'LN'* denotes the layer normalization. *'DCT'* denotes the DCT transformation. *'Spa. only'* means that all FC layers are on spatial dimensions (w/o transpose operations before/after MLP blocs). *'Temp. only'* means that all FC layers are on temporal dimensions (w/o any transpose operations).

Ablation	80	160	320	400	560	720	880	1000
Spa. only, w/o LN	23.7	44.0	75.5	87.6	106.3	120.4	130.5	135.6
Spa. only	23.8	43.0	73.4	85.2	102.0	116.3	125.3	131.9
Temp. only	9.9	22.4	47.2	58.4	77.2	91.1	102.8	110.5
w/o LN	12.7	29.0	62.3	76.2	97.4	111.6	121.6	127.3
w/o DCT	9.9	22.4	47.3	58.4	76.9	91.2	102.8	110.5
siMLPe (ours)	**9.6**	**21.7**	**46.3**	**57.3**	**75.7**	**90.1**	**101.8**	**109.4**

Table 7. **Ablation of data augmentation** on Human3.6M. We only use front-back flip as our data augmentation, *i.e.*, we randomly invert the motion sequence during the training.

	80	160	320	400	560	720	880	1000
w/o aug	10.0	22.6	48.3	59.7	78.2	92.0	103.4	110.8
w aug	**9.6**	**21.7**	**46.3**	**57.3**	**75.7**	**90.1**	**101.8**	**109.4**

MPJPE in *mm* at different prediction time steps up to 1000ms.

Human3.6M dataset. In Table 1, we compare our method with other state-of-the-art methods on the Human3.6M dataset. Our method outperforms all previous methods on every frame with much fewer parameters.
As explained in Section 4.1, some different methods have taken different test sampling strategies. Following [41], we choose to test with 256 samples on 22 joints. To make a fair comparison, we evaluate all the methods using the same testing protocol. Our method outperforms all previous methods on every frame with a much less number of parameters. Besides, previous works usually report short-term ($0 \sim 500ms$) and long-term ($500 \sim 1000ms$) predictions separately, and [38] reports short-/long- term results using two different models. In our tables, all the results from $0 \sim 1000ms$ are predicted by a single model, and for [38], we report the results of their model which achieves the best performance on long-term prediction. In addition, we also evaluate the two simple approaches mentioned in Section 1 on the Human3.6M dataset in Table 1: 'Repeating Last-frame' takes the last input pose and repeats it N times to serve as output, and 'One FC' uses only one single fully connected layer trained on Human3.6m. These results show that the task of human motion prediction could be potentially modeled in a completely different and simple way without explicitly fusing spatial and temporal information. Furthermore, similar to all the previous works, we also detail the action-wise results in Table 2.

AMASS and 3DPW datasets. In Table 3, we report the performance of the model trained on AMASS and tested on the AMASS-BMLrub and 3DPW datasets, following the evaluation protocol of [41]. Different from the Human3.6M dataset where the training and testing data are from the same types of actions performed by different actors, the difference between training and testing data under this protocol is much larger, which makes the task more challenging in terms of generalization. As shown in the table, our approach performs consistently better on long-term prediction. Moreover, our model is much lighter. For example, the parameter size of our model is $\sim 4\%$ of Hisrep [41].

While the commonly used evaluation protocol is to consider the predicted error at different time steps, some works [49, 61] report their result by taking the average error from the first time step to a certain time step. We report the predicted error at different time steps in all the tables, except in Table 4, where we report the average error for comparison with [49, 61]. Our approach also achieves better performance than these two methods.

Qualitative results. In addition to the quantitative results, we provide some qualitative results of our method in Figure 3, showing some testing examples on the Human3.6M dataset. We could find that the predictions of our method perfectly match the ground-truth on short-term prediction, and globally fits the ground-truth on long-term prediction. The error becomes larger when looking into longer predictions, which is a common problem for all the motion prediction methods as shown in Table 1 and Table 3. This is because most of the current methods use auto-regression for predicting a longer future, which will make the error accumulate. Moreover, uncertainty grows very quickly with time when predicting human motions.

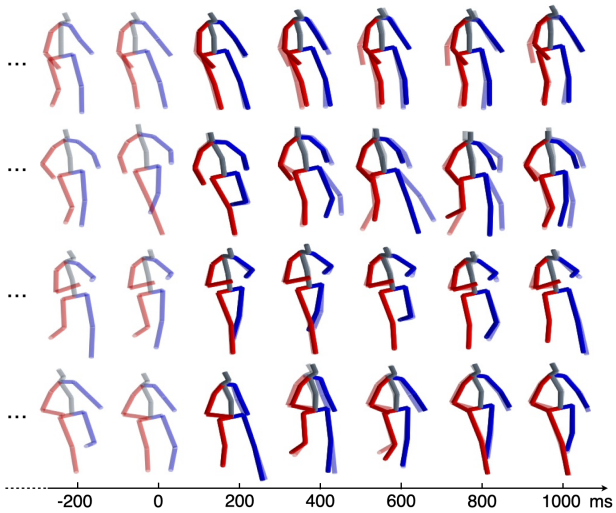

Figure 3. **Qualitative results of our method SIMLPE.** The skeletons in light colors are the input (before 0ms) and the ground-truth (after 0ms). Those with dark colors represent the predicted motions. Our prediction results are close to the ground-truth.

4.4. Ablation study

We evaluate below the influence of the different components of our approach on the Human3.6M dataset.

Number of MLP blocks. We ablate the of the number of MLP blocks m in Table 5. Our proposed architecture already achieves good performance using only 2 MLP blocks with $0.014M$ parameters. The network achieves its best performance with 48 MLP blocks.

Network architecture. In Table 6, we ablate the different components of our network. As the table shows, the temporal feature fusion and layer normalization are both of vital importance to our network. If the network just operates along the spatial dimension of the motion sequence without merging any information across different frames, it will lead to degraded results. However, if the network just operates along the temporal dimension, the network will still achieve comparable performance. Besides, the use of DCT transformation can further improve the performance slightly.

Data augmentation. In Table 7, we ablate the use of front-back flip data augmentation and find that the data augmentation slightly improves the performance.

Loss. In Table 8, we evaluate the importance of different loss terms used during training. As shown in the table, with the help of the velocity loss $\mathcal{L}_v$, the network achieves better performance on long-term predictions while maintaining the same performance on the short-term.

Learning residual displacement. In Table 9, we analyze the importance of the proposed residual displacement and compare it to other types of residual used in previous works [42, 43]. Our method aims to predict the differences between each future pose and the last observed pose, after the IDCT transformation. When predicting directly the absolute 3D pose ('w/o residual'), the performance drops dramatically. We also test other types of residual by either learning the residual in the DCT space, before applying the IDCT transformation ('Before IDCT') following [42], or learning the velocity of the motion ('consecutive') following [43], and both achieve inferior performance compared to our proposed residual displacement.

Table 8. **Ablation of different loss terms** on Human3.6M.

$\mathcal{L}_{re}$	$\mathcal{L}_v$	80	160	320	400	560	720	880	1000
✓		9.6	21.8	46.5	57.5	76.7	91.5	103.5	111.3
✓	✓	**9.6**	**21.7**	**46.3**	**57.3**	**75.7**	**90.1**	**101.8**	**109.4**

Table 9. **Analysis of different types of residual displacement** on Human3.6M. SIMLPE predicts the differences of each future frame with the last observation (after IDCT). *'Before IDCT'* learns the residual before applying the IDCT transformation. *'Consecutive'* learns the velocity between consecutive frames. *'w/o residual'* predicts directly the absolute 3D poses.

Residual	80	160	320	400	560	720	880	1000
w/o residual	12.4	25.1	50.7	61.6	80.1	93.9	105.5	113.0
Consecutive	9.7	22.0	46.8	57.8	76.5	90.7	102.4	110.1
Before IDCT	10.4	23.0	48.2	59.1	77.9	91.8	103.2	110.5
SIMLPE (ours)	**9.6**	**21.7**	**46.3**	**57.3**	**75.7**	**90.1**	**101.8**	**109.4**

5. Conclusion

In this paper, we present SIMLPE, a simple-yet-effective network for human motion prediction. SIMLPE is composed of only fully connected layers, layer normalization, and transpose operations. The only non-linear operation is thus the layer normalization. While using much fewer parameters, SIMLPE achieves state-of-the-art performance on various benchmarks. The reported ablation study also demonstrates the interest of various design choices, highlighting the importance of temporal information fusion in this task. We hope the simplicity of SIMLPE will help the community to rethink the task of human motion prediction.

6. Acknowledgement

This research was supported by ANR-3IA MIAI (ANR-19-P3IA-0003), ANR-JCJC ML3RI (ANR-19-CE33-0008-01), H2020 SPRING (funded by EC under GA #871245), by the Spanish government with the project MoHuCo PID2020-120049RB-I00 and by an Amazon Research Award. This project has received funding from the CHIST-ERA IPALM project.

References

[1] Ijaz Akhter and Michael J. Black. Pose-Conditioned Joint Angle Limits for 3D Human Pose Reconstruction. In *Conference on Computer Vision and Pattern Recognition*, 2015. 6

[2] Emre Aksan, Manuel Kaufmann, Peng Cao, and Otmar Hilliges. A Spatio-Temporal Transformer for 3D Human Motion Prediction. In *3DV*, 2021. 2, 3

[3] Andreas Aristidou, Ariel Shamir, and Yiorgos Chrysanthou. Digital Dance Ethnography: Organizing Large Dance Collections. *J. Comput. Cult. Herit.*, 12(4), Nov. 2019. 6

[4] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer Normalization. In *arXiv Preprint*, 2016. 2, 3, 4

[5] Federica Bogo, Javier Romero, Gerard Pons-Moll, and Michael J. Black. Dynamic FAUST: Registering Human Bodies in Motion. In *Conference on Computer Vision and Pattern Recognition*, 2017. 6

[6] Arij Bouazizi, Adrian Holzbock, Ulrich Kressel, Klaus Dietmayer, and Vasileios Belagiannis. Motionmixer: Mlp-based 3d human body pose forecasting. *arXiv preprint arXiv:2207.00499*, 2022. 3

[7] Matthew Brand and Aaron Hertzmann. Style Machines. In *Computer Graphics and Interactive Techniques*, 2000. 1

[8] Judith Butepage, Michael J. Black, Danica Kragic, and Hedvig Kjellstrom. Deep Representation Learning for Human Motion Prediction and Classification. In *Conference on Computer Vision and Pattern Recognition*, 2017. 2

[9] Judith Bütepage, Hedvig Kjellström, and Danica Kragic. Anticipating Many Futures: Online Human Motion Prediction and Generation for Human-Robot Interaction. In *International Conference on Robotics and Automation*, 2018. 2

[10] Yujun Cai, Lin Huang, Yiwei Wang, Tat-Jen Cham, Jianfei Cai, Junsong Yuan, Jun Liu, Xu Yang, Yiheng Zhu, Xiaohui Shen, and Others. Learning Progressive Joint Propagation for Human Motion Prediction. In *European Conference on Computer Vision*, 2020. 2, 3

[11] Anargyros Chatzitofis, Leonidas Saroglou, Prodromos Boutis, Petros Drakoulis, Nikolaos Zioulis, Shishir Subramanyam, Bart Kevelham, Caecilia Charbonnier, Pablo Cesar, Dimitrios Zarpalas, and Others. HUMAN4D: A Human-Centric Multimodal Dataset for Motions and Immersive Media. In *American Control Conference*, 2020. 6

[12] Hsu-kuang Chiu, Ehsan Adeli, Borui Wang, De-An Huang, and Juan Carlos Niebles. Action-Agnostic Human Pose Forecasting. In *IEEE Winter Conference on Applications of Computer Vision*, 2019. 1, 2

[13] Lingwei Dang, Yongwei Nie, Chengjiang Long, Qing Zhang, and Guiqing Li. MSR-GCN: Multi-Scale Residual Graph Convolution Networks for Human Motion Prediction. In *International Conference on Computer Vision*, 2021. 2, 3, 4, 5, 6

[14] Advanced Computing Center for the Arts and Design. ACCAD MoCap Dataset. 6

[15] Katerina Fragkiadaki, Sergey Levine, Panna Felsen, and Jitendra Malik. Recurrent Network Models for Human Dynamics. In *International Conference on Computer Vision*, 2015. 1, 2

[16] Saeed Ghorbani, Kimia Mahdaviani, Anne Thaler, Konrad Kording, Douglas James Cook, Gunnar Blohm, and Nikolaus F. Troje. MoVi: A Large Multipurpose Motion and Video Dataset, 2020. 6

[17] Haifeng Gong, Jack Sim, Maxim Likhachev, and Jianbo Shi. Multi-Hypothesis Motion Planning for Visual Object Tracking. In *International Conference on Computer Vision*, 2011. 1

[18] Liang-Yan Gui, Yu-Xiong Wang, Xiaodan Liang, and José MF Moura. Adversarial Geometry-Aware Human Motion Prediction. In *European Conference on Computer Vision*, 2018. 2

[19] Wen Guo, Xiaoyu Bie, Xavier Alameda-Pineda, and Francesc Moreno-Noguer. Multi-Person Extreme Motion Prediction. In *Conference on Computer Vision and Pattern Recognition*, 2022. 2

[20] Alejandro Hernandez, Jurgen Gall, and Francesc Moreno-Noguer. Human Motion Prediction via Spatio-Temporal Inpainting. In *International Conference on Computer Vision*, 2019. 2

[21] Ludovic Hoyet, Kenneth Ryall, Rachel Mcdonnell, and Carol O'sullivan. Sleight of Hand: Perception of Finger Motion from Reduced Marker Sets. In *ACM SIGGRAPH*, 2012. 6

[22] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018. 3

[23] 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*, 2013. 1, 2, 6

[24] Ashesh Jain, Amir R. Zamir, Silvio Savarese, and Ashutosh Saxena. Structural-Rnn: Deep Learning on Spatio-Temporal Graphs. In *Conference on Computer Vision and Pattern Recognition*, 2016. 1, 2

[25] Diederik P. Kingma and Jimmy Ba. Adam: A Method for Stochastic Optimization. In *arXiv Preprint*, 2014. 6

[26] Thomas N. Kipf and Max Welling. Semi-Supervised Classification with Graph Convolutional Networks. In *International Conference for Learning Representations*, 2017. 3

[27] Hema Swetha Koppula and Ashutosh Saxena. Anticipating Human Activities for Reactive Robotic Response. In *International Conference on Intelligent Robots and Systems*, 2013. 1

[28] Bio Motion Lab. BMLhandball Motion Capture Database. 6

[29] Tim Lebailly, Sena Kiciroglu, Mathieu Salzmann, Pascal Fua, and Wei Wang. Motion Prediction Using Temporal Inception Module. In *Asian Conference on Computer Vision*, 2020. 3

[30] Andreas M. Lehrmann, Peter V. Gehler, and Sebastian Nowozin. Efficient Nonlinear Markov Models for Human Motion. In *Conference on Computer Vision and Pattern Recognition*, 2014. 2

[31] Chen Li, Zhen Zhang, Wee Sun Lee, and Gim Hee Lee. Convolutional Sequence to Sequence Model for Human Dynamics. In *Conference on Computer Vision and Pattern Recognition*, 2018. 2, 4, 5, 6

[32] Maosen Li, Siheng Chen, Xu Chen, Ya Zhang, Yanfeng Wang, and Qi Tian. Symbiotic Graph Neural Networks for 3D Skeleton-Based Human Action Recognition and Motion Prediction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2021. 2, 3

[33] Maosen Li, Siheng Chen, Yangheng Zhao, Ya Zhang, Yanfeng Wang, and Qi Tian. Dynamic Multiscale Graph Neural Networks for 3D Skeleton Based Human Motion Prediction. In *Conference on Computer Vision and Pattern Recognition*, 2020. 2, 3

[34] Zhenguang Liu, Shuang Wu, Shuyuan Jin, Qi Liu, Shijian Lu, Roger Zimmermann, and Li Cheng. Towards Natural and Accurate Future Motion Prediction of Humans and Animals. In *Conference on Computer Vision and Pattern Recognition*, 2019. 1, 2

[35] Matthew Loper, Naureen Mahmood, and Michael J. Black. MoSh: Motion And Shape Capture from Sparse Markers. *IEEE Transactions on Robotics and Automation*, 33(6), Nov. 2014. 6

[36] Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. SMPL: A Skinned Multi-Person Linear Model. *IEEE Transactions on Robotics and Automation*, 34(6), 2015. 6

[37] Eyes JAPAN Co Ltd. Eyes Japan MoCap Dataset. 6

[38] Tiezheng Ma, Yongwei Nie, Chengjiang Long, Qing Zhang, and Guiqing Li. Progressively Generating Better Initial Guesses Towards Next Stages for High-Quality Human Motion Prediction. In *Conference on Computer Vision and Pattern Recognition*, 2022. 2, 3, 4, 5, 6, 7

[39] Naureen Mahmood, Nima Ghorbani, Nikolaus F. Troje, Gerard Pons-Moll, and Michael J. Black. AMASS: Archive of Motion Capture as Surface Shapes. In *International Conference on Computer Vision*, 2019. 2, 6

[40] Christian Mandery, Ömer Terlemez, Martin Do, Nikolaus Vahrenkamp, and Tamim Asfour. The KIT Whole-Body Human Motion Database. In *ICAR*, 2015. 6

[41] Wei Mao, Miaomiao Liu, and Mathieu Salzmann. History Repeats Itself: Human Motion Prediction via Motion Attention. In *European Conference on Computer Vision*, 2020. 2, 3, 4, 5, 6, 7

[42] Wei Mao, Miaomiao Liu, Mathieu Salzmann, and Hongdong Li. Learning Trajectory Dependencies for Human Motion Prediction. In *International Conference on Computer Vision*, 2019. 1, 2, 3, 4, 5, 6, 8

[43] Julieta Martinez, Michael J. Black, and Javier Romero. On Human Motion Prediction Using Recurrent Neural Networks. In *Conference on Computer Vision and Pattern Recognition*, 2017. 1, 2, 3, 4, 5, 6, 8

[44] M. Müller, T. Röder, M. Clausen, B. Eberhardt, B. Krüger, and A. Weber. Documentation Mocap Database HDM05. Technical Report CG-2007-2, Universität Bonn, June 2007. 6

[45] Vinod Nair and Geoffrey E. Hinton. Rectified Linear Units Improve Restricted Boltzmann Machines. In *International Conference on Machine Learning*, 2010. 2

[46] Brian Paden, Michal Čáp, Sze Zheng Yong, Dmitry Yershov, and Emilio Frazzoli. A Survey of Motion Planning and Control Techniques for Self-Driving Urban Vehicles. *IEEE Transactions on Robotics and Automation*, 2016. 1

[47] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, and Others. Pytorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems*, 2019. 6

[48] Leonid Sigal, Alexandru O. Balan, and Michael J. Black. HumanEva: Synchronized Video and Motion Capture Dataset and Baseline Algorithm for Evaluation of Articulated Human Motion. *International Journal of Computer Vision*, 2010. 6

[49] Theodoros Sofianos, Alessio Sampieri, Luca Franco, and Fabio Galasso. Space-Time-Separable Graph Convolutional Network for Pose Forecasting. In *International Conference on Computer Vision*, 2021. 6, 7

[50] Alessandro Sperduti and Antonina Starita. Supervised Neural Networks for the Classification of Structures. *IEEE Transactions on Robotics and Automation*, 1997. 3

[51] Graham W. Taylor, Geoffrey E. Hinton, and Sam T. Roweis. Modeling Human Motion Using Binary Latent Variables. In *Advances in Neural Information Processing Systems*, 2007. 2

[52] Ilya O Tolstikhin, Neil Houlsby, Alexander Kolesnikov, Lucas Beyer, Xiaohua Zhai, Thomas Unterthiner, Jessica Yung, Andreas Steiner, Daniel Keysers, Jakob Uszkoreit, et al. Mlp-mixer: An all-mlp architecture for vision. *Advances in Neural Information Processing Systems*, 34:24261–24272, 2021. 3

[53] Nikolaus F. Troje. Decomposing Biological Motion: A Framework for Analysis and Synthesis of Human Gait Patterns. *Journal of Vision*, 2(5), Sept. 2002. 6

[54] Matt Trumble, Andrew Gilbert, Charles Malleson, Adrian Hilton, and John Collomosse. Total Capture: 3D Human Pose Estimation Fusing Video and Inertial Sensors. In *British Machine Vision Conference*, 2017. 6

[55] Carnegie Mellon University. CMU MoCap Dataset. 6

[56] Simon Fraser University and National University of Singapore. SFU Motion Capture Database. 6

[57] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. In *Advances in Neural Information Processing Systems*, 2017. 3

[58] 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 *European Conference on Computer Vision*, 2018. 2, 6

[59] Jack M. Wang, David J. Fleet, and Aaron Hertzmann. Gaussian Process Dynamical Models. In *Advances in Neural Information Processing Systems*, 2005. 2

[60] Jack M. Wang, David J. Fleet, and Aaron Hertzmann. Gaussian Process Dynamical Models for Human Motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2007. 1

[61] Chongyang Zhong, Lei Hu, Zihao Zhang, Yongjing Ye, and Shihong Xia. Spatio-Temporal Gating-Adjacency GCN For Human Motion Prediction. In *Conference on Computer Vision and Pattern Recognition*, 2022. 6, 7