

---

# A Spatio-temporal Transformer for 3D Human Motion Prediction

---

**Emre Aksan\***

Department of Computer Science  
ETH Zurich  
eaksan@inf.ethz.ch

**Peng Cao\*<sup>†</sup>**

Peking University  
caopeng2016@pku.edu.cn

**Manuel Kaufmann**

Department of Computer Science  
ETH Zurich  
manuel.kaufmann@inf.ethz.ch

**Otmar Hilliges**

Department of Computer Science  
ETH Zurich  
otmar.hilliges@inf.ethz.ch

## Abstract

In this paper, we propose a novel Transformer-based architecture for the task of generative modelling of 3D human motion. Previous works commonly rely on RNN-based models considering shorter forecast horizons reaching a stationary and often implausible state quickly. Instead, our focus lies on the generation of plausible future developments over longer time horizons. To mitigate the issue of convergence to a static pose, we propose a novel architecture that leverages the recently proposed self-attention concept. The task of 3D motion prediction is inherently spatio-temporal and thus the proposed model learns high dimensional embeddings for skeletal joints followed by a decoupled *temporal* and *spatial* self-attention mechanism. This allows the model to access past information directly and to capture spatio-temporal dependencies explicitly. We show empirically that this reduces error accumulation over time and allows for the generation of perceptually plausible motion sequences over long time horizons up to 20 seconds as well as accurate short-term predictions.

## 1 Introduction

Imagine a person running or riding a bicycle – while your inner motion depiction may not be an accurate prediction, we are able to effortlessly forecast the complex dynamics of human motion in a plausible fashion over arbitrarily long time horizons. Despite much progress on short-term 3D human motion modeling, the long-term tasks remains challenging for machines. Most previous works define short-term prediction as 0.5 seconds and long-term prediction as 1 second, respectively. In this paper we ask the question if machines can learn to predict plausible human motion over much longer time frames such as of 5-20 seconds. Having access to such a model would have significant implications for many application domains, including self-driving cars, robotics, and human computer interaction.

From a learning perspective the problem of 3D human motion modeling can be seen as a generative modeling task: A network learns to synthesize a sequences of human poses where the model is conditioned on the seed sequence. Hence, we hypothesize that a model that is truly capable of learning motion dynamics should be able to generate natural motion sequences over arbitrarily long horizons and should not converge to a mean pose, especially for periodic motions like walking or

---

\*The first two authors contributed equally to this work.

<sup>†</sup>Peng performed this work during an internship at ETH.

running. However, most existing work typically suffers from a collapse to a static pose, suggesting a lack of ability to accurately capture long-term motion dependencies.

Given the temporal nature of human motion, it is not surprising that recurrent neural networks (RNNs) are the most popular choice [1–6]. RNNs model short and long-term dependencies by propagating information through their hidden state. Convolutional neural networks (CNN) in a sequence-to-sequence framework have also been proposed for this task [7], sometimes combined with spatial priors [8]. Such models follow an auto-regressive approach, factorizing the predictions into step-wise conditionals based on previous predictions. This approach causes the model to accumulate error over

time and eventually the predictions collapse to a non-plausible pose. This issue can be associated with the exposure bias problem [9] due to discrepancies between data and model distributions. Previous work has applied various strategies to work around this problem, such as using model predictions during training [4, 5], applying noise to the inputs [1, 3, 7, 10], or using adversarial losses [6, 7].

In this work, we present a novel architecture for 3D human motion modeling, which considers the spatio-temporal aspects of the data explicitly without relying on the propagation of a hidden state. Our approach is motivated by the recent success of the Transformer model [11] in tasks such as NLP [11, 12], music [13], or images [14, 15]. While the vanilla Transformer is designed for 1D sequences with a self-attention mechanism [11, 16, 17], we note that the task of 3D motion prediction is inherently spatio-temporal and we propose to decouple the temporal and spatial dimensions.

The proposed spatio-temporal attention mechanism is trained to identify useful information from a known sequence to construct the next output pose. For every joint we define *temporal attention* over the same joint in the past and *spatial attention* over the other joints at the same time step (see Fig. 1). The spatial attention block draws information from the joint features at the current time step whereas the temporal block focuses on distilling information from the previous records of individual joints. A prediction is then made by summarizing previous time steps as a weighted combination. This dual self-attention over the sequence allows the model to access past information directly and hence capture the dependencies explicitly [17, 11], mitigating the error accumulation over time. It furthermore enables interpretability since the attention weights indicate informative sequence components that led to the prediction.

In the remainder of this paper we discuss related work (Sec. 2), introduce our spatio-temporal Transformer (ST-Transformer) in Sec. 3 and finally evaluate our generative modeling effort in Sec. 4. Our experiments show that a naive application of 1D self-attention still suffers from collapsing poses, whereas the ST-Transformer is able to produce convincing long-term predictions (up to 20 seconds for periodic motions) and beating the SoA in short-term horizons.

## 2 Related Work

**Recurrent Models** RNNs are the dominant architecture for 3D motion modeling tasks [1–3, 10, 18–20]. Fragkiadaki et al. [2] propose the Encoder-Recurrent-Decoder (ERD) model where an LSTM cell operates in latent space. Jain et al. [3] build a skeleton-like st-graph with RNNs as nodes. Aksan et al. [1] replace the dense output layer of a RNN architecture with a structured prediction layer that follows the kinematic chain. The authors furthermore introduce a large-scale human motion dataset, AMASS [21], to the task of motion prediction. The error accumulation problem is typically combated by exposing the model to dropout or Gaussian noise during training. Ghosh et al. [10] more explicitly train a separate de-noising autoencoder that refines the noisy RNN predictions.

Martinez et al. [4] introduce a sequence-to-sequence (seq2seq) architecture and skip connection from in- to output on the decoder to address the transition problems between the seed and predictions. They also propose training the model with the predictions to alleviate the exposure bias problem. Similarly, Pavllo et al. [5] suggest using a teacher-forcing ratio to gradually expose the model to its

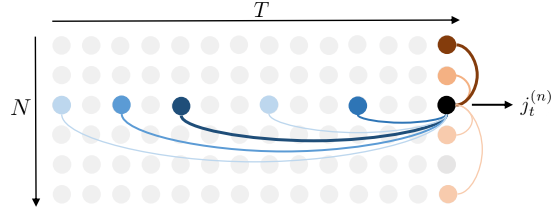


Figure 1: **Spatio-temporal attention.** The motion sequence is a matrix containing  $N$  joint configurations  $\hat{j}_t^{(n)}$  over  $T$  time steps. For a given joint (black), the *temporal* (blue) and *spatial* attention (red) are illustrated. Color intensity indicates attention weight.

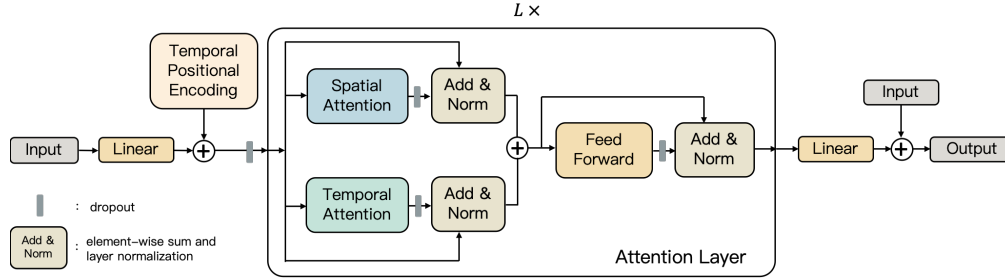


Figure 2: **Architecture overview.** We start by projecting every joint into an embedding space. After injecting positional encodings and applying dropout, the embeddings are then fed to  $L$  consecutive attention layers. We employ a novel spatio-temporal multi-head attention mechanism based on [11]. It is split into a *temporal* attention block that updates a joint’s embedding by looking at the past instances of the same joint and a *spatial* block that attends over all the joints in the current time step. Finally, we estimate the next pose by projecting the embeddings back to the joint space and using a residual connection from input to output, following [4].

own predictions. In [18], the seq2seq framework is modified to explicitly model different time scales using a hierarchy of RNNs. Wang et al. [6] propose a geodesic loss and adversarial training and Wang et al. [20] replace the likelihood objective with a policy gradient method via imitation learning. The acRNN of Zhou et al. [22] uses an augmented conditioning scheme which allows for long-term motion synthesis. However their model is trained on specific motion types, whereas in this work we strive to train a single model encompassing multiple motion types.

While the use of domain-specific priors and objectives can improve short term accuracy, the inherent problems of the underlying RNN architectures still exist. Maintaining long-term dependencies is an issue due to the need of summarizing the entire history in a hidden state of fixed size. Inspired by similar observations in the field of NLP [11, 17], we introduce a spatio-temporal self-attention mechanism to mitigate this problem. In doing so we let the network explicitly reason about past frames without the need to compress the past into a single hidden vector.

**Non-recurrent Models** [23, 24] use dense layers on sliding windows of the motion sequences. In [25, 26] convolutional models are introduced for motion synthesis conditioned on trajectories. More recently, Hernandez et al. [27] propose to treat motion prediction as an image inpainting task and use a convolutional model with adversarial losses. Joints are represented as 3D positions, often requiring auxiliary losses such as bone length and joint limits to ensure anatomical consistency. Li et al. [7] use CNNs instead of RNNs in the sequence-to-sequence framework to improve long-term dependencies.

Mao et al. [28] represent sequences of joints via discrete cosine transform (DCT) coefficients and train a graph convolutional network (GCN) to learn inter-joint dependencies. Like the DCT-GCN, our ST-Transformer also learns to identify informative joints in the skeleton. The GCN operates on temporal windows of poses and produces the entire output in one go. In contrast, our model is fully autoregressive and hence can easily be deployed to produce arbitrarily long sequences.

Transformers have recently been shown to be very successful for a wide variety of sequence modeling tasks, such as music [13], images [14, 15], or videos [29]. We follow in these footsteps by presenting a transformer tailored to 3D human motion modeling. To the best of our knowledge, we are the first to do so.

In summary, most existing work has introduced regularization techniques, auxiliary loss terms, or adversarial objectives to address the inherent problems of the underlying architectures, making these models often hard to train. We show that the self-attention concept is very effective in learning motion dynamics and allows for the design of a versatile mechanism that is effective and easy to train.

### 3 Method

We now explain the architecture of the proposed spatio-temporal transformer (ST-Transformer) in detail. For an overview please refer to Fig. 2. Our approach first decomposes the skeleton into joints and independently projects them onto a higher dimensional representation space. This step can be considered as learning embeddings for joint angles. We later refine the joint representation by using the relevant inter- and intra-joint dependencies which are identified by the attention blocks. Finally, we assemble the skeleton from the predicted joints which constitutes the next pose. Our method

uses the building blocks of the Transformer [11], but with two main differences: (1) a decoupled spatio-temporal attention mechanism and (2) a fully auto-regressive model.

### 3.1 Problem Formulation

A motion sample can be represented by a sequence  $\mathbf{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_T\}$  where a frame  $\mathbf{x}_t = \{\mathbf{j}_t^{(1)}, \dots, \mathbf{j}_t^{(N)}\}$  denotes a pose at time step  $t$  with joints  $\mathbf{j}_t^{(n)} \in \mathbb{R}^M$ . Each joint  $\mathbf{j}$  is an  $M$ -dimensional vector where  $M$  is determined by the pose parameterization, e.g., 3D position, rotation matrix, angle-axis, or quaternion. We use a rotation matrix representation, i.e.,  $M = 9$ . Following a predefined order, a sequence sample  $\mathbf{X}$  can be written as a matrix of size  $NM \times T$ , where blocks of  $M$  rows represent the  $i$ -th joint configuration at time step  $t$ , i.e.  $\mathbf{X}_{Mi:M(i+1), t} = \mathbf{j}_t^{(i)}$ .

In our notation the subscript denotes the time step. We use  $n$ -tuples in the superscript ordered by joint index, layer index, and optionally attention head index. For example,  $\mathbf{W}^{(n,I)}$  stands for the weight matrix of the input ( $I$ ) of joint  $n$ . Please note that projection matrices  $\mathbf{W}$  and bias terms  $\mathbf{b}$  are trainable.  $n$  in the superscript indicates that it is only used by joint  $n$ .

### 3.2 Spatio-temporal Transformer

**Joint Embeddings** We first project all joints into a  $D$ -dimensional space via a single linear layer:

$$\mathbf{e}_t^{(n)} = \mathbf{W}^{(n,E)} \mathbf{j}_t^{(n)} + \mathbf{b}^{(n,E)} \quad (1)$$

where  $\mathbf{W}^{(n,E)} \in \mathbb{R}^{D \times M}$  and  $\mathbf{b}^{(n,E)} \in \mathbb{R}^D$  correspond to the weights and bias in the embedding layer for joint  $n$ . Unlike RNNs or CNNs, the Transformer has no notion of ordering. Following [11], we inject sinusoidal positional encoding into the joint embeddings, reported to be helpful in extrapolating to longer sequences [11]. After dropouts [30], the joint embeddings are passed to a stack of  $L$  attention blocks where we apply the spatio-temporal attention in parallel to update the embeddings.

**Temporal Attention** In the temporal attention block, the embedding of every joint is updated by using the past frames of the same joint. Formally, for each joint  $n \in \{1..N\}$ , we calculate a temporal summary  $\bar{\mathbf{E}}^{(n)} = [\bar{\mathbf{e}}_1^{(n)}, \dots, \bar{\mathbf{e}}_T^{(n)}]^T \in \mathbb{R}^{T \times D}$ .

We use the scaled dot-product attention proposed by [11], requiring *query*,  $\mathbf{Q}$ , *key*,  $\mathbf{K}$ , and *value*  $\mathbf{V}$  representations. Intuitively, the *value* corresponds to the set of past representations that are indexed by the *keys*. For the joint of interest, we compare its *query* representation with all *keys* w.r.t. the dot-product similarity. If the *query* and the *key* are similar (i.e., high attention weight), then the corresponding *value* is assumed relevant. The attention operation yields a weighted sum of *values*  $\mathbf{V}$ :

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}, \mathbf{M}) = \text{softmax} \left( \frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{D}} + \mathbf{M} \right) \mathbf{V} = \mathbf{A}\mathbf{V}, \quad (2)$$

where the mask  $\mathbf{M}$  prevents information leakage from future steps. For the temporal attention mechanism, we refer to matrix  $\mathbf{A}$  as  $\bar{\mathbf{A}} \in \mathbb{R}^{T \times T}$ . It contains the temporal attention weights, where each row  $i$  in  $\bar{\mathbf{A}}$  represents how much attention is given to the previous frames in the sequence.

The  $\mathbf{Q}$ ,  $\mathbf{K}$ , and  $\mathbf{V}$  matrices are projections of the input embeddings  $\mathbf{E}^{(n)} = [\mathbf{e}_1^{(n)}, \dots, \mathbf{e}_T^{(n)}]^T \in \mathbb{R}^{T \times D}$ . Following [11], we use a multi-head attention (MHA) mechanism to project the  $D$ -dimensional representation into subspaces calculated by different attention heads  $i \in \{1..H\}$ :

$$\begin{aligned} \mathbf{Q}^{(n,i)} &= \mathbf{E}^{(n)} \mathbf{W}^{(n,Q,i)}, \quad \mathbf{Q}^{(n,i)} \in \mathbb{R}^{T \times F} \\ \mathbf{K}^{(n,i)} &= \mathbf{E}^{(n)} \mathbf{W}^{(n,K,i)}, \quad \mathbf{K}^{(n,i)} \in \mathbb{R}^{T \times F} \\ \mathbf{V}^{(n,i)} &= \mathbf{E}^{(n)} \mathbf{W}^{(n,V,i)}, \quad \mathbf{V}^{(n,i)} \in \mathbb{R}^{T \times F} \end{aligned} \quad (3)$$

where we set  $F = D/H$ . Using multiple heads allows the model to gather information from different sets of timesteps into a single embedding. For example, every head in a MHA with 4 heads, outputs 16D chunks of a 64D representation. Hence, different attention heads enable accessing different components. The results are then concatenated and projected back into representation space using the

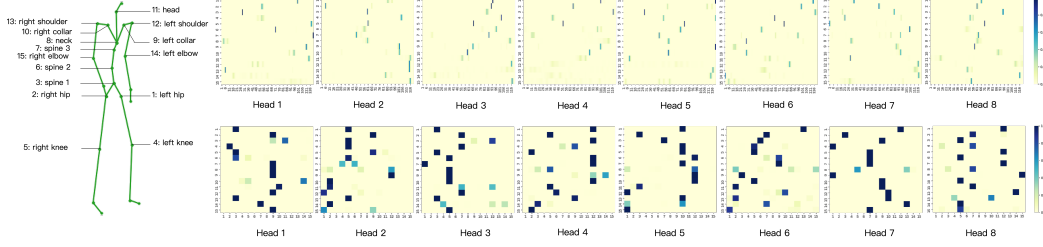


Figure 3: Temporal (*top*) and spatial (*bottom*) attention weights of the first layer given 120 frames. Each row corresponds to a joint’s attention pattern. The columns stand for time-steps and joints for the temporal and spatial attention matrices, respectively. We visualize 8 heads showing that they attend to different joints and timesteps. Similarly, the relevant joints and timesteps differ with respect to the joint of interest.

weight matrix  $\mathbf{W}^{(n,O)} \in \mathbb{R}^{HF \times D}$

$$\begin{aligned} \text{head}_i &= \text{Attention} \left( \mathbf{Q}^{(n,i)}, \mathbf{K}^{(n,i)}, \mathbf{V}^{(n,i)}, \mathbf{M} \right) \\ \tilde{\mathbf{E}}^{(n)} &= \text{Concat}(\text{head}_1, \dots, \text{head}_H) \mathbf{W}^{(n,O)}. \end{aligned} \quad (4)$$

**Spatial Attention** In the vanilla Transformer, the attention block operates on the entire input vector  $\mathbf{x}_t$  where the relation between the elements are implicitly captured. We introduce an additional spatial attention block to learn dynamics and inter-joint dependencies from the data explicitly.

The spatial attention mechanism considers all joints of the same timestep. Moreover, the projections we use to calculate the *key* and *value* are shared across joints. Since we aim to identify the most relevant joints, we project them into the same embedding space and compare with the joint of interest.

For a given pose embedding  $\mathbf{E}_t = [\mathbf{e}_t^{(1)}, \dots, \mathbf{e}_t^{(N)}]^T \in \mathbb{R}^{N \times D}$ , the spatial summary of joints  $\tilde{\mathbf{E}}_t^{(n)}$  is calculated as a function of all other joints by using the multi-head attention:

$$\begin{aligned} \mathbf{Q}_t^{(i)} &= \left[ \left( \mathbf{W}^{(1,Q,i)} \right)^T \mathbf{e}_t^{(1)}, \dots, \left( \mathbf{W}^{(N,Q,i)} \right)^T \mathbf{e}_t^{(N)} \right]^T \\ \mathbf{K}_t^{(i)} &= \mathbf{E}_t \mathbf{W}^{(K,i)} \\ \mathbf{V}_t^{(i)} &= \mathbf{E}_t \mathbf{W}^{(V,i)} \\ \text{head}_i &= \text{Attention}(\mathbf{Q}_t^{(i)}, \mathbf{K}_t^{(i)}, \mathbf{V}_t^{(i)}, \mathbf{0}) = \tilde{\mathbf{A}} \mathbf{V}_t^{(i)} \\ \tilde{\mathbf{E}}_t^{(n)} &= \text{Concat}(\text{head}_1, \dots, \text{head}_H) \mathbf{W}^{(O)} \end{aligned} \quad (5)$$

where  $\mathbf{Q}_t^{(i)} \in \mathbb{R}^{N \times S}$ ,  $\mathbf{K}_t^{(i)} \in \mathbb{R}^{N \times S}$ ,  $\mathbf{V}_t^{(i)} \in \mathbb{R}^{N \times S}$ ,  $S = F = D/H$ ,  $\text{head}_i \in \mathbb{R}^{N \times S}$ , and  $\mathbf{W}^{(O)} \in \mathbb{R}^{HS \times D}$ . The spatial attention  $\tilde{\mathbf{A}} \in \mathbb{R}^{N \times N}$  denotes how much attention a joint  $i$  pays to the other joints  $j$ . Since it iterates over the joints at a single time-step, we no longer require a mask.

**Aggregation** The temporal and spatial attention blocks run in parallel to calculate summaries  $\tilde{\mathbf{E}}$  and  $\tilde{\mathbf{E}}$ , respectively. They are summed and passed to a 2-layer pointwise feedforward network [11], which is followed by a dropout and layer normalization. We stack  $L$  such attention layers to successively update the joint embeddings and thus to refine the pose predictions.

**Joint Predictions** Finally, the joint prediction  $\hat{\mathbf{j}}_{t+1}^{(n)}$  is obtained by projecting the corresponding  $D$ -dimensional embedding  $\mathbf{e}_t^{(n)}$  from the  $L$ -th attention layer back to the  $M$ -dimensional joint angle space. Following [4], we apply a residual connection between the previous pose and the prediction.

### 3.3 Training and Inference

We train our model by predicting the next step both for the seed and the target sequences. We use the per-joint  $\ell_2$  distance on rotation matrices directly [1]:

$$\mathcal{L}(\mathbf{X}, \hat{\mathbf{X}}) = \sum_{t=2}^{T+1} \sum_{n=1}^N \left\| \mathbf{j}_t^{(n)} - \hat{\mathbf{j}}_t^{(n)} \right\|_2$$

Table 1: **AMASS results.** Lower is better for *Euler*, *Joint Angle* and *Positional* metrics. For the *Area Under the Curve (AUC)*, higher is better. ST-Transformer stands for our Spatio-temporal Transformer model. \* indicates our own evaluation of the respective model. For more details how we evaluated [28] please refer to the appendix.

	Euler				Joint Angle				Positional				PCK (AUC)			
milliseconds	100	200	300	400	100	200	300	400	100	200	300	400	100	200	300	400
Zero-Velocity [4, 1]	1.91	5.93	11.36	17.78	0.37	1.22	2.44	3.94	0.14	0.48	0.96	1.54	0.86	0.83	0.84	0.82
Seq2seq [4, 1]	2.01	5.99	11.22	17.33	0.37	1.17	2.27	3.59	0.14	0.45	0.88	1.39	0.86	0.84	0.85	0.83
QuaterNet [5, 1]	1.49	4.70	9.16	14.54	0.26	0.89	1.83	3.00	0.10	0.34	0.71	1.18	0.90	0.87	0.88	0.85
DCT-GCN (ST)* [28]	1.23	4.00	8.05	13.04	0.24	0.77	1.60	2.66	0.09	0.31	0.63	1.06	0.92	0.89	0.89	0.87
DCT-GCN (LT)* [28]	1.27	4.18	8.37	13.38	0.24	0.80	1.65	2.71	0.09	0.31	0.65	1.07	0.91	0.89	0.89	0.87
RNN-SPL [1]	1.33	4.13	8.03	12.84	0.22	0.73	1.51	2.51	<b>0.08</b>	0.28	<b>0.57</b>	<b>0.96</b>	<b>0.93</b>	<b>0.90</b>	<b>0.90</b>	<b>0.88</b>
Transformer	1.30	4.01	7.88	12.69	0.22	0.73	1.52	2.54	<b>0.08</b>	0.28	0.58	0.97	0.92	<b>0.90</b>	<b>0.90</b>	<b>0.88</b>
ST-Transformer	<b>1.11</b>	<b>3.61</b>	<b>7.31</b>	<b>12.04</b>	<b>0.20</b>	<b>0.68</b>	<b>1.45</b>	<b>2.48</b>	<b>0.08</b>	<b>0.27</b>	<b>0.57</b>	0.97	<b>0.93</b>	<b>0.90</b>	<b>0.90</b>	<b>0.88</b>

At test time, we compute the prediction in an auto-regressive manner. That is, given a pose sequence  $\{x_1, \dots, x_T\}$ , we get the prediction  $\hat{x}_{T+1}$ . Due to memory limitations, we apply the temporal attention over a sliding window of poses which we set as the length of the seed sequence. In other words, to produce  $\hat{x}_{T+2}$  we condition on the sequence  $\{x_2, \dots, \hat{x}_{T+1}\}$ .

## 4 Experiments

Traditionally, motion prediction has been benchmarked on the H3.6M [31] dataset. Since our focus lies on generative modeling of diverse long-term predictions, the newly introduced AMASS [21], which is about 14 times the size of H3.6M, is most suitable for our task and we follow [1] in our evaluations. For completeness, we also include results on H3.6M in the appendix, where we perform on par with SoA, sometimes exceeding it.

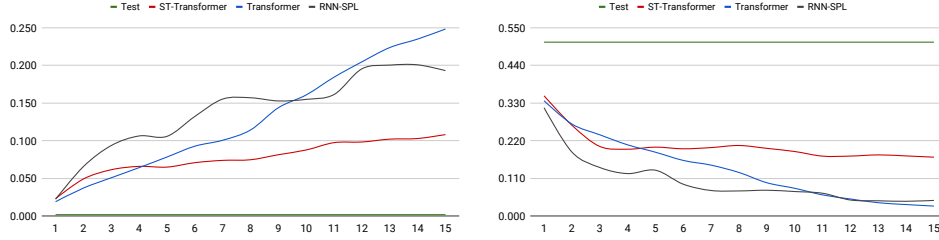
We evaluate the ST-Transformer on AMASS in Sec. 4.1 following the protocol of [1] for short-term predictions and adopting distribution-based metrics for long-term predictions. Sec. 4.2 and Sec. 4.3 show qualitative results and attention weights, thus providing insights into how the model forms predictions. Finally, we validate design choices through ablation studies in Sec. 4.4. More implementation details and experimental settings are provided in the appendix. Code will be made publicly available.

### 4.1 Quantitative Evaluation

**Short-Term** Table 1 summarizes the results with angle- and position metrics up to 400 ms following the protocol of [1]. We compare our ST-Transformer with the vanilla Transformer and previously reported RNN-based architectures. Our model achieves similar or better than state-of-the-art performance. It is also evident that the spatio-temporal decoupling of attention is indeed beneficial when compared to the vanilla Transformer.

Our model improves the most under the Euler metric. This can be explained by the similarity between our training objective and the metric. More specifically, the Euler metric calculates the error by using local joint angles, i.e. angles relative to a joint’s parent. Our model is also trained to predict local angles. Although the underlying representation is different (Euler angles for the metric, rotation matrices for our model), our joint-oriented architecture exploits the local information, thus favoring a locality-based metric. Furthermore, our approach is fully generative and while it maintains local consistency, the global positions may deviate from the ground-truth under natural variation.

**Long-Term** For time horizons longer than 1 second, direct comparison to the ground-truth via MSE becomes increasingly problematic [2]. Hence, in addition to the standard metrics in [1, 4], we conduct further analysis by using complementary metrics in the frequency domain. Hernandez et al. [27] propose distribution-based metrics in the power spectrum (PS) space. While *PS KLD* measures the discrepancy between the prediction and test distributions via the KL divergence, *PS Entropy* measures the entropy in the power spectrum. In Figure 4, we compare our ST-Transformer, the vanilla Transformer and RNN-SPL [1] on these PS metrics for predictions up to 15 seconds. This is the longest prediction horizon presented so far in the literature. We do not include older RNN baselines since they typically collapse to a static pose after 1-2 sec.



(a) **PS KLD.** Symmetric KLD between data and model distributions. Lower is better. (b) **PS Entropy.** Entropy of model distributions. Higher is better.

Figure 4: Power Spectrum (PS) metrics. Our model (red) remains closer to the test dataset for longer predictions. The vanilla Transformer is competitive between 2 – 4 seconds. However, both the vanilla Transformer and the RNN-SPL show degraded performance as prediction horizon increases.

The reference values (green) from the training and test samples are calculated on randomly extracted 1-second windows (60 frames). Similarly, we compute statistics over the predictions (red and blue) by shifting a 1-second window. Thus, we compare every second of the prediction with real 1-second clips. As is expected, the prediction statistics do deviate from the ground-truth statistics with increasing prediction horizon. However, our model remains much closer to the real data statistics than the two baselines including the vanilla transformer. The PS Entropy plot in Fig. 4 shows that the ST-Transformer has a higher entropy than the baselines, thus indicating its power to mitigate the collapse to a static pose. It furthermore indicates that the vanilla transformer cannot alleviate this problem as much as the ST-Transformer. The difference is more pronounced with horizon length.

## 4.2 Qualitative Evaluation

Here, we evaluate the generative capabilities via conditional synthesis up to 20 seconds. In other words, we provide a particular motion sequence of 2 seconds and predict autoregressively beyond its training horizon (i.e., 400 ms).

We qualitatively compare our model with the vanilla Transformer and RNN-SPL on a walking sample from the AMASS dataset in Fig. 5. For RNN-SPL any variation quickly disappears within 5 seconds, indicating that it converges to a static pose. The vanilla Transformer mitigates this problem for longer and shows the benefits of the attention mechanism over recurrent networks. However, it still suffers from the same problem around second 15, whereas our ST-Transformer maintains the periodic pattern up to 20 seconds in this sample. More samples can be found in the appendix and video.

While our model performs well on periodic motions for long horizons, the prediction horizon is limited to a few seconds for aperiodic motion types as the motion cycle is completed. This still

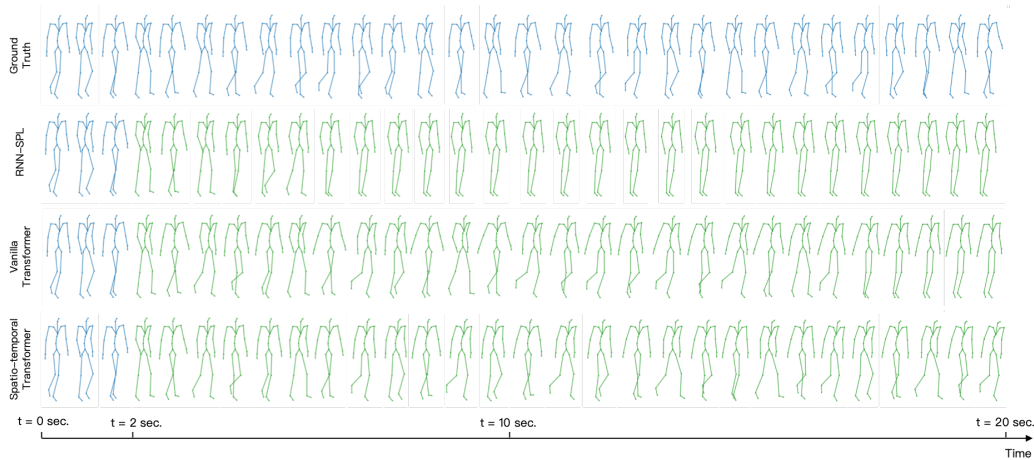


Figure 5: A walking motion predicted by RNN-SPL [1], vanilla Transformer, and our Spatio-temporal Transformer. The length of the seed sequence is 2 seconds and the length of the whole sequence is 20 seconds.

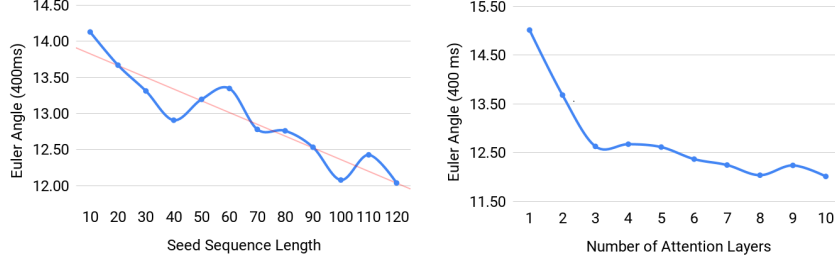


Figure 6: Performance of our model when trained with seed sequence of different lengths (*left*) and different number of attention layers (*right*) on AMASS dataset.

exceeds previously reported horizons significantly. Also, it is not unexpected since the model is unlikely to be exposed to transition patterns as it is trained on rather short 2.4-second windows (i.e., 2 second seed and 400 ms target).

### 4.3 What does Attention Look Like?

The underlying attention mechanism of our model provides insights into the model’s predictions. Fig. 3 visualizes the temporal and spatial attention weights  $\bar{A}$ ,  $\tilde{A}$  taken from an AMASS sequence. Those weights are used to predict the first frame given the seed sequence of 120 frames. In our visualizations, we used the first layer as it uses the initial joint embeddings directly. In the upper layers, the information is already gathered and hence the attention is dispersed.

First, we observe that there is a diverse set of attention patterns across heads, enriching the representation through gathering information from multiple sources. Second, the temporal attention weights reveal that the model is able to attend over a long horizon. While some heads focus on near frames, some look to the very beginning which would be difficult for an RNN. Finally, in the spatial attention, we observe joint-dependencies not only on the kinematic chain but also across the left and right parts of the skeleton. For example, while predicting the left knee joint, the model attends to the spine, left collar and right hip, knee and collar joints. Similarly, for the right elbow, the most informative joints are spine, the left collar, the hips and knee joints. To see how attention weights change over time, as the model constructs the next predictions, please refer to Fig. 7 in the appendix and the video.

### 4.4 Ablations

Fig. 6 plots the performance of our model when trained with seed sequences of varying length, showing the performance w.r.t. the temporal attention window. The decreasing trend suggests that our model benefits from longer sequences. This hypothesis is supported via the temporal attention masks showing that our model accesses poses from the beginning of the sequence (cf. Fig. 3 and Fig. 7 in the appendix).

We train our model with varying number of layers. Fig. 6 shows that reasonable performance on AMASS is reached with only 3 layers. In our AMASS experiments, we use a model with 8 layers trained on a seed sequences of 120 steps whereas a 4-layer architecture with 50 input frames is used on the smaller H3.6M dataset.

## 5 Conclusion

We introduce a novel spatio-temporal transformer (ST-Transformer) network for generative modeling of 3D human motion. We posit that the task can be interpreted as a pure generative modeling task and propose a novel architecture that learns intra- and inter-joint dependencies via its decoupled temporal and spatial attention blocks. We show that the self-attention concept is very effective in making both short- and long-term predictions. It mitigates the long-term dependency issue observed in RNN-based architectures and is able to synthesize motion sequences up to 20 seconds conditioned on periodic motion types such as walking or running. We also demonstrate that the attention mechanism can be used to obtain insights about the model’s behavior. Finally, our ablation studies suggest that our model can make efficient use of large datasets with long seed sequences such as the recent AMASS.



## Broader Impact

Human motion prediction is a building block in several domains such as robotics, autonomous driving, or computer graphics. Applied to robotics, a motion model could be used to make robot-human interaction safer in shared workspaces. Similarly, an autonomous car will be able to take different, overall safer actions, inferring how a pedestrian behaves in the near-term future. In computer graphics, a generative motion model can be used to simulate motion more easily, or to reduce latency (and thus possibly motion sickness) in VR/AR headsets.

Whether motion prediction will have negative effects thus largely depends on its use in the concrete application domain. The common denominator for most applications is that motion modelling can be seen as a driver for automation. Automation in turn always bears the risk of potentially putting existing workers out of a job. At this point, it is however extremely hard to estimate the extent of which motion modeling efforts will contribute to this. This is partially due to such methods still being in their infancy.

## Acknowledgments and Disclosure of Funding

This project has received funding from the European Research Council (ERC) under the European Union’s Horizon 2020 research and innovation programme grant agreement No 717054. We thank the NVIDIA Corporation for the donation of GPUs used in this work.

## References

- [1] Emre Aksan, Manuel Kaufmann, and Otmar Hilliges. Structured prediction helps 3d human motion modelling. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2019.
- [2] Katerina Fragkiadaki, Sergey Levine, Panna Felsen, and Jitendra Malik. Recurrent network models for human dynamics. In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ICCV ’15, pages 4346–4354, Washington, DC, USA, 2015. IEEE Computer Society. ISBN 978-1-4673-8391-2. doi: 10.1109/ICCV.2015.494. URL <http://dx.doi.org/10.1109/ICCV.2015.494>.
- [3] Ashesh Jain, Amir Roshan Zamir, Silvio Savarese, and Ashutosh Saxena. Structural-rnn: Deep learning on spatio-temporal graphs. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 5308–5317, 2016. doi: 10.1109/CVPR.2016.573. URL <https://doi.org/10.1109/CVPR.2016.573>.
- [4] Julieta Martinez, Michael J. Black, and Javier Romero. On human motion prediction using recurrent neural networks. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2017*, Piscataway, NJ, USA, July 2017. IEEE.
- [5] Dario Pavullo, David Grangier, and Michael Auli. Quaternet: A quaternion-based recurrent model for human motion. In *British Machine Vision Conference 2018, BMVC 2018, Northumbria University, Newcastle, UK, September 3-6, 2018*, page 299, 2018. URL <http://bmvc2018.org/contents/papers/0675.pdf>.
- [6] Yuxiong Wang, Liang-Yan Gui, Xiaodan Liang, and Jose M. F. Moura. Adversarial geometry-aware human motion prediction. In *European Conference on Computer Vision (ECCV)*. Springer, October 2018.
- [7] Chen Li, Zhen Zhang, Wee Sun Lee, and Gim Hee Lee. Convolutional sequence to sequence model for human dynamics. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.
- [8] Yanran Li, Zhao Wang, Xiaosong Yang, Meili Wang, Sebastian Iulian Poiana, Ehtzaz Chaudhry, and Jianjun Zhang. Efficient convolutional hierarchical autoencoder for human motion prediction. *The Visual Computer*, 35(6-8):1143–1156, 2019.
- [9] Marc’Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732*, 2015.
- [10] Partha Ghosh, Jie Song, Emre Aksan, and Otmar Hilliges. Learning human motion models for long-term predictions. In *2017 International Conference on 3D Vision, 3DV 2017, Qingdao, China, October 10-12, 2017*, pages 458–466, 2017. doi: 10.1109/3DV.2017.00059. URL <https://doi.org/10.1109/3DV.2017.00059>.

- [11] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc., 2017. URL <http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf>.
- [12] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.
- [13] Cheng-Zhi Anna Huang, Ashish Vaswani, Jakob Uszkoreit, Ian Simon, Curtis Hawthorne, Noam Shazeer, Andrew M. Dai, Matthew D. Hoffman, Monica Dinculescu, and Douglas Eck. Music transformer. In *International Conference on Learning Representations*, 2019. URL <https://openreview.net/forum?id=rJe4ShAcF7>.
- [14] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019.
- [15] Niki J. Parmar, Ashish Vaswani, Jakob Uszkoreit, Łukasz Kaiser, Noam Shazeer, and Alexander Ku. Image transformer. 2018. URL <https://arxiv.org/abs/1802.05751>.
- [16] Rami Al-Rfou, Dokook Choe, Noah Constant, Mandy Guo, and Llion Jones. Character-level language modeling with deeper self-attention. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 3159–3166, 2019.
- [17] Ankur P Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. A decomposable attention model for natural language inference. *arXiv preprint arXiv:1606.01933*, 2016.
- [18] Hsu-Kuang Chiu, Ehsan Adeli, Borui Wang, De-An Huang, and Juan Carlos Niebles. Action-agnostic human pose forecasting. *2019 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 1423–1432, 2018.
- [19] Xiaoxiao Du, Ram Vasudevan, and Matthew Johnson-Roberson. Bio-Istm: A biomechanically inspired recurrent neural network for 3d pedestrian pose and gait prediction. *IEEE Robotics and Automation Letters (RA-L)*, 2019. accepted.
- [20] Borui Wang, Ehsan Adeli, Hsu-kuang Chiu, De-An Huang, and Juan Carlos Niebles. Imitation learning for human pose prediction. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.
- [21] Naureen Mahmood, Nima Ghorbani, Nikolaus F. Troje, Gerard Pons-Moll, and Michael J. Black. Amass: Archive of motion capture as surface shapes. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2019. URL <https://amass.is.tue.mpg.de>.
- [22] Yi Zhou, Zimo Li, Shuangjiu Xiao, Chong He, Zeng Huang, and Hao Li. Auto-conditioned recurrent networks for extended complex human motion synthesis. In *International Conference on Learning Representations*, 2018. URL <https://openreview.net/forum?id=r11Q2S1RW>.
- [23] Judith Bütetage, Michael J. Black, Danica Kragic, and Hedvig Kjellström. Deep representation learning for human motion prediction and classification. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1591–1599, 2017.
- [24] Judith Bütetage, Hedvig Kjellström, and Danica Kragic. Anticipating many futures: Online human motion prediction and generation for human-robot interaction. In *2018 IEEE International Conference on Robotics and Automation, ICRA 2018, Brisbane, Australia, May 21-25, 2018*, pages 1–9, 2018. doi: 10.1109/ICRA.2018.8460651. URL <https://doi.org/10.1109/ICRA.2018.8460651>.
- [25] Daniel Holden, Jun Saito, and Taku Komura. A deep learning framework for character motion synthesis and editing. *ACM Trans. Graph.*, 35(4):138:1–138:11, July 2016. ISSN 0730-0301. doi: 10.1145/2897824.2925975. URL <http://doi.acm.org/10.1145/2897824.2925975>.
- [26] Daniel Holden, Jun Saito, Taku Komura, and Thomas Joyce. Learning motion manifolds with convolutional autoencoders. In *SIGGRAPH Asia 2015 Technical Briefs*, SA ’15, pages 18:1–18:4, New York, NY, USA, 2015. ACM. ISBN 978-1-4503-3930-8. doi: 10.1145/2820903.2820918. URL <http://doi.acm.org/10.1145/2820903.2820918>.
- [27] Alejandro Hernandez, Jurgen Gall, and Francesc Moreno-Noguer. Human motion prediction via spatio-temporal inpainting. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.

- [28] Wei Mao, Miaomiao Liu, Mathieu Salzmann, and Hongdong Li. Learning trajectory dependencies for human motion prediction. In *The IEEE International Conference on Computer Vision (ICCV)*, October 2019.
- [29] Dirk Weissenborn, Oscar Täckström, and Jakob Uszkoreit. Scaling autoregressive video models. In *International Conference on Learning Representations*, 2020. URL <https://openreview.net/forum?id=rJgsskrFwH>.
- [30] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1): 1929–1958, 2014.
- [31] 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, jul 2014.
- [32] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [33] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL <https://www.tensorflow.org/>. Software available from tensorflow.org.

## 6 Appendix

In these supplementary materials, we provide additional details on the implementation of the proposed ST-Transformer and further experimental evidence of its performance. We visualize more attention weights in Sec. 6.1 and explain the multi-head attention mechanism in more detail in Sec. 6.2. Sec. 6.3 details how we evaluated the DCT-GCN model [28] and Sec. 6.4 provides definitions and more results of the Power Spectrum metrics. Finally Sec. 6.5 presents the results of the ST-Transformer on H3.6M [31] and Sec. 6.6 lists hyper-parameters we used to train models on AMASS and H3.6M.

### 6.1 What does Attention Look Like?

In order to adapt to changing spatio-temporal patterns, our model calculates the attention weights at every step. In Fig. 7, we visualize the change in the attention weights over time. The focus of the network changes as expected. Although the attention window shifts in time, we observe that for quasi-static joints like the hips or spine, the model maintains the focus on the same time-step in this particular attention head. When we calculate the inter-joint dependencies via spatial attention, a large number of joints attend to the left and right collars. This could indicate that such mostly static joints are used as reference.

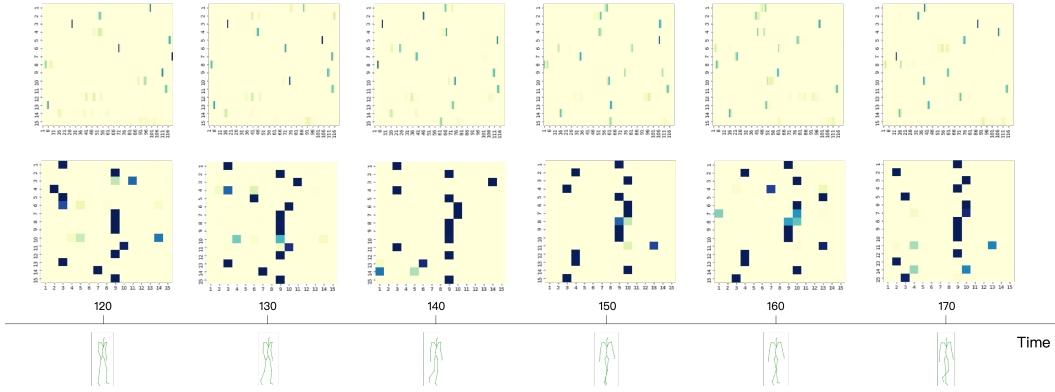


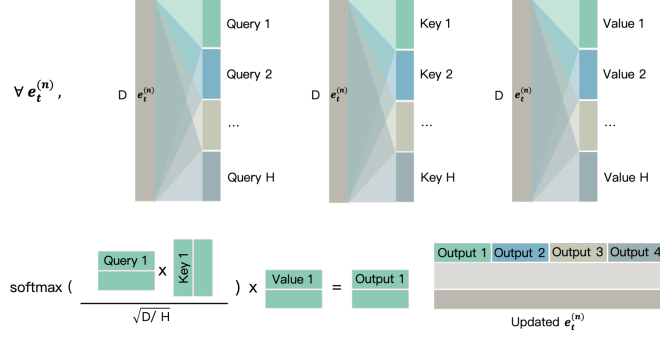
Figure 7: Temporal (*top*) and spatial (*middle*) attention weights aligned with the poses (*bottom*) over 1 second prediction. We visualize a single head in the first layer. Our model identifies the informative joints dynamically. Note that the temporal attention is calculated over a sliding window. Some of the past frames remain in the focus of the temporal attention despite the shift in the inputs. Similarly, the spatial focus is preserved on the more static joints such as left and right collars (i.e., columns 9 and 10 in the middle row).

### 6.2 Multi-head Attention

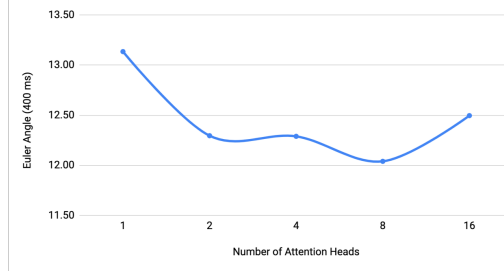
Fig. 8a illustrates the multi-head self-attention mechanism mentioned in Eq. (3) and (4) of the main paper. The sub-spaces of query, key, and value for each attention head are calculated from the joint embeddings  $e_t^{(n)}$ . From this figure it becomes clear that the joint embedding size  $D$  must be evenly divisible by the number of heads  $H$ . This in turn means the projection matrices involved for query, key, and value are of dimension  $\mathbb{R}^{D \times \ell}$ , where  $\ell = D/H$ . In the main paper we refer to  $\ell$  as either  $S$  or  $F$  to distinguish between the temporal and spatial attention layer.

In the temporal attention blocks, we use separate query, key, and value weight matrices for different joints. In the spatial attention blocks, while the key and value weight matrices are shared across joints, the query weight matrices are not. Having obtained all the query, key, and value embeddings, we can get the output of each head according to Eq. (4) in the main submission. Note that the attention is over  $T$  time steps in the temporal attention blocks and  $N$  joints in the spatial attention blocks. Finally, the outputs of all heads are concatenated and then fed to a feed forward network consisting of two dense layers and computing the updated embedding of  $\bar{E}^{(n)}$ .

We experimented with varying number of attention heads  $H$ . As shown in Fig. 8b, the best performance on AMASS is achieved with 8 attention heads. A model with 2 attention heads also yields reasonable performance. Comparison between the performance of multi- and single-head attention mechanism suggests that the model benefits from using more than one spatio-temporal configuration.



(a) The multi-head self-attention mechanism. Each color represents one head.



(b) Performance of our model when trained with different number of attention heads on AMASS.

Figure 8: Multi-head attention. Visual depiction and ablation on the number of heads  $H$ .

### 6.3 Evaluation of DCT-GCN on AMASS

In Tab. 1 of the main paper we report the performance of the DCT-GCN model [28] on the AMASS dataset. The results correspond to the best results we obtained after hyper-parameter-tuning, explained in more detail in the following.

To train and evaluate the DCT-GCN on AMASS we used the code provided by Mao et al. [28] but swapped out the data loading pipeline to load AMASS instead of H3.6M. As in the original paper, we train two networks - one for short-term (up to 400 milliseconds) and one for long-term (up to 1 second) predictions. In [28] the input to the network is 400 milliseconds (10 frames) worth of data. As AMASS is sampled at 60 Hz, we hence feed 24 frames as input and let the model predict 24 frames in the short-term case and 60 frames in the long-term case.

We fine-tuned the learning rate as well as the number of DCT coefficients. For the number of DCT coefficients we tried the original 35, but also the maximum number of coefficients (48 for short-term, 84 for long-term). As reported by [28] we found this to make little difference, but using 35 coefficients led to slightly better results.

The remaining hyper-parameters are as follows, which mostly corresponds the original setting. We are employing the Adam [32] optimizer with a learning rate of 0.001 and batch size of 16. We train for maximum 100 epochs with early stopping. The learning rate is decayed by a factor of 0.96 every other epoch. The input window size for the short-term model is 48 frames and 84 frames for the long-term model. The model parameters are kept as proposed in the original paper resulting in a model size of roughly 2.23 Mio. parameters.

### 6.4 Power Spectrum Metrics

On AMASS, we aim to show our model’s capability in making very long predictions (i.e., up to 15 – 20 sec). Such long prediction horizons prevent us from using pairwise metrics such as the MSE because the ground-truth targets are often much shorter than the prediction horizon. Hence, we use Power Spectrum (PS) metrics originally proposed by Hernandez et al. [27] for the H3.6M benchmark.

In order to adapt the metrics into our new setup, we slightly modify the evaluation protocol. We use 3D joint positions instead of angles as it is straightforward to convert any angle-based representation into positions by applying forward kinematics. This also allow us to compare models operating

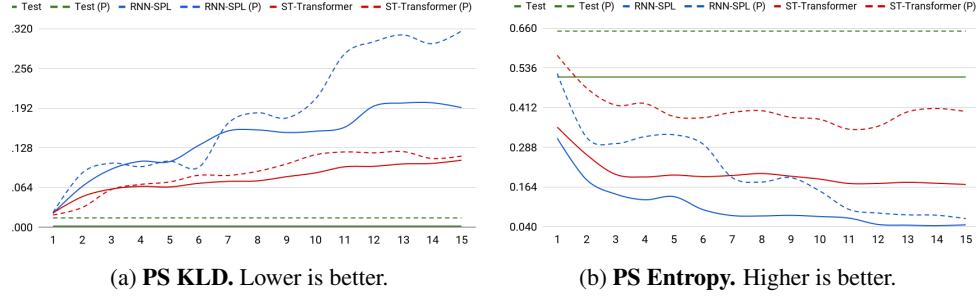


Figure 9: **PS Metrics for periodic motions.** We compare our ST-Transformer with RNN-SPL [1] on the entire test set and a subset of periodic motions, indicated by \*.P and dashed lines. We also provide the metric result on the real test distribution  $G$  as a reference. For PS Entropy, lower values are an indication of a lack of activity (convergence to a static pose especially for long prediction horizons). We observe that the RNN-based model clearly suffers from this problem. In contrast, our model alleviates this issue thanks to the spatio-temporal self-attention mechanism. For PS KLD, the prediction quality degrades as the prediction horizon increases. While this is expected, our model’s predictions remain closer to the real test distribution than the baselines.

on arbitrary rotation representations. Given a sequence  $\mathbf{X}$ , we treat every coordinate of every joint over time as a feature sequence  $\mathbf{x}_f$  following [27]. The power spectrum PS is then equal to  $PS(\mathbf{x}_f) = \|FFT(\mathbf{x}_f)\|^2$  where  $FFT$  denotes the Fast Fourier Transform.

**PS Entropy** It is defined as

$$PS\ Entropy(\mathcal{X}) = \frac{1}{|\mathcal{X}|} \sum_{\mathbf{X} \in \mathcal{X}} \frac{1}{F} \sum_{f=1}^F \sum_{e=1}^E -\|PS(\mathbf{x}_f)\| * \log(\|PS(\mathbf{x}_f)\|)$$

where  $\mathcal{X}$  is either the ground-truth test or training dataset, or the predictions made of a respective model on the corresponding test dataset.  $f$  and  $e$  correspond to a feature and frequency, respectively.

**PS KLD** To compute the PS KLD metric, we use the following approach. Instead of using the variable-length ground-truth targets, we randomly get 20’000 sequences of length 1 sec (i.e., 60 frames) from the test dataset and calculate the power spectrum distribution  $G$ . Then, we get non-overlapping windows of 1 sec from the predictions to get  $P_t$  where  $t$  stands for the corresponding prediction window of length 1 second. For example,  $P_5$  is the power spectrum distribution for the predictions between 5 and 6 seconds. This enables us to measure the quality of arbitrarily long predictions by comparing every second of the predictions with the real reference data.

The symmetric PS KLD metric is then defined as

$$PS\ KLD(G, \mathcal{X}, t) = \frac{1}{2|\mathcal{X}|} \sum_{P_t \in \mathcal{X}} KLD(G \parallel P_t) + KLD(P_t \parallel G)$$

We use publicly available implementations to compute the metrics ([27], Github link). It is worthwhile to mention that the PS KLD metric does not make pairwise comparisons between ground-truth and predictions. Instead, it measures the discrepancy between the real and predicted data distributions.

In Fig. 4 of the main paper we report the PS KLD and PS Entropy metrics for various  $P_t$  where  $t \in \{0..15\}$ . In Fig. 9 we show further results where we compute the metrics on a subset of sequences containing periodic motion samples. Fig. 9 indicates that the ST-Transformer alleviates the mean pose convergence both for periodic and non-periodic motions and in both cases outperforms the RNN-based baselines considerably. For more qualitative analysis, please refer to the video.

## 6.5 H3.6M Results

H3.6M dataset [31] has been an established benchmark for 3D motion prediction tasks. However, our focus lies on generative modeling of diverse long-term predictions, for which the newly introduced AMASS is more suitable. For completeness we still evaluate our ST-Transformer on H3.6M following the usual protocols (Sec. 6.5.1). We furthermore provide results on H3.6M using the PS metrics (Sec. 6.5.2).

Considering the evaluation results with MSE, PS KLD, and PS Entropy metrics on H3.6M, our model is comparable to the baseline models and achieves SoA in specific cases. We would like to note that H3.6M evaluation exhibits high variance due to its limited size as previously stated in Pavlo et al. [5] and Aksan et al. [1]. Our detailed evaluation and inconclusive comparisons presented in the following also imply that the H3.6M dataset is possibly saturated.

### 6.5.1 Euler Angle Metrics

Table 2 summarizes the results for the four activities most commonly reported in the literature. Our ST-Transformer outperforms the vanilla Transformer, showing the benefits of our decoupled attention approach for motion modelling. AGED [6], the recently proposed cosine transformation based technique (dubbed DCT-GCN) [28], and our model share the state-of-the-art entries. The DCT-GCN method particularly excels in the *walking* category. We would like to note that DCT-GCN is trained with a pre-determined prediction horizon as it is designed to predict the entire 400 ms (*ST*) or 1 sec (*LT*) in one go. In contrast, Seq2seq, AGED, and our model follow an auto-regressive approach. In other words, we can make arbitrarily long predictions at test time while DCT-GCN is limited to a particular horizon.

In Tab. 3 we provide short-term (i.e. up to 400 ms) evaluation results for the remaining 11 actions not covered in Tab. 2. We observe that this table continues the trend reported in Tab. 2, namely that ST-Transformer sometimes outperforms the state of the art, and otherwise performs similarly. Finally, Tab. 4, reports long-term (i.e., up to 1 sec) evaluations for 4 main actions which have been reported by the baseline methods.

Table 2: **H3.6M results** for *walking*, *eating*, *smoking* and *discussion* activities. Values are the Euler angle metric measured at the given time step (lower is better). ST-Transformer stands for our Spatio-temporal Transformer model, consistently outperforming the vanilla Transformer. DCT-GCN (ST) [28] stands for the short-term model which is designed to make predictions up to 400 ms. DCT-GCN (LT) has a 1 sec prediction horizon while the remaining models can make arbitrarily long predictions auto-regressively.

milliseconds	Walking				Eating				Smoking				Discussion			
	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400
Zero-Velocity [4]	0.39	0.68	0.99	1.15	0.27	0.48	0.73	0.86	0.26	0.48	0.97	0.95	0.31	0.67	0.94	1.04
Seq2seq. [4]	0.28	0.49	0.72	0.81	0.23	0.39	0.62	0.76	0.33	0.61	1.05	1.15	0.31	0.68	1.01	1.09
AGED [6]	0.22	0.36	0.55	0.67	0.17	<b>0.28</b>	0.51	0.64	0.27	0.43	<b>0.82</b>	0.84	0.27	0.56	<b>0.76</b>	<b>0.83</b>
DCT-GCN (ST) [28]	<b>0.18</b>	<b>0.31</b>	<b>0.49</b>	<b>0.56</b>	<b>0.16</b>	0.29	0.50	0.62	<b>0.22</b>	<b>0.41</b>	0.86	<b>0.80</b>	0.20	<b>0.51</b>	0.77	0.85
DCT-GCN (LT) [28]	0.20	0.34	0.52	0.59	0.17	0.31	0.52	0.64	0.23	0.42	0.85	<b>0.80</b>	0.22	0.58	0.87	0.96
Transformer	0.25	0.42	0.67	0.79	0.21	0.32	0.54	0.68	0.26	0.49	0.94	0.90	0.31	0.67	0.95	1.04
ST-Transformer	0.21	0.36	0.58	0.63	0.17	0.30	<b>0.49</b>	<b>0.60</b>	<b>0.22</b>	0.43	0.88	0.82	<b>0.19</b>	0.52	0.79	0.88

### 6.5.2 Power Spectrum Metrics

The MSE metric on Euler angles is considered to be the de facto evaluation protocol in 3D motion prediction tasks on H3.6M. For completeness, we also evaluate H3.6M under the PS Metrics as proposed by Hernandez et al. [27] and which we adopted for our evaluations of AMASS as well. Tab. 5 lists the results comparing our model with DCT-GCN, which currently holds the SoA on H3.6M.

For a definition of the PS metrics, please refer to Sec. 6.4. In contrast to the PS metrics on AMASS, we compute the PS metrics on H3.6M using the Euler angle representation, which follows Hernandez et al. [27]. In contrast to [27], we do not use global translation and rotation, and consider only the active joints. In other words, we use the same predictions and targets that are used to calculate the MSE metric in Tab. 2. Also note that our reported numbers correspond to the symmetric KL-divergence. Hence, we note that the STMI-GAN [27] results are not directly comparable because STMI-GAN considers global translation and rotation predictions which all other models, including ours, do not.

In Tab. 5, PS KLD and PS Entropy metrics are computed over the corresponding sequence length (i.e., 400 ms for short-term and 1 sec for long-term prediction). We observe that neither our ST-Transformer nor the current state-of-the-art DCT-GCN model can consistently outperform the other across all metrics.

Table 3: **Short-term H3.6M results** for the remaining activities omitted in Tab. 2. Values are the Euler angle metric measured at the given time step (lower is better). DCT-GCN (ST) [28] stands for the short-term model which is designed to make predictions up to 400 ms. DCT-GCN (LT) has a 1 sec prediction horizon while the remaining models can make arbitrarily long predictions auto-regressively.

	Directions				Greeting				Phoning				Posing			
milliseconds	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400
Seq2seq. [4]	0.26	0.47	0.72	0.84	0.75	1.17	1.74	1.83	0.23	0.43	0.69	0.82	0.36	0.71	1.22	1.48
AGED [6]	<b>0.23</b>	0.39	<b>0.63</b>	<b>0.69</b>	0.56	0.81	1.30	1.46	<b>0.19</b>	<b>0.34</b>	<b>0.50</b>	<b>0.68</b>	0.31	0.58	1.12	1.34
DCT-GCN (ST) [28]	0.26	0.45	0.71	0.79	0.36	<b>0.60</b>	<b>0.95</b>	<b>1.13</b>	0.53	1.02	1.35	1.48	<b>0.19</b>	<b>0.44</b>	<b>1.01</b>	<b>1.24</b>
DCT-GCN (LT) [28]	0.29	0.47	0.69	0.76	0.36	0.61	0.97	1.14	0.54	1.03	1.34	1.47	0.21	0.47	1.07	1.31
ST-Transformer	0.25	<b>0.38</b>	0.75	0.86	<b>0.35</b>	0.61	1.10	1.32	0.53	1.04	1.41	1.54	0.61	0.68	1.05	1.28

	Purchases				Sitting				Sitting down				Taking photos			
milliseconds	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400
Seq2seq. [4]	0.51	0.97	1.07	1.16	0.41	1.05	1.49	1.63	0.39	0.81	1.40	1.62	0.24	0.51	0.90	1.05
AGED [6]	0.46	0.78	<b>1.01</b>	<b>1.07</b>	0.41	0.76	1.05	1.19	0.33	0.62	0.98	1.10	0.23	0.48	0.81	0.95
DCT-GCN (ST) [28]	<b>0.43</b>	<b>0.65</b>	1.05	1.13	<b>0.29</b>	<b>0.45</b>	0.80	0.97	<b>0.30</b>	<b>0.61</b>	<b>0.90</b>	<b>1.00</b>	<b>0.14</b>	<b>0.34</b>	<b>0.58</b>	<b>0.70</b>
DCT-GCN (LT) [28]	0.50	0.72	1.06	1.12	0.31	0.46	<b>0.79</b>	<b>0.95</b>	0.31	0.64	0.94	1.07	0.17	0.38	0.62	0.74
ST-Transformer	<b>0.43</b>	0.77	1.30	1.37	<b>0.29</b>	0.46	0.84	1.01	0.32	0.66	0.98	1.10	0.15	0.38	0.64	0.75

	Waiting				Walking dog				Walking together				Average			
milliseconds	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400
Seq2seq. [4]	0.28	0.53	1.02	1.14	0.56	0.91	1.26	1.40	0.31	0.58	0.87	0.91	0.36	0.67	1.02	1.15
AGED [6]	0.24	<b>0.50</b>	1.02	<b>1.13</b>	0.50	0.81	1.15	1.27	0.23	0.41	0.56	0.62	0.31	0.54	0.85	0.97
DCT-GCN (ST) [28]	0.23	<b>0.50</b>	<b>0.91</b>	1.14	0.46	0.79	1.12	1.29	<b>0.15</b>	<b>0.34</b>	<b>0.52</b>	<b>0.57</b>	<b>0.27</b>	<b>0.52</b>	<b>0.83</b>	<b>0.95</b>
DCT-GCN (LT) [28]	0.25	0.52	0.96	1.17	0.49	0.80	<b>1.11</b>	<b>1.26</b>	0.18	0.39	0.56	0.63	0.30	0.54	0.86	0.97
ST-Transformer	<b>0.22</b>	0.51	0.98	1.22	<b>0.43</b>	<b>0.78</b>	1.15	1.30	0.17	0.37	0.58	0.62	0.30	0.55	0.90	1.02

Table 4: **Long-term H3.6M results** for the four main activities as commonly reported in the literature. Values are the Euler angle metric measured at the given time step (lower is better).

	Walking		Eating		Smoking		Discussion		Average	
milliseconds	560	1000	560	1000	560	1000	560	1000	560	1000
Seq2seq. [4]	0.93	1.03	0.95	1.08	1.25	1.50	1.43	1.69	1.14	1.33
AGED [6]	0.78	0.91	0.86	<b>0.93</b>	1.06	<b>1.21</b>	<b>1.25</b>	<b>1.30</b>	0.99	<b>1.09</b>
DCT-GCN (LT) [28]	<b>0.65</b>	<b>0.67</b>	0.76	1.12	<b>0.87</b>	1.57	1.33	1.70	<b>0.90</b>	1.27
ST-Transformer	0.72	0.78	<b>0.73</b>	1.05	0.89	1.57	1.29	1.85	0.91	1.31

Table 5: **PS Metrics on H3.6M** in the short- (400 ms) and long-term (1 sec) setting. We present the average of the four main actions (i.e., *walking*, *eating*, *smoking*, and *discussion*) and all 15 actions in the dataset. For example, *Main (400 ms)* shows the average over the four main actions for predictions up to 400 ms. DCT-GCN (ST) makes predictions only up to 400 ms. The *Euler* metric is the standard MSE metric taken from Tab. 2. The *Test set* entry is provided for reference and corresponds to the PS metrics evaluated on the test set. The PS KLD entries for the test set row is calculated between the seed and target data distributions. For *Euler* and *PS KLD* lower is better, for *PS Ent* higher (i.e. closer to the Test set entry) is better.

	Main (400 ms)			All (400 ms)			Main (1000 ms)			All (1000 ms)		
metrics	Euler	PS KLD	PS Ent	Euler	PS KLD	PS Ent	Euler	PS KLD	PS Ent	Euler	PS KLD	PS Ent
Test set	-	0.008	0.137	-	0.011	0.145	-	0.005	0.403	-	0.002	0.427
DCT-GCN (ST) [28]	<b>0.71</b>	0.018	0.048	<b>0.95</b>	0.048	0.037	-	-	-	-	-	-
DCT-GCN (LT) [28]	0.75	0.020	0.044	0.97	0.052	0.034	<b>1.26</b>	<b>0.065</b>	0.148	<b>1.590</b>	0.096	0.110
ST-Transformer	0.73	<b>0.015</b>	<b>0.061</b>	1.02	<b>0.040</b>	<b>0.054</b>	1.31	0.072	<b>0.155</b>	1.722	<b>0.091</b>	<b>0.128</b>

## 6.6 Experimental Details

We implementend all models using TensorFlow [33]. Hyper-parameters we used for the experiments are listed in Tab. 6. Due to the limited data size of H3.6M, we achieved better results by using a smaller network.

As suggested by Vaswani et al. [11], the Transformer architecture is sensitive to the learning rate. We apply the same learning rate schedule as proposed in [11]. The learning rate is calculated as a function of the training step as follows:

$$\text{learning rate} = D^{-0.5} \cdot \min(\text{step}^{-0.5}, \text{step} \times \text{warmup}^{-1.5}) \quad ,$$



Table 6: Hyper-parameters for H3.6M and AMASS experiments.

	H3.6M	AMASS
Batch Size	32	32
Window size (num. frames)	75	120
Num. Attention Blocks (L)	4	8
Joint Embedding Size (D)	64	128
Num. Attention Heads (H)	4	8
Feedforward Size	128	256
Dropout Rate	0.1	0.1

where  $D$  is the joint embedding size. Warmup is set to 10000.

We use a batch size of 32 and the Adam optimizer [32] with its default parameters. Following the training protocol in [1], we apply early stopping with respect to the joint angle metric. Since our approach does not fall into the category of sequence-to-sequence (seq2seq) models, we use the entire sequence (i.e., seed and target) for training. On H3.6M, the temporal attention window size is set to 75 frames (i.e., 2-sec seed and 1-sec target at 25 fps). On AMASS, we fed the model with sequences of 120 frames (2-sec seed at 60 fps) due to memory limitations. We followed an auto-regressive approach and train our model by predicting the next pose given the frames so far. In other words, the input sequence is shifted by 1 step to obtain the target frames.

Each attention block contains a feed forward network after the temporal and spatial attention layers. This feed forward network consists of two dense layers where the first one maps the  $D = 128$  dimensional joint embeddings into 256-dimensional space for AMASS (128-dimensional space for H3.6M), followed by a ReLU activation function. The second dense layer always projects back into the  $D$ -dimensional joint embedding space. We use the same dropout rate of 0.1 for all dropout layers in our network.