A Spatio-temporal Transformer for 3D Human Motion Prediction

A Spatio-temporal Transformer for 3D Human Motion Prediction

Emre Aksan[1] Manuel Kaufmann[1] Peng Cao[2*] Otmar Hilliges[1]
[1]ETH Zürich, Department of Computer Science [2]Massachusetts Institute of Technology
[1]{eaksan,kamanuel,otmarh}@inf.ethz.ch [2]pengcao@mit.edu

Abstract

We propose a novel Transformer-based architecture for the task of generative modelling of 3D human motion. Previous work commonly relies on RNN-based models considering shorter forecast horizons reaching a stationary and often implausible state quickly. Recent studies show that implicit temporal representations in the frequency domain are also effective in making predictions for a predetermined horizon. Our focus lies on learning spatio-temporal representations autoregressively and hence generation of plausible future developments over both short and long term. The proposed model learns high dimensional embeddings for skeletal joints and how to compose a temporally coherent pose via a decoupled temporal *and* spatial *self-attention mechanism. Our dual attention concept allows the model to access current and past information directly and to capture both the structural and the temporal dependencies explicitly. We show empirically that this effectively learns the underlying motion dynamics and reduces error accumulation over time observed in auto-regressive models. Our model is able to make accurate short-term predictions and generate plausible motion sequences over long horizons. We make our code publicly available at* `https://github.com/eth-ait/motion-transformer`.

1. Introduction

3D human motion modelling is typically formulated as the prediction of future poses given a past horizon. Humans are able to effortlessly forecast the complex dynamics of motion in a plausible fashion due to our strong structural and temporal priors. From a learning perspective this problem can be seen as a generative modelling task: A network learns to synthesize a sequence of human poses, where the model is conditioned on the seed sequence. The task requires learning of pose priors for natural articulation and of underlying dynamics to yield plausible motion predictions. Since these factors are highly latent and entangled, introducing inductive biases and tailoring architectures for the task is essential for modelling of 3D human motion data.

Given the temporal nature of human motion, it is not surprising that recurrent neural networks (RNNs) are the most popular choice [2, 11, 20, 27, 31, 37]. 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 [13, 21, 23]. Such approaches focus on modelling the temporal aspect of the problem following an auto-regressive approach, but neglect structural priors. Instead, vectorized poses are passed as inputs at every step and the spatial dependencies are assumed to be learned implicitly. However, considering the skeletal structure in the architectural level is shown to be an effective inductive bias in [2, 4, 20, 24].

Since the auto-regressive approach factorizes the predictions into step-wise conditionals based on previous predictions, these models tend 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 [32] 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 [27, 31], applying noise to the inputs [2, 12, 20, 23], or using adversarial losses [23, 37].

Recent works [6, 26, 38] model the temporal aspects of 3D human motion by encoding every joint's trajectory with the discrete cosine transformation (DCT). Both the observations and the predicted future frames are represented as a set of DCT coefficients which are then used to model inter-joint dependencies. Such an implicit modelling of the temporal information inherently mitigates the failure cases of the auto-regressive models. DCT appears to be an effective non-learning based representation.

In this work, we present a novel architecture for 3D human motion modelling, which attempts to learn a spatio-temporal representation explicitly without relying on the propagation of a hidden state as in RNNs or fixed temporal encodings such as DCT coefficients. Our approach is motivated by the recent success of the Transformer model [35]

[*]Affiliated with Peking University at the time of work.

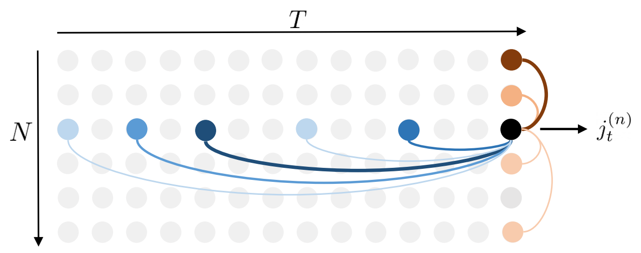


Figure 1: **Spatio-temporal attention.** The motion sequence is a matrix containing N joint configurations $\boldsymbol{j}_t^{(n)}$ over T time steps. For a given joint (black), the *temporal* (blue) and *spatial attention* (red) are illustrated. Color intensity indicates attention weight.

in tasks such as NLP [9, 35], music [17], or images [7, 29]. While the vanilla Transformer is designed for 1D sequences with a self-attention mechanism [3, 28, 35], we note that the task of 3D motion prediction is inherently spatio-temporal and we propose a novel representation that decouples 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* time steps of individual joints. A prediction is then made by summarizing current joint information and previous time steps as a weighted combination.

Our model learns to construct temporally coherent poses from individual joints by considering the temporal and spatial representations learned from the data. The dual self-attention over the sequence allows the model to access past information directly and hence capture the dependencies explicitly [28, 35], mitigating error accumulation over time. It also enables interpretability since attention weights indicate informative sequence parts that led to the prediction. Our experiments show that a naive application of 1D self-attention still suffers from the collapsing pose problem, whereas our proposed model, the ST-Transformer, is able to outperform the state-of-the-art models in short-term horizons and also produce convincing long-term predictions (up to 20 seconds for periodic motions).

2. Related Work

Decomposition of the attention mechanism across different dimensions has been shown to be effective in other domains. In [18], attention is applied separately to the height and width dimensions of images for semantic segmentation. [33] proposes a similar model for skeleton-based action recognition where attention is applied sequentially first to the joints and then to the temporal dimension. [14] introduces axial attention applying self-attention on different dimensions in parallel to reduce computational complexity. Our work is conceptually similar to those but crucially differs in the task domain. In the remainder of this section, we provide a summary of the related work on 3D motion modelling.

Recurrent Models RNNs are the dominant architecture for 3D motion modelling tasks [2, 8, 10–12, 20, 36]. Fragkiadaki *et al.* [11] propose the Encoder-Recurrent-Decoder (ERD) model where an LSTM cell operates in latent space. Jain *et al.* [20] build a skeleton-like st-graph with RNNs as nodes. Aksan *et al.* [2] 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 [25], 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.* [12] more explicitly train a separate de-noising autoencoder that refines the noisy RNN predictions.

Martinez *et al.* [27] introduce a sequence-to-sequence (seq2seq) architecture with a skip connection from the in- to the 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.* [31] suggest to use a teacher-forcing ratio to gradually expose the model to its own predictions. In [8], the seq2seq framework is modified to explicitly model different time scales using a hierarchy of RNNs. Gui *et al.* [37] propose a geodesic loss and adversarial training and Wang *et al.* [36] replace the likelihood objective with a policy gradient method via imitation learning. The acRNN of Zhou *et al.* [39] 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 present 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 [28, 35], 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 Bütepage *et al.* [4, 5] use dense layers on sliding windows of the motion sequences. In [15, 16] convolutional models are introduced for motion synthesis conditioned on trajectories. More recently, Hernandez *et al*. [13] 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*. [23] use CNNs instead of RNNs in the sequence-to-sequence framework to improve long-term dependencies. Similarly, Kaufmann *et al*. [21] propose a convolutional autoencoder for the 3D motion infilling task to fill in large gaps between given sequences.

Implicit Temporal Models Mao *et al*. [26] represent sequences of joints via discrete cosine transform (DCT) coefficients and train a graph convolutional network (GCN) to learn inter-joint dependencies. Since the GCN operates on temporal windows of poses and produces the entire output in one go, the predictions are limited to a pre-determined length. In follow-up work [38], DCT coefficients are instead extracted from shorter sub-sequences in an overlapping sliding window fashion which are then aggregated via a 1D attention block. Similarly, Cai *et al*. [6] leverage a Transformer architecture on the DCT coefficients extracted from the seed sequence and make joint predictions progressively by following the kinematic tree.

Our model is related to these approaches, but differs in three aspects. First, the DCT requires windowed inputs and produces the entire output in one go. This limits full generative modelling of arbitrarily long sequences with sufficient diversity. Instead, we aim to learn spatio-temporal representations directly from the data. Second, we follow a fully auto-regressive approach and model the temporal dependencies explicitly by leveraging the recursive nature of human motion. Third, temporal and spatial modelling is interleaved in our design, whereas in previous work the temporal information is modeled first via the DCT [26] and aggregated with an attention mechanism [6, 38], and then the spatial structure is captured by a GCN or a Transformer. In contrast, our model stacks several computation blocks each of which aggregates temporal and spatial information and passes it to the subsequent layer in a message passing fashion.

In summary, existing 3D motion modelling works have introduced regularization, structural priors, frequency transformations, or auxiliary and adversarial loss terms to address the inherent problems of the underlying architectures. We show that the self-attention concept itself 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-Transfomer) in detail. For an overview please refer to Fig. 2. Our method uses the building blocks of the Transformer [35], 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 $\boldsymbol{X} = \{\boldsymbol{x}_1, \dots, \boldsymbol{x}_T\}$ where a frame $\boldsymbol{x}_t = \{\boldsymbol{j}_t^{(1)}, \dots, \boldsymbol{j}_t^{(N)}\}$ denotes a pose at time step t with joints $\boldsymbol{j}_t^{(n)} \in \mathbb{R}^M$. Each joint $\boldsymbol{j}$ 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 $\boldsymbol{X}$ 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. $\boldsymbol{X}_{Mi:M(i+1),\, t} = \boldsymbol{j}_t^{(i)}$.

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, $\boldsymbol{W}^{(n,I)}$ denotes the weight matrix of the input (I) of joint n. Note that projection matrices $\boldsymbol{W}$ and biases $\boldsymbol{b}$ are trainable. The superscript n 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, i.e. $\boldsymbol{e}_t^{(n)} = \boldsymbol{W}^{(n,E)}\boldsymbol{j}_t^{(n)} + \boldsymbol{b}^{(n,E)}$. Note that the weights $\boldsymbol{W}^{(n,E)} \in \mathbb{R}^{D\times M}$ and bias $\boldsymbol{b}^{(n,E)} \in \mathbb{R}^D$ are per joint n. Following [35], to inject a notion of ordering we add sinusoidal positional encoding to the joint embeddings, reported to be helpful in extrapolating to longer sequences [35]. After dropouts [34], 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, i.e., for each joint $n \in \{1..N\}$, we calculate a temporal summary $\bar{\boldsymbol{E}}^{(n)} = [\bar{\boldsymbol{e}}_1^{(n)}, \dots, \bar{\boldsymbol{e}}_T^{(n)}]^T \in \mathbb{R}^{T\times D}$.

We use the scaled dot-product attention proposed by [35], requiring *query* $\boldsymbol{Q}$, *key* $\boldsymbol{K}$, and *value* $\boldsymbol{V}$ 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* $\boldsymbol{V}$:

$$\mathrm{Attn}(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}, \boldsymbol{M}) = \tau\left(\frac{\boldsymbol{Q}\boldsymbol{K}^T}{\sqrt{D}} + \boldsymbol{M}\right)\boldsymbol{V} = \boldsymbol{A}\boldsymbol{V} \quad (1)$$

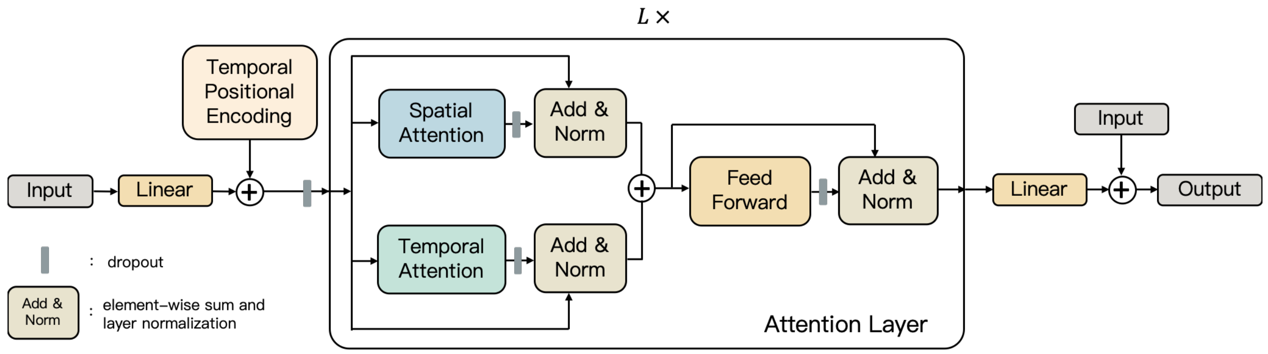


Figure 2: **Architecture overview.** We first project every joint into an embedding space and then inject positional encodings and apply dropout. Next, the embeddings are fed to $L = 8$ stacked attention layers. We employ a novel spatio-temporal multi-head attention mechanism based on [35]. It is split 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 [27].

where the mask $\boldsymbol{M}$ prevents information leakage from future steps and τ is either the softmax function σ or a simple normalization by the sum of all attention scores. For the temporal attention mechanism, we refer to matrix $\boldsymbol{A}$ as $\bar{\boldsymbol{A}} \in \mathbb{R}^{T\times T}$. It contains the temporal attention weights, where each row i in $\bar{\boldsymbol{A}}$ represents how much attention is given to the previous frames in the sequence.

The $\boldsymbol{Q}, \boldsymbol{K}$, and $\boldsymbol{V}$ matrices are projections of the input embeddings $\boldsymbol{E}^{(n)} = [\boldsymbol{e}_1^{(n)}, \ldots, \boldsymbol{e}_T^{(n)}]^T \in \mathbb{R}^{T\times D}$. Following [35], we use a multi-head attention (MHA) mechanism to project the $D-$dimensional representation into subspaces calculated by different attention heads $i \in \{1..H\}$:

$$\begin{aligned} \boldsymbol{Q}^{(n,i)} &= \boldsymbol{E}^{(n)}\boldsymbol{W}^{(n,Q,i)}, \ \ \boldsymbol{Q}^{(n,i)} \in \mathbb{R}^{T\times F} \\ \boldsymbol{K}^{(n,i)} &= \boldsymbol{E}^{(n)}\boldsymbol{W}^{(n,K,i)}, \ \ \boldsymbol{K}^{(n,i)} \in \mathbb{R}^{T\times F} \\ \boldsymbol{V}^{(n,i)} &= \boldsymbol{E}^{(n)}\boldsymbol{W}^{(n,V,i)}, \ \ \boldsymbol{V}^{(n,i)} \in \mathbb{R}^{T\times F} \end{aligned} \tag{2}$$

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 16-dimensional chunks of a 64-dimensional representation. Hence, different attention heads enable accessing different components. The results are then concatenated and projected back into representation space using the weight matrix $\boldsymbol{W}^{(n,O)} \in \mathbb{R}^{HF\times D}$

$$\begin{aligned} \text{head}_i &= \text{Attn}\left(\boldsymbol{Q}^{(n,i)}, \boldsymbol{K}^{(n,i)}, \boldsymbol{V}^{(n,i)}, \boldsymbol{M}\right) \\ \bar{\boldsymbol{E}}^{(n)} &= \text{Concat}\left(\text{head}_1, \ldots, \text{head}_H\right)\boldsymbol{W}^{(n,O)}. \end{aligned} \tag{3}$$

Spatial Attention In the vanilla Transformer, the attention block operates on the entire input vector $\boldsymbol{x}_t$ and 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 $\boldsymbol{E}_t = [\boldsymbol{e}_t^{(1)}, \ldots, \boldsymbol{e}_t^{(N)}]^T \in \mathbb{R}^{N\times D}$, the spatial summary of joints $\tilde{\boldsymbol{E}}_t^{(n)}$ is calculated as a function of all other joints by using the multi-head attention:

$$\begin{aligned} \boldsymbol{Q}_t^{(i)} &= \left[\left(\boldsymbol{W}^{(1,Q,i)}\right)^T \boldsymbol{e}_t^{(1)}, \ldots, \left(\boldsymbol{W}^{(N,Q,i)}\right)^T \boldsymbol{e}_t^{(N)}\right]^T \\ \boldsymbol{K}_t^{(i)} &= \boldsymbol{E}_t\boldsymbol{W}^{(K,i)}, \quad \boldsymbol{V}_t^{(i)} = \boldsymbol{E}_t\boldsymbol{W}^{(V,i)} \\ \text{head}_i &= \text{Attn}(\boldsymbol{Q}_t^{(i)}, \boldsymbol{K}_t^{(i)}, \boldsymbol{V}_t^{(i)}, \boldsymbol{0}) = \tilde{\boldsymbol{A}}\boldsymbol{V}_t^{(i)} \\ \tilde{\boldsymbol{E}}_t^{(n)} &= \text{Concat}\left(\text{head}_1, \ldots, \text{head}_H\right)\boldsymbol{W}^{(O)} \end{aligned} \tag{4}$$

where $\boldsymbol{Q}_t^{(i)} \in \mathbb{R}^{N\times S}$, $\boldsymbol{K}_t^{(i)} \in \mathbb{R}^{N\times S}$, $\boldsymbol{V}_t^{(i)} \in \mathbb{R}^{N\times S}$, $S = F = D/H$, $\text{head}_i \in \mathbb{R}^{N\times S}$, and $\boldsymbol{W}^{(O)} \in \mathbb{R}^{HS\times D}$. The spatial attention $\tilde{\boldsymbol{A}} \in \mathbb{R}^{N\times N}$ 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 $\bar{\boldsymbol{E}}$ and $\tilde{\boldsymbol{E}}$, respectively. They are summed and passed to a 2-layer pointwise feed-forward network [35], which is followed by a dropout and layer normalization. We stack $L = 8$ such attention layers to successively update the joint embeddings and thus to refine the pose predictions.

Joint Predictions Finally, the joint prediction $\hat{\boldsymbol{j}}_{t+1}^{(n)}$ is obtained by projecting the corresponding D-dimensional embedding $\boldsymbol{e}_t^{(n)}$ from the L-th attention layer back to the M-

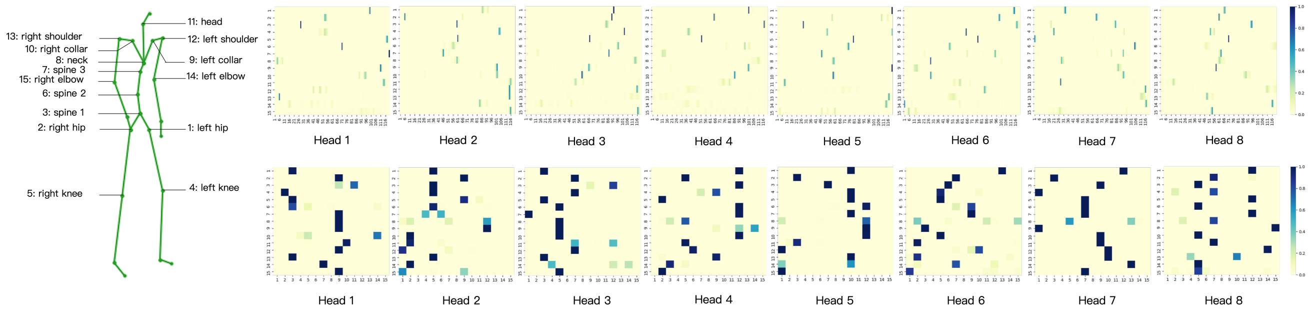


Figure 3: Temporal (*top*) and spatial (*bottom*) attention weights of the first layer given 120 frames. Each row corresponds to a joint's attention pattern. The columns represent time steps (*top*) or joints (*bottom*). We visualize 8 attention heads showing that they attend to different joints and time steps. Similarly, different joints exhibit different spatial and temporal attention patterns.

dimensional joint angle space. Like [27], 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 ℓ_2 distance on rotation matrices directly [2]:

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

At test time, we compute the prediction in an auto-regressive manner. That is, given a pose sequence $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T\}$, we get the prediction $\hat{\boldsymbol{x}}_{T+1}$. 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 $\hat{\boldsymbol{x}}_{T+2}$ we condition on the sequence $\{\boldsymbol{x}_2, \ldots, \hat{\boldsymbol{x}}_{T+1}\}$.

4. Experiments

We evaluate the ST-Transformer on AMASS [25] and H3.6M [19] in Sec. 4.1 following the standard protocols for short-term predictions and adopting distribution-based metrics for long-term predictions. We note that both benchmarks focus on modeling 3D joint angles unlike the 3D position-based benchmarks presented in the previous work [36, 38]. We argue that modeling the 3D position representation of human pose is prone to errors as the models are free to violate the skeletal configuration. In other words, the outputs may contain artifacts such as inconsistent bone lengths across the frames [15, 21]. In contrast, the joint angle representation implicitly preserves the skeletal structure which is important in many downstream tasks.

Sec. 4.2 and Sec. 4.4 show qualitative results and attention weights, thus providing insights into how the model forms predictions. We validate design choices through ablation studies in Sec. 4.3. We run our models and the baselines on various joint angle representations including rotation matrix, quaternion and angle-axis, and report the best performance. Implementation details for our model and the baselines are provided in the supplementary material.

4.1. Quantitative Evaluation

AMASS We follow Aksan *et al.* [2] and evaluate our model on the large-scale motion dataset AMASS [25]. Table 1 summarizes the results with pairwise angle- and position metrics up to 400 ms. For longer time horizons, direct comparison to the ground-truth via MSE becomes increasingly problematic particularly for auto-regressive models [11]. Hence, in addition to the standard metrics in [2, 27], we conduct further analysis by using complementary metrics in the frequency domain allowing benchmarking up to 15 seconds (Fig. 4).

In the short-term evaluations, we compare our ST-Transformer with the vanilla Transformer, previously reported RNN-based architectures and two DCT-based architectures [26, 38]. We could not compare to [6] as no implementation is publicly available. The vanilla Transformer follows an auto-regressive approach similar to our ST-Transformer but applies 1D attention on the pose vectors. Furthermore, we include results of a variant of our model that does not use softmax σ in the attention (cf. Eq. 1) as the softmax may lead to gradient instabilities. Instead, we normalize by the sum of all attention scores similar to [38]. Our ST-Transformers achieve state-of-the-art in all metrics while LTD-Attention [38] remains competitive at 400 ms.

Long-Term Our approach is fully generative, so while it maintains local consistency, the global positions may deviate from the ground-truth under natural variation. Hence, we propose to use distribution-based metrics in the power spectrum (PS) space proposed by Hernandez *et al.* [13] instead of a direct comparison with the ground-truth frames. We report two metrics, (i) *PS KLD* which measures the discrepancy between the prediction and the test distributions via the KL

Table 1: **AMASS results**. Lower is better for *Euler*, *Joint Angle* and *Positional* metrics. For the *Area Under the Curve (AUC)*, higher is better. Column-wise best result in **bold**, second best underlined. In contrast to [2] we report the mean over the sequence, rather than the sum, and the positional metric was converted from meters to millimeters. ST-Transformer stands for our Spatio-temporal Transformer model. * indicates our own evaluation of the respective model after hyper-parameter tuning.

	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 [2, 27]	0.318	0.494	0.631	0.741	0.061	0.102	0.136	0.164	23.6	39.6	53.1	64.2	86.4	83.3	84.5	81.8
Seq2seq [2, 27]	0.336	0.499	0.623	0.722	0.062	0.097	0.126	0.150	23.6	37.4	48.6	57.9	86.5	83.8	85.4	82.9
QuaterNet [2, 31]	0.248	0.391	0.509	0.606	0.044	0.074	0.102	0.125	16.7	28.7	39.6	49.0	90.4	87.4	88.0	85.4
RNN-SPL [2]	0.221	0.344	0.446	0.535	0.037	0.061	0.084	0.105	13.6	23.0	31.9	40.0	92.5	89.9	90.3	87.9
LTD-10-10* [26]	0.205	0.333	0.447	0.544	0.039	0.065	0.089	0.111	15.6	25.7	35.3	44.1	91.5	88.8	89.3	86.9
LTD-Attention* [38]	0.207	0.321	0.418	0.499	0.035	0.060	0.083	0.102	13.4	23.2	32.1	39.7	92.5	89.8	90.2	87.9
Transformer	0.216	0.332	0.433	0.523	0.036	0.060	0.083	0.104	13.3	22.8	31.8	40.0	92.5	89.9	90.2	87.8
ST-Transformer	**0.178**	**0.291**	**0.395**	**0.490**	0.033	0.057	0.081	0.103	13.2	22.6	32.0	40.9	93.1	90.2	90.3	87.7
ST-Transformer w/o σ	0.187	0.302	0.409	0.504	**0.032**	**0.056**	**0.079**	**0.101**	**12.8**	**21.8**	**30.9**	**39.5**	**93.5**	**90.6**	**90.7**	**88.2**

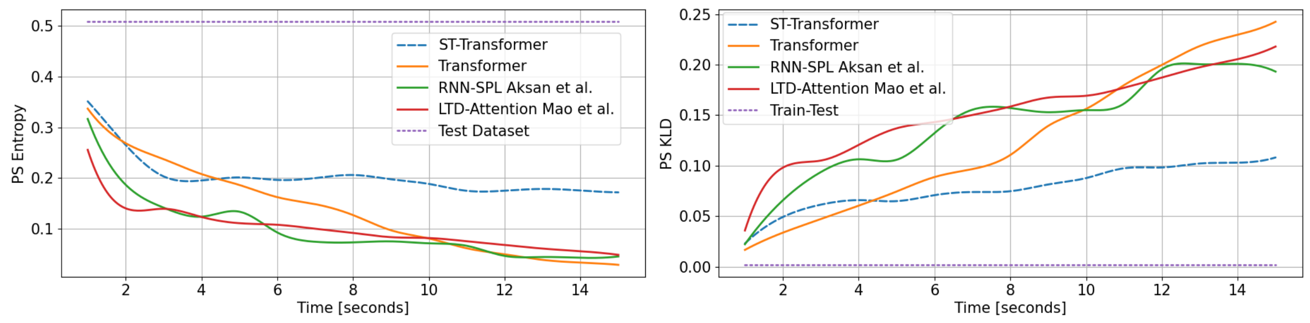


Figure 4: **Power Spectrum (PS) metrics.** *(left)* PS Entropy, higher is better. Our model (blue, dashed) exhibits higher entropy suffering less from the static pose problem in longer predictions. *(right)* PS KLD, lower is better. Our model's prediction distribution stays closer to the data distribution as indicated by the symmetric KLD.

divergence, and (ii) *PS Entropy* capturing the entropy of the prediction distribution in the power spectrum. The latter measures how diverse the predictions are. Models collapsing to static pose predictions end up with lower entropy values. However, *PS Entropy* can be deceived by random predictions and it should thus be interpreted in conjunction with the *PS KLD* where only a set of predictions that are similar to the real data samples can achieve a lower score.

In Figure 4, we compare our ST-Transformer, the vanilla Transformer, RNN-SPL [2] and LTD-Attention with DCT representations [38] on these PS metrics for predictions up to 15 seconds. To produce long sequences with LTD-Attention, we run it autoregressively on its own predictions and in a sliding window fashion. The reference values (dotted, purple) from the training and test samples are calculated on randomly extracted 1-second windows (60 frames). Similarly, we compute statistics over the predictions of the respective model 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 any of the baselines. This indicates that our model does generate plausible poses that are similar to the data distribution without memorizing the exact sequences.

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 none of the baselines can alleviate this problem as much as the ST-Transformer. The difference is more pronounced with horizon length. These observations are additionally corroborated by our visualizations in the supplementary video and Fig. 6 where we clearly see that the walking motion produced by the baselines phase out earlier compared to our ST-Transformer's output.

H3.6M Traditionally, motion prediction has been benchmarked on H3.6M [19]. Tab. 2 compares our model in this setting, where we are competitive and often achieve state-of-the-art. H3.6M is roughly 14 times smaller than AMASS and its test split consists only of a few sequences, which has been reported to cause high variance [2, 30]. Furthermore, as is evident from Tab. 2, improvements are often marginal and recent works seem to converge to the same error for most actions. For these reasons we argue that the AMASS benchmark introduced by [2] carries more weight.

Discussion It is evident that the spatio-temporal decoupling of attention is indeed beneficial when compared to the vanilla

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 and surpassing or performing on par with state-of-the-art.

	Walking				Eating				Smoking				Discussion			
milliseconds	80	160	320	400	80	160	320	400	80	160	320	400	80	160	320	400
RNN-SPL [2]	0.26	0.40	0.67	0.78	0.21	0.34	0.55	0.69	0.26	0.48	0.96	0.94	0.30	0.66	0.95	1.05
AGED [37]	0.22	0.36	0.55	0.67	0.17	0.28	0.51	0.64	0.27	0.43	**0.82**	0.84	0.27	0.56	0.76	0.83
LTD-10-10 [26]	0.18	0.31	0.49	**0.56**	0.16	0.29	0.50	0.62	0.22	0.41	0.86	0.80	0.20	0.51	0.77	0.85
LTD-Attention [38]	0.18	**0.30**	**0.46**	0.51	0.16	0.29	0.49	0.60	0.22	0.42	0.86	0.80	0.20	0.52	0.78	0.87
Propagation [6]	**0.17**	**0.30**	0.51	0.55	0.16	0.29	0.50	0.61	**0.21**	**0.40**	0.85	**0.78**	0.22	**0.39**	**0.62**	**0.69**
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	0.19	0.33	0.56	0.64	**0.15**	**0.27**	**0.45**	**0.57**	0.22	0.41	0.87	0.82	**0.19**	0.53	0.81	0.93

Transformer. In all settings our ST-Transformer significantly outperforms the vanilla counterpart. Compared to the RNN-based autoregressive baselines such as RNN-SPL, Seq2seq or AGED, our model makes more accurate predictions in short-term horizon as well as plausible longer-term generations by mitigating the error accumulation problem.

The DCT-based baselines are the most competitive and also conceptually more relevant to our work. We argue that the task favors the DCT-based representations as a temporal window is encoded and decoded in one go. In other words, the entire prediction horizon is predicted at once in contrast to our frame-by-frame predictions. Hence, we observe that our model shows strong performance in the shortest prediction horizon with respect to the *pairwise comparisons with the ground-truth* on both AMASS and H3.6M (cf. Tab. 1, 2). Our model's error with respect to the ground-truths seems to increase with longer prediction horizons, which is expected for an auto-regressive model due to error accumulation. Yet, it remains very competitive and the distribution-based metrics in Figure 4 also highlight that our model is statistically closer to the real data in very long-term predictions.

4.2. Qualitative Evaluation

Here, we evaluate the generative capabilities up to 20 seconds. We feed the model with a particular motion sequence of 2 seconds and auto-regressively predict beyond its training horizon (i.e., 400 ms or 1 sec).

We qualitatively compare our model with the vanilla Transformer, RNN-SPL [2] and LTD-Attention [38] on a walking sample from AMASS in Fig. 6. With RNN-SPL any variation quickly disappears within 5 seconds. The vanilla Transformer does not reach a static pose for longer, which shows the benefits of the attention mechanism over recurrent networks. However, it still collapses around second 15, whereas our ST-Transformer maintains the walking motion over the entire duration of 20 seconds. Also the LTD-Attention model produces walking motion longer than RNN-SPL, but still converges before 10 seconds. 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 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-second windows.

4.3. Ablations

2D Attention To unpack our contribution more clearly we implement and compare to a 2D transformer architecture. Here, every joint attends to all other joints across all frames (Fig. 5). The complexity of the attention thus becomes $O(N \times T)$. With this design, memory requirements increase drastically. Hence, for training we either reduce the model complexity or decrease T or the batch size. The performance of the best configuration we found is summarized in Tab. 6. It clearly falls behind the ST-Transformer. Our approach reduces the complexity from $O(N \times T)$ to $O(N + T)$, allowing to attend to a longer history and to use a larger model, highlighting our contribution on an architectural level.

Number of Layers We train our model with varying number of attention layers L. Fig. 7 shows that competitive performance on AMASS is reached with only 3 layers. As the number of layer increases, the representations learned by our

Table 3: **2D Attention Ablation** on AMASS.

	Euler ↓		Joint Angle ↓		Positional ↓	
milliseconds	100	400	100	400	100	400
2D-Transformer	0.213	0.555	0.037	0.115	14.5	45.2
ST-Transformer	**0.178**	**0.490**	**0.033**	**0.103**	**13.2**	**39.5**

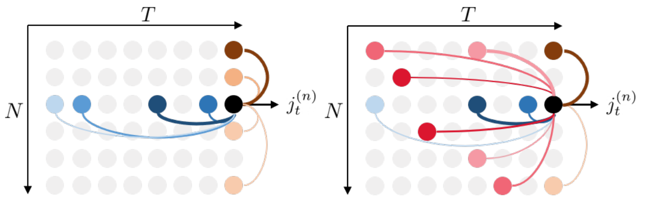


Figure 5: **Spatio-temporal attention vs. 2D attention.** All the joints except the future ones are available in the naive 2D attention case as highlighted in red-ish colors.

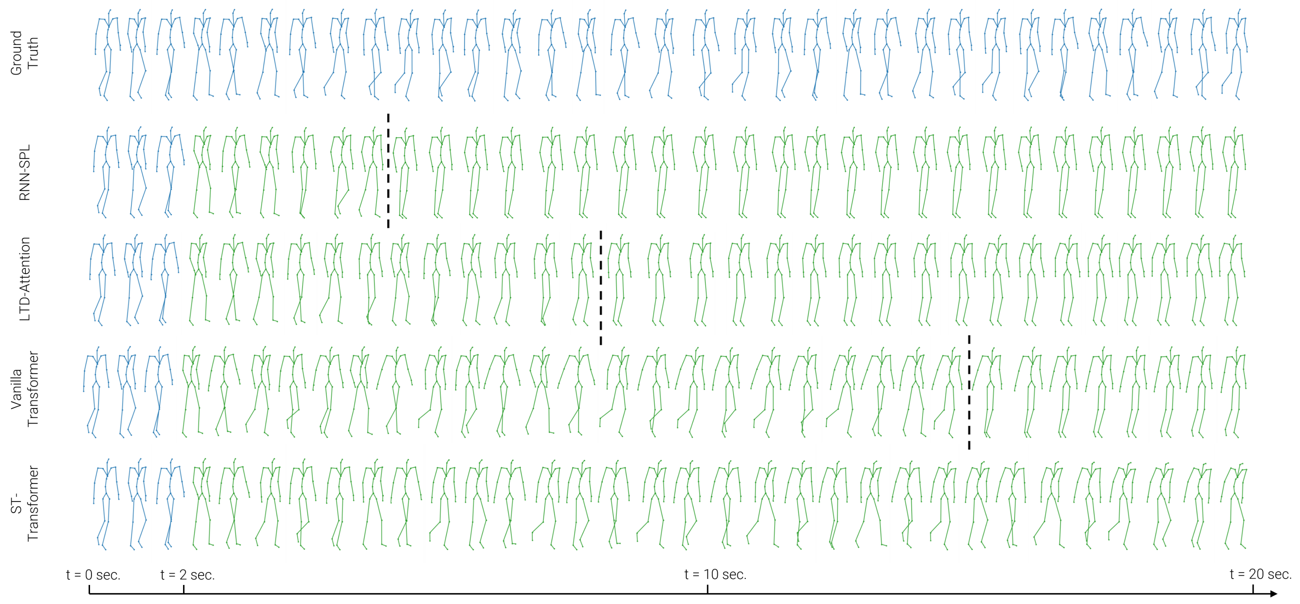


Figure 6: **Qualitative Results.** A walking motion predicted up to 20 seconds by RNN-SPL [2], LTD-Attention [38], vanilla Transformer, and our ST-Transformer. Seed length is 2 sec. The dashed lines indicate when the motion comes to a halt.

model are tuned better. It can be considered as the number of message passing steps to update the available representation.

4.4. 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 $\bar{\boldsymbol{A}}$, $\tilde{\boldsymbol{A}}$ 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, please refer to Fig. 7 in the appendix and the video.

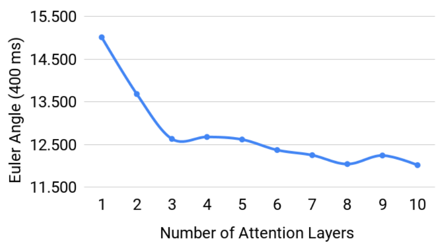


Figure 7: **Ablation** on value of L on AMASS.

5. Conclusion

We introduce a novel spatio-temporal transformer (ST-Transformer) network for generative modeling of 3D human motion. Our proposed architecture learns intra- and inter-joint dependencies explicitly via its decoupled temporal and spatial attention blocks. We show that the self-attention concept can be very effective in learning representations for both short- and long-term motion predictions compared to the DCT-based motion representations. Furthermore, it mitigates the long-term dependency issue observed in auto-regressive architectures and is able to synthesize motion sequences up to 20 seconds conditioned on periodic motion types such as locomotion.

Acknowledgments 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.

References

[1] 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. Software available from tensorflow.org.

[2] 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.

[3] 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.

[4] Judith Bütepage, 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.

[5] Judith Bütepage, 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.

[6] Yujun Cai, Lin Huang, Yiwei Wang, Tat-Jen Cham, Jianfei Cai, Junsong Yuan, Jun Liu, Xu Yang, Yiheng Zhu, Xiaohui Shen, et al. Learning progressive joint propagation for human motion prediction. In *European Conference on Computer Vision*, pages 226–242. Springer, 2020.

[7] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. *arXiv preprint arXiv:1904.10509*, 2019.

[8] 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.

[9] 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.

[10] Xiaoxiao Du, Ram Vasudevan, and Matthew Johnson-Roberson. Bio-lstm: A biomechanically inspired recurrent neural network for 3d pedestrian pose and gait prediction. *IEEE Robotics and Automation Letters (RA-L)*, 2019. accepted.

[11] 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.

[12] 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.

[13] 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.

[14] Jonathan Ho, Nal Kalchbrenner, Dirk Weissenborn, and Tim Salimans. Axial attention in multidimensional transformers. *arXiv preprint arXiv:1912.12180*, 2019.

[15] 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.

[16] 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.

[17] 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.

[18] Zilong Huang, Xinggang Wang, Lichao Huang, Chang Huang, Yunchao Wei, and Wenyu Liu. Ccnet: Criss-cross attention for semantic segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 603–612, 2019.

[19] 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.

[20] 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.

[21] Manuel Kaufmann, Emre Aksan, Jie Song, Fabrizio Pece, Remo Ziegler, and Otmar Hilliges. Convolutional autoencoders for human motion infilling. *arXiv preprint arXiv:2010.11531*, 2020.

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

[23] 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.

[24] 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.

[25] 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.

[26] 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.

[27] 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.

[28] Ankur P Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. A decomposable attention model for natural language inference. *arXiv preprint arXiv:1606.01933*, 2016.

[29] Niki J. Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Noam Shazeer, and Alexander Ku. Image transformer. 2018.

[30] Dario Pavllo, Christoph Feichtenhofer, Michael Auli, and David Grangier. Modeling human motion with quaternion-based neural networks. *CoRR*, abs/1901.07677, 2019.

[31] Dario Pavllo, 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.

[32] Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. Sequence level training with recurrent neural networks. *arXiv preprint arXiv:1511.06732*, 2015.

[33] Lei Shi, Yifan Zhang, Jian Cheng, and Hanqing Lu. Decoupled spatial-temporal attention network for skeleton-based action recognition. *arXiv preprint arXiv:2007.03263*, 2020.

[34] 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.

[35] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz 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.

[36] 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.

[37] 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.

[38] Mao Wei, Liu Miaomiao, and Salzemann Mathieu. History repeats itself: Human motion prediction via motion attention. In *ECCV*, 2020.

[39] 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.

Supplementary Material for

A Spatio-temporal Transformer for 3D Human Motion Prediction

The Supplementary Material includes this document and a video. We provide additional details on the implementation of the proposed ST-Transformer and further experimental evidence of its performance. We explain experimental details in Sec. A and details of the multi-head attention mechanism in Sec. B. We visualize more attention weights in Sec. C and Sec. D provides additional ablations. In Sec. E we provide more insights into how our ST-attention compares to naive 2D attention. Sec. F and Sec. G detail how we evaluated the LTD [26] and LTD-Attention [38] models on the AMASS dataset, respectively. Finally, in Sec. H we provide definitions of the Power Spectrum metrics.

A. Experimental Details

We implemented our models in TensorFlow [1]. Hyper-parameters we used for the experiments are listed in Tab. 4. Due to the limited data size of H3.6M, we achieved better results by using a smaller network as larger models usually suffered from overfitting.

As suggested by Vaswani *et al.* [35], the Transformer architecture is sensitive to the learning rate. We apply the same learning rate schedule as proposed in [35]. The learning rate is calculated as a function of the training step as follows:

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

where D is the joint embedding size. Warmup is set to 10000 for our AMASS and H3.6M datasets. We use a batch size of 32 and the Adam optimizer [22] with its default parameters. Before updating the parameters, gradients are clipped with respect to the global gradient norm (i.e., clip by global norm) with a maximum norm value of 1.0.

Following the training protocol in [2], 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.

Data represenations 3D joint angles can be represented with various representations such as angle-axis [27], quaternion [31] or rotation matrix [2]. In our experiments both on the AMASS and H3.6M datasets, we trained and evaluated our model on all three joint angle representations. Similarly, we also evaluated the baseline models LTD and LTD-Attention (cf. Sec. F and Sec. G) by using all the joint angle representations and reported the best performance.

We found that all models achieved their best performances with the rotation matrix representation (see Tab. 5 for our model's results). Since the model's predictions might not be valid rotation matrices, we project them to the nearest valid rotation matrix in SO(3) using the singular value decomposition. The evaluation metrics are then computed on the projected rotation matrices.

Weight sharing In our initial design both temporal and spatial attention were alike (i.e., following Eq. (3)). We experimentally found that our current design, i.e., sharing the *key* and *value* weights across joints but keeping the *query* separate, achieves better performance. We keep the temporal

Table 4: H3.6M and AMASS experiment hyper-parameters.

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

Table 5: Our models performance when trained with different data representations. We report only the Euler error performance at 400ms on the AMASS dataset.

Representation	Euler Error
Quaternion	0.543
6D Rotations	0.527
Angle-axis	0.522
Rotation Matrix	0.490

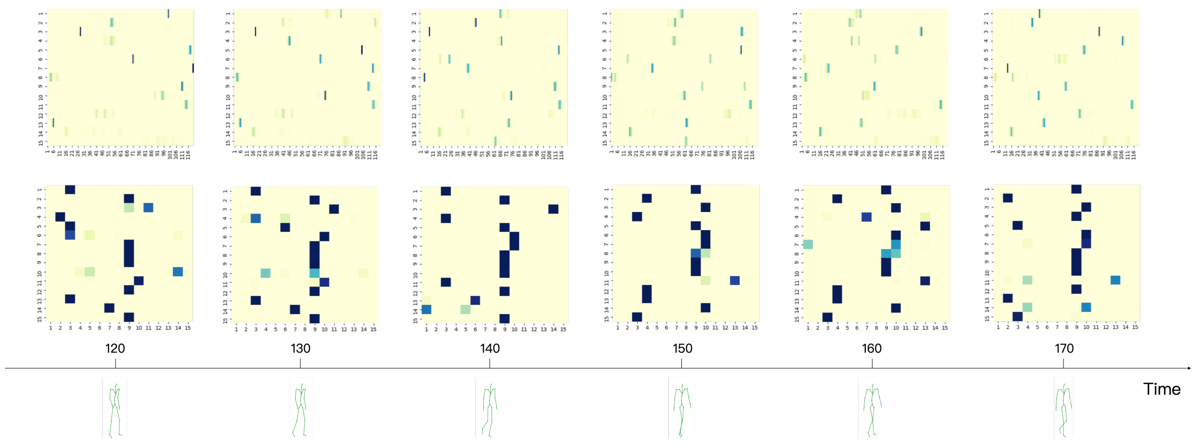


Figure 8: Temporal (*top*) and spatial (*middle*) attention weights aligned with the poses (*bottom*) over 1 second prediction. We visualize a single head in the first layer. Our model identifies the informative joints dynamically. Note that the temporal attention is calculated over a sliding window. 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).

attention as is and compare our current design (i.e., Eq. (5)) with two alternatives: projection weights W^Q, W^K, W^V are (A) separate (i.e., joint-wise as in Eq. (3)) or (B) shared across joints. On AMASS at 400ms, (A) achieves 0.511 Euler error and (B) 0.504 (vs 0.490 with the current design).

Data augmentation Finally, we observed a benefit of data augmentation on the H3.6M dataset. With a random chance of 0.5, we reverted or mirrored a sample sequence. Mirroring the joints was previously reported to be useful in [31]. For the former one, we hypothesize a reverted sequence still posses valid human poses and backward motion dynamics, which may help the model to minimize the null space.

B. Multi-head Attention

Fig. 9 illustrates the multi-head self-attention mechanism mentioned in Eq. (3) and (4) of the main paper. The subspaces of query, key, and value for each attention head are calculated from the joint embeddings $\boldsymbol{e}_t^{(n)}$. From this figure

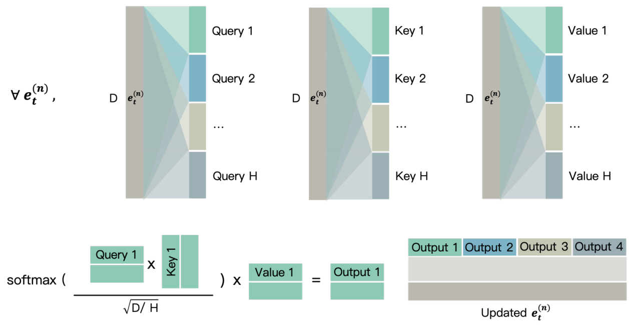


Figure 9: The multi-head self-attention mechanism. Each color represents one head.

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 ℓ 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 $\bar{\boldsymbol{E}}^{(n)}$.

C. 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. 8, 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.

D. Hyper-parameters

We experiment with varying number of attention heads H. As shown in Fig. 10a, 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.

Fig. 10b 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. 8).

We train our model with varying number of layers. Fig. 10c shows that reasonable performance on AMASS is reached with only 3 layers. However, as the number of layer increases, the representations learned by our model is tuned better. It can also be considered as the number of message passing steps to update the available representation.

E. Additional Ablation on 2D Attention

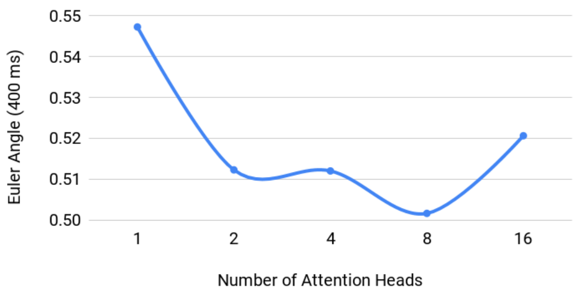
To provide more insights into the computational efficiency of our ST-attention compared to the naive 2D attention discussed in Sec. 4.3, we present several additional comparisons in Tab. 6.

In rows 1-3 of Tab. 6, we maximize one of the three hyper-parameters for the 2D attention where row 1 exhausts the batch size and row 2 the window size at the cost of the batch size. We observe that prioritizing one of the three parameters is usually detrimental for the performance of the 2D attention model and we get the best result with a trade-off between the three (cf. rows 4, 6). Furthermore, we also observe that those best 2D configurations are always outperformed when switching to our decoupled ST attention (cf. rows 5, 7). Row 8 reports the best configuration we found with our ST-attention. These results show that our decoupled attention mechanism is superior to the plain 2D attention even when controlling for computational efficiency.

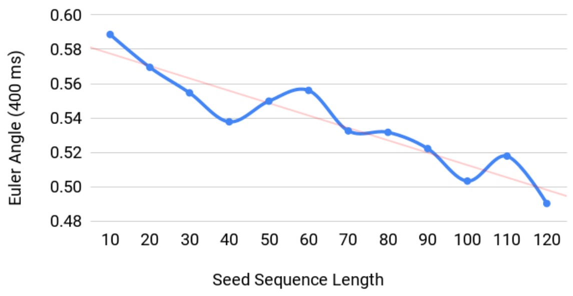
F. Evaluation of LTD on AMASS

In Tab. 1 of the main paper we report the performance of the LTD model [26] (cf. LTD-10-10 entry) on the AMASS dataset. The results correspond to the best results we obtained after hyper-parameter-tuning, explained in more detail in the following.

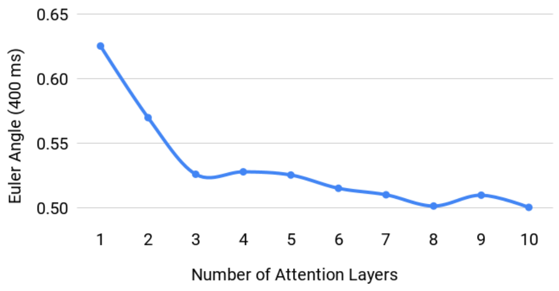
To train and evaluate LTD on AMASS we use the code



(a) Different number of attention heads (H).



(b) Seed sequences of different lengths.



(c) Different number of attention layers (L).

Figure 10: Ablations on the AMASS dataset. We evaluate our model with varying configurations and report Euler angle loss at 400 ms.

provided by Mao *et al*. [26] but swap out the data pipeline to load AMASS instead of H3.6M. In [26] the inputs to the network and the outputs are 400 milliseconds (10 frames) worth of data. As AMASS is sample at 60 Hz, we hence pass 24 frames as input and let the model predict 24 frames.

We fine-tune the learning rate as well as the number of DCT coefficients. For the number of DCT coefficients we use the original 35, but also the maximum number of coefficients 48. As reported by [26] we find this to make little difference, but 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 [22] optimizer with a learning rate of 0.001 and batch size of 16. We train for maximum 100 epochs with

		Euler ↓		Joint Angle ↓		Positional ↓	
ID	Milliseconds →	100	400	100	400	100	400
1	[2D] L4 - W80 - B32	0.218	0.583	0.038	0.121	14.4	46.9
2	[2D] L8 - W120 - B5	0.256	0.627	0.044	0.129	16.5	50.2
3	[2D] L8 - W60 - B16	0.238	0.602	0.041	0.123	15.7	48.6
4	[2D] L4 - W100 - B16	0.213	0.555	0.037	0.115	14.5	45.2
5	[ST] L4 - W100 - B16	0.201	0.538	0.035	0.111	13.1	42.6
6	[2D] L8 - W40 - B32	0.212	0.575	0.036	0.119	13.6	45.5
7	[ST] L8 - W40 - B32	0.204	0.538	0.036	0.111	14.2	43.6
8	**[ST] L8 - W120 - B32**	**0.178**	**0.490**	**0.033**	**0.103**	**12.8**	**39.5**

Table 6: 2D attention ablation. ST refers to our decoupled attention. Each row corresponds to a different configuration denoted by **L** (number of stacked attention layers), **W** (number of context frames, i.e., temporal attention length) and **B** (batch size).

early stopping. The learning rate is decayed by a factor of 0.96 every other epoch. The input window size is 48. The model parameters are kept as proposed in the original paper resulting in a model size of roughly 2.23 Mio. parameters.

G. Evaluation of LTD-Attention on AMASS

To evaluate the LTD-Attention [38] model on AMASS, we again use the publicly available code and plug in our AMASS data loading pipeline. We adjust the number of input and output frames to reflect the different framerate on AMASS, i.e. we feed seeds of length 120 (2 seconds) and predict 60 frames (1 second). We have found that this resulted in better performance than predicting 24 frames (400 milliseconds) directly.

We keep the Adam optimizer with the originally proposed learning rate decay, but fine-tuned the learning rate to 0.005 with a batch size of 128. Similarly, we found 45 DCT coefficients to work best and we kept the kernel size at the original value of 10. The model is trained for 100 epochs and all other model parameters are kept the same with the exception of the size of the inputs and outputs. Like for our model, we also tried out different joint angle representations, of which rotation matrices performed best.

H. Power Spectrum Metrics

On AMASS, we 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.* [13].

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 on arbitrary rotation representations. Given a sequence $\boldsymbol{X}$, we treat every coordinate of every joint over time as a feature sequence $\boldsymbol{x}_f$ following [13]. The power spectrum PS is then equal to $PS(\boldsymbol{x}_f) = ||FFT(\boldsymbol{x}_f)||^2$ where FFT denotes the Fast Fourier Transform.

PS Entropy It is defined as

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

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

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

We use publicly available implementations of the metrics ([13], 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.