KEY-FRAME BASED MOTION REPRESENTATIONS FOR POSE SEQUENCES

HARRISH PATRICK THASARATHAN

A THESIS SUBMITTED TO
THE FACULTY OF GRADUATE STUDIES
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

GRADUATE PROGRAM IN COMPUTER SCIENCE
YORK UNIVERSITY
TORONTO, ONTARIO

DECEMBER, 2023

Abstract

Modelling human motion is critical for computer vision tasks that aim to perceive human behaviour. Extending current learning-based approaches to successfully model long-term motions remains a challenge. Recent works rely on autoregressive methods, in which motions are modelled sequentially. These methods tend to accumulate errors, and when applied to typical motion modelling tasks, are limited up to only four seconds. We present a non-autoregressive framework to represent motion sequences as a set of learned key-frames without explicit supervision. We explore continuous and discrete generative frameworks for this task and design a key-framing transformer architecture to distill a motion sequence into key-frames and their relative placements in time. We validate our learned key-frame placement approach with a naive uniform placement strategy and further compare key-frame distillation using our transformer architecture with an alternative common sequence modelling approach. We demonstrate the effectiveness of our method by reconstructing motions up to 12 seconds.

Acknowledgements

I would like to express my gratitude to my supervisors, Kosta Derpanis and Marcus Brubaker, for their support and guidance throughout my Master's journey. Under their mentorship, I've learned that research transcends SOTA results and bold numbers; it's about posing insightful questions and being principled in the way we approach them. Their passion and excitement have left me consistently reassured that I am on the right path, and I am truly grateful for their influence. I would also like to give my sincere thanks to my collaborator and friend Saeed Ghorbani. His generous contributions from his extensive expertise in motion modelling have been invaluable.

Additionally, I extend my heartfelt thanks to my fellow members of the CVIL lab. You have not only been fantastic role models and collaborators but, most importantly, wonderful new friends. I have learned so much from each of you and I look forward to the next (give or take) four years of learning and growing alongside you.

I consider myself incredibly fortunate to be surrounded by friends and family who have supported every decision I've made. I couldn't ask for more. Thank you all for everything.

Contents

Abstract ii

Acknowledgements iii

Contents iv

List of Tables vi

List of Figures vii

1 Introduction 1

2 Background 6

2.1 Generative methods 6

2.1.1 Variational autoencoders 7

2.1.2 Vector quantized variational autoencoders 8

2.2 Sequence modelling with deep neural networks 10

2.2.1 Recurrent neural networks 11

2.2.2 Convolutional neural networks 12

2.2.3 Transformers 14

2.3 Related work 19

2.3.1 Unimodal motion forecasting 19

2.3.2 Multi-modal motion forecasting 20

2.3.3 Long-term motion forecasting 21

2.3.4 Human motion synthesis 22
2.4 Summary 23

3 Technical Approach **24**
3.1 Latent key-frames 24
3.1.1 Latent pose quantization 26
3.2 Irregular key-framing 28
3.3 Temporal subsampling with transformers 31
3.4 Summary 33

4 Experimental Evaluation **35**
4.1 Baselines 35
4.2 Implementation and training 36
4.3 Datasets and evaluation 37
4.4 Metrics 39
4.5 Quantitative results 41
4.6 Qualitative results 46
4.7 Discussion 52
4.8 Summary 56

5 Conclusion and Future Work **62**
5.1 Broader impact 64
5.2 Future work 65

Bibliography **67**

Appendix **76**
A.1 Additional implementation details 76
A.2 Entropy regularization 77
A.3 Additional samples 79

List of Tables

4.1 Quantitative comparison of baselines on four second motions 42
4.2 Average Displacement Error on the MotionSynth dataset 57
4.3 Mean Per Joint Position Error on the MotionSynth dataset 58
4.4 Average Displacement Error on the LAFAN1 dataset 59
4.5 Mean Per Joint Position Error on the LAFAN1 dataset 60

A.1 Entropy regularization ablation 79

List of Figures

2.1 Unrolled RNN cell 12
2.2 Example 1D CNN architecture for sequence modelling 13
2.3 Multi-head Attention 15
2.4 Transformer block 17
2.5 Cross-attention 18

3.1 Voronoi 27
3.2 Key-framing Architecture 34

4.1 Quantitative metrics comparison on MotionSynth 30Hz 43
4.2 Quantitative metrics comparison on MotionSynth 120 Hz 44
4.3 Quantitative metrics comparison on LAFAN1 45
4.4 Velocity histogram comparison for a four second motion 47
4.5 Angular velocity distribution comparison for a four second motion 48
4.6 KL-Divergence between velocity distributions on MotionSynth with VQ . 49
4.7 KL-Divergence between velocity distributions on LAFAN1 50
4.8 Motion sample of learned versus uniform spacing 50
4.9 12s motion sample at four kf/s 51
4.10 Heel and hand joint velocity out of phase 52
4.11 Acceleration plot of heel and hand joints 53
4.12 Box-plot of learned key-frame spacings 55
4.13 Velocity reconstruction at each key-frame sample rate 55

A.1 Box-plot of learned key-frame spacings without entropy regularization . . 78
A.2 MotionSynth 30 Hz reconstructed joints with continuous key-framing transformer 80
A.3 MotionSynth 30 Hz reconstructed joints with VQ-based key-framing transformer 81
A.4 MotionSynth 120 Hz reconstructed joints with VQ key-framing transformer 82
A.5 LAFAN1 reconstructed joints with continuous key-framing transformer . 83
A.6 LAFAN1 reconstructed joints with VQ-based key-framing transformer . . 84
A.7 Heel and hand joint velocity out of phase 85
A.8 Heel and hand joint acceleration example 85
A.9 Heel and hand joint velocity example 86
A.10 Heel and hand joint acceleration example 86
A.11 12s motion sample reconstructed with learned and uniform spacing 87
A.12 12s motion sample at six kf/s 87

Chapter 1

Introduction

Similar to most mammals, human movement gives us agency in our environments. The complexity of human motion has allowed our species to create comprehensive systems of nonverbal communication [46, 4] (e.g., American Sign Language and forms of art through dance) and is a vehicle for conveying emotion during social interactions [42]. Human movement provides an immensely rich context to images, video, and 3D worlds that centre around our experiences [39, 53]. Recently, machine learning models that learn from images and video have achieved high performance in a number of tasks previously considered challenging [30, 17, 59] e.g., recognition. Since these achievements, a greater emphasis has been placed on perceiving human behavior by applying similar techniques to human motion [50, 41, 21].

Toward this goal, researchers have begun to leverage the rich representational capacity of deep neural networks (DNNs) [18] to model the space of human motion. These models can be used for improving animation tools for artists through automatic animation comple-

tion [58], motion planning for autonomous vehicles via predicting pedestrian trajectories [57], and the generation of realistic avatars for augmented and virtual reality [13].

While progress has been made in these domains, modelling long temporal sequences continues to be a challenge for DNN approaches. Autoregressive methods, characterized by sequentially modelling future time steps given the past, typically model only a single time step at a time. In a common task like motion forecasting, this framework limits us to modelling motions up to a mere four seconds [15, 38, 35]. This is due to the accumulation of errors when sequentially sampling from a continuous space conditioned on the previous sample. At each time step, any errors in previous samples are propagated to successive time steps. Another commonly occurring phenomena is regression to the mean, which occurs by construction when assuming the output distribution of future poses is unimodal, i.e., there is only one possible future. The diversity of plausible future motions means our underlying data distribution is multi-modal. Therefore our converged solution can only model a single mode, and seeks to average the many modes of the underlying data distribution to reduce the training loss.

Handling long temporal motion sequences is also challenging in animation. Motion capture data is often unstructured with a single subject having up to hours of footage in a single sample. One difficult job of 3D animators is searching through unstructured motion capture footage when producing animation for games and film [52]. Once again, the extended duration of sequences becomes a significant bottleneck in a variety of tasks, especially when working with pose sequences. Various industrial applications exist for compressed efficient motion representations, including motion retrieval, where given a short motion sample, an animator might want to search a database of motions captured

from a studio for similar motions to their sample.

With the advent of computer animation supported by software tools and algorithms, many of the most laborious parts of the animation process have been greatly simplified. One of these processes is key-framing and interpolation [31]. Production studios typically animate sequences of frames in a non-sequential, irregular fashion divided into two major stages. In the first stage, key artists, are first responsible for animating key-frames that block out the overall intended action of a character in the scene, as well as the relative timesteps at which they occur. Key-frame's are placed at irregularly spaced intervals according to the desired action, which is a principle of animation known as "timing and spacing" responsible for giving characters a dynamic and life-like feeling.

In traditional 2D animation, key-frames and their relative times were passed to in-between artists that would be responsible for the laborious task of filling in missing frames of the motion. More recently, when animating in 3D, animators leverage various interpolation functions to complete the motion that was originally outlined at a coarse temporal resolution. With the help of these interpolators, the animation sequence is now brought to the desired frame rate ensuring that the major action-defining frames occur at the specified times.

In this thesis, we target the challenging task of modelling pose sequences over long time horizons inspired by the computer-assisted animation pipeline. According to the commonly modelled durations in previous work, we consider long-term motions to have any duration beyond four seconds. We show that a key-frame-based motion representation learned in a completely self-supervised fashion can effectively summarize pose sequences through a generative learning framework. We explore discrete and continuous

latent representations for key-frames, and show the effectiveness of our approach for pose sequence reconstruction and explore various architecture choices for this task. Without labelled training data, which is expensive to acquire, we explore architectures for the self-supervised discovery of key-frames and define a representation of time suited to represent the learned placements of these key-frames. We also show that a simple linear interpolation scheme provides perceptually reasonable results to fill motions between learned key-frames. A learned key-frame-based representation of pose sequences presents a myriad of potential downstream benefits. For example, we would be able to represent motion sequences by only a small fraction of their poses saving memory. Further, we can take advantage of the learned latent space of our generative model (operating in a temporally downsampled regime) for tasks like efficient motion retrieval or even long-term motion prediction without sacrificing multi-modality.

In summary, our contributions are as follows:

- We present a non-autoregressive method for modelling long-term human motions and demonstrate reconstructing 12 second motion durations.
- We recast the problem of long-term motion modelling by proposing a method to disentangle motions in terms of key-frames and in-between frames inspired by the traditional cel animation pipeline, greatly simplifying the extension to downstream tasks like long-term motion forecasting or retrieval.
- We present both a vector quantized and continuous cross-attention key-framing transformer to temporally subsample a motion sequence into key-frames and their placements in time.

Empirically, we show that our method outperforms common sequence modelling baselines for reconstruction trained on MotionSynth [25] and Ubisoft LAFAN1 [22] datasets. We show that learning to place key-frames more closely models the long-tail component of the ground truth velocity distribution in comparison to even spacing of key-frames.

Chapter 2

Background

This section presents the relevant works within generative modelling that lay the foundation for our method. We additionally motivate our approach by discussing adjacent work that demonstrates the challenging nature of long-term sequence modelling in various downstream tasks.

2.1 Generative methods

In deep probabilistic generative modelling, the aim is to learn to approximate a data generating distribution, $P(X)$, from a set of observations, $X \in \mathbb{R}^D$, from our dataset. If we effectively learn to approximate $P(X)$, we can perform a few useful tasks. For one, we can sample new observations thereby generating novel data points that do not exist in the dataset. Additionally, if the distribution is modelled explicitly, we also gain the ability to measure the likelihood of new observations we may encounter at inference and reason about the uncertainty of generated observations for greater interpretability out of the box

compared to discriminative methods. We can approximate this distribution by constructing a generative process which we parameterize with a deep neural network.

2.1.1 Variational autoencoders

Latent Variable Models (LVMs) are a class of generative model which allow us to approximate the data generating distribution, and can be parameterized with a deep neural network. The motivation behind LVMs is the manifold hypothesis, which posits that within the high-dimensional space of data, such as the space of natural images, there exists a lower dimensional manifold in which the relevant data points lie. For example, in the case of RGB images of size 256×256, the image space D is $255^{256\times256\times3}$. However, a large portion of this space likely does not contain perceptually relevant information. LVMs try to capture this lower dimensional latent manifold. Variational Autoencoders [28] are a type of LVM where the generative process is constructed in terms of some latent variable, $z \in \mathbb{R}^K$, where $K < D$ and the data distribution is factored as $p(x, z) = p(x|z)p(z)$. Given that we only have access to our observations, x, we marginalize out the latent variable according to

$$p(x) = \int p(x|z)p(z)dz. \tag{2.1}$$

With Monte Carlo sampling, we could in theory sample many latents, z, from our prior, $p(z)$, to approximate the expectation of this distribution. However, as the dimensionality of z increases, the amount of samples needed to cover the space increases exponentially due to the curse of dimensionality. When learning a lower dimensional manifold of a high

dimensional space such as natural images, the size of z is still large enough such that this issue arises. One solution to this problem is variational inference, where the logarithm of the marginal likelihood is approximated using an amortized variational posterior giving the following lower bound on our likelihood:

$$\ln p(x) \geq \mathbb{E}_{z\sim q_\phi(z|x)} \left[\ln p_\theta(x|z)\right] - \mathbb{E}_{z\sim q_\phi(z|x)} \left[\ln q_\phi(z|x) - \ln p(z)\right], \tag{2.2}$$

where the first term on the right-hand side of the equation represents the reconstruction objective "decoding" the observation from a latent z encoded using our approximate posterior in a stochastic manner, and the second term represents the Kullback-Leibler (KL) divergence between the approximate posterior and our prior latent distribution. In practice, our approximate posterior and decoder are parameterized using neural networks with weights θ and ϕ, respectively, and the choice of prior is often Gaussian.

2.1.2 Vector quantized variational autoencoders

Until recently, most generative modelling methods focused on learning a continuous approximation of the data-generating distribution. Intuitively, many common data modalities are continuous making the choice sensible. However, consider the way we tend to think abstractly about the world as a whole. This is done mainly in a compositional manner, using discrete symbols, like language and object-level reasoning. This is demonstrated by the way we use language to describe images. Language is fundamentally discrete, we string together words to represent various phenomena allowing us to think abstractly. Perhaps learning to represent the data distribution in a discrete formulation can provide

downstream benefits for complex reasoning, planning and prediction in a compositional fashion.

Van Den Oord et al. [48] introduced an extension of variational autoencoders that replaces the continuous Gaussian prior with a discrete learned dictionary/codebook to represent our latent variable, z. This formulation, dubbed Vector Quantized Variational Autoencoders (VQ-VAE), provides a few immediate benefits. For one the common issue of posterior collapse, in which the latent variable is ignored is now circumvented in the absence of the Gaussian prior. Instead, we opt for a prior and posterior that index a finite, discrete dictionary. Additionally, enforcing a finite dictionary can simplify the learning process. With a finite representational capacity in the latent space, the network is forced to encode the most relevant high-level information, instead of modelling local low-level imperceptible details. Once the discrete latent dictionary is learned, we can learn a prior over the discrete latent variables for downstream compositional tasks.

The VQ-VAE is implemented using a traditional autoencoding framework, where an encoder maps an input observation, x, into an initial continuous latent representation, z_e. The latent embedding space, $e \in \mathbb{R}^{K \times D}$, can be thought of as a dictionary or codebook with K vectors each with dimensionality D. The discrete latent representation, z_q, for a given input, x, is then computed using a simple nearest neighbour search along the dictionary for the closest entries, $e_j \in 1, \ldots, K$, for every z_e^i for $i \in 1, \ldots, C$, where C is the number of discrete pieces that the z_q is composed of. The nearest neighbour search is defined as follows:

$$z_q^{(i)} = e_k \text{ where } k = \arg\min_j \left\| z_e^{(i)} - e_j \right\|_2. \tag{2.3}$$

Since the selection process is non-differentiable, during backpropagation the gradients from the decoder input, z_q, are copied through to the encoder output, z_e, through stop gradients [7], denoted as sg []. The codebook entries, e, are updated via vector quantization which uses the l_2 distance measure to move codebook entries, e, towards the encoder outputs, z_e, during training. The final objective then becomes

$$\mathcal{L} = \log p\left(x|z_q\right) + \left\|\text{sg}\left[z_e\right] - e\right\|_2 + \beta \left\|z_e - \text{sg}\left[e\right]\right\|_2, \tag{2.4}$$

where β is a scalar weight to enforce z_e to be closer to the codebook entries, e. The leftmost term represents the reconstruction objective of the decoder, while the codebook is optimized according to the middle term.

2.2 Sequence modelling with deep neural networks

The representational capabilities of DNNs have been shown to achieve promising results in a variety of tasks in computer vision, such as recognition, classification, and generation. Vanilla feed-forward multilayer perceptrons struggle when modelling modalities with a sequential structure. Specifically, they are limited to fixed-length input and output sizes, do not explicitly model sequential dependencies, and do not have a mechanism for retaining memory. Extending DNN capabilities to sequential data continues to be an active area of research, with the following architectures exploiting various inductive biases relevant to sequences.

2.2.1 Recurrent neural networks

One of the most popular approaches for sequence modelling is the use of recurrent neural networks [44], initially developed to include a notion of persistence to hidden representations and handle variable sequence lengths. The key idea is to leverage parameter sharing to apply the same model iteratively to each data point while maintaining a notion of recurrence. RNNs implement recurrence by having the output of the model at the current timestep be a function of the output or hidden state from the previous timestep. The hidden state at the current timestep, $\mathbf{h}^{(t)}$, and output, $\mathbf{o}^{(t)}$, is computed given the input, $\mathbf{x}^{(t)}$, in the forward pass from $t = 1$ to $t = \tau$ as follows:

$$\mathbf{h}^{(t)} = \tanh\left(\mathbf{b} + \mathbf{W}\mathbf{h}^{(t-1)} + \mathbf{U}\mathbf{x}^{(t)}\right) \tag{2.5}$$

$$\mathbf{o}^{(t)} = \mathbf{c} + \mathbf{V}\mathbf{h}^{(t)} \tag{2.6}$$

$$\hat{\mathbf{y}}^{(t)} = \text{softmax}\left(\mathbf{o}^{(t)}\right), \tag{2.7}$$

where $\hat{\mathbf{y}}^{(t)}$ are the probabilities over the output, $\mathbf{o}^{(t)}$, and $\mathbf{W}$, $\mathbf{U}$, and $\mathbf{V}$ are learned weights which are persistent across timesteps. In this instance, the recurrence is implemented via the hidden state being a function of the previous timestep's hidden state. According to this formulation, the hidden state encodes information about previous timesteps, but its fixed capacity makes it challenging to represent long-term correlations. Furthermore, RNNs are prone to vanishing and exploding gradients during training since the gradients at early layers in the unrolled network are computed by repeatedly multiplying partial derivatives of the same set of weights across each timestep. As a result, training can become increas-

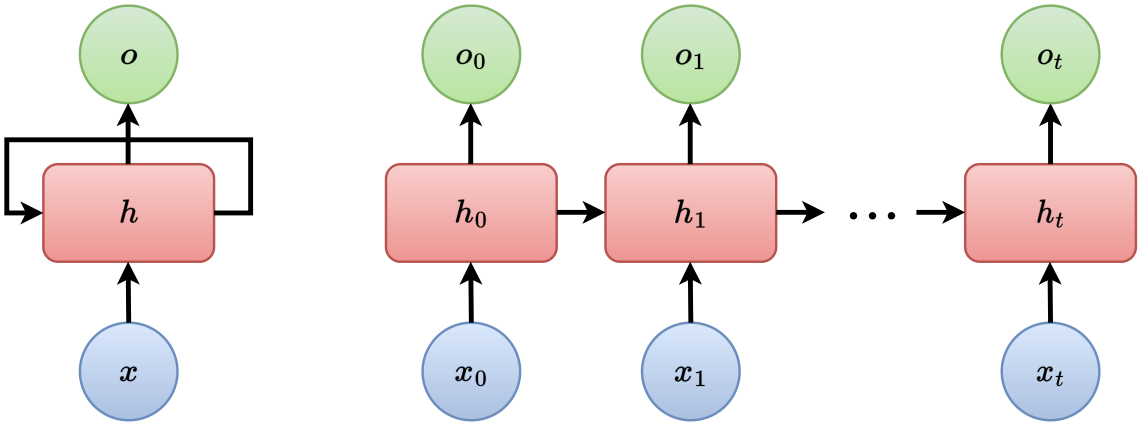


Figure 2.1: An unrolled RNN cell. At each timestep a hidden state h_t (Red) is computed according to Eq. 2.5 given a data point x_t (Blue) and the previous timesteps hidden state h_{t-1}. The output o_t (Green) is then computed according to Eq. 2.7. The hidden state learns to preserve temporal context from previous timesteps, but the length of this context may be rather limited.

ingly unstable as we increase the number of modelled timesteps. Since the success of the vanilla RNN, various extensions have been developed that build on and further improve recurrence schemes for more successful long-term modelling, including Gated Recurrent Units (GRU) [20], and Long Short Term Memory (LSTM) [23]. This is accomplished through the addition of extra computational units, such as gating mechanisms which improve gradient stability and encourage modelling longer-range dependencies by selectively determining what information to retain or discard at each timestep.

2.2.2 Convolutional neural networks

At a high level, Convolutional Neural Networks (CNNs) [32] for images are a cross-correlation between images and a series of learnable 2D filters whose weights are updated via gradient descent. The intuition is that the learned filters across layers reveal the

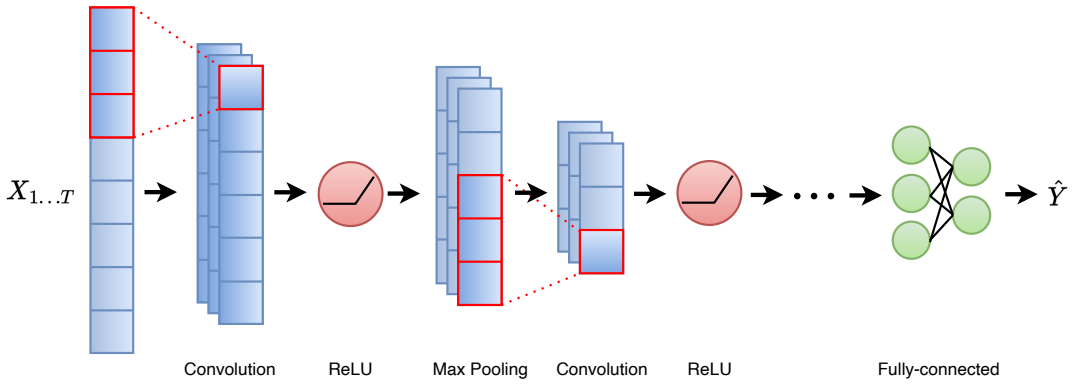


Figure 2.2: An example 1D CNN architecture for classifying a temporal signal. The temporal signal (Blue) $X_{1...T}$ undergoes a series of operations, including convolutions with learned filters, max pooling, and non-linearities. The number of channels in the response is determined by the number of 1D filters that are applied at the layer. Each successive convolution operation produces activations with a greater receptive field, meaning the kernel operates over a larger temporal context at later layers.

underlying hierarchical features that represent an image. Specifically, they leverage the inductive bias of local pixel neighbourhoods roughly the size of the kernel having relevant and related semantic information. At earlier layers, the cross-correlation between 2D filters and local pixel neighbourhoods reveals low-level features, such as edges and lines. These features hierarchically compose more complex features, like objects, at later layers as the receptive field of the activations from the cross-correlation grows.

For sequential data, a simple extension of this approach is the 1D convolution over temporal signals. The cross-correlation is now computed over a single dimension instead of two. Similar to the 2D case, the 1D convolutional filters learn to recognize local temporal features contextualized over the temporal window defined by the length of the kernel. As the temporal receptive field grows per layer, the cross-correlation captures larger temporal contexts and hierarchical features.

2.2.3 Transformers

A popular task in sequence modelling is machine translation. When translating between two different languages, it is common for the placement of associated nouns, pronouns, adjectives and even sequence length to differ between languages. The computational definition of attention [5] was initially introduced to handle this challenging domain by providing a novel way to quantify the association between tokens in different sequences irrespective of their placement or sequence length. Vaswani et al. [49] take this notion a step forward developing the well-known transformer architecture to be an attention-only sequence modelling method without convolutions or recurrence layers.

Self-attention. The attention mechanism can be more intuitively understood from the lens of retrieval systems. Given key and value pairs from a database, queries are used to compute relevancy against keys in our database, finally returning the value of the most relevant item in our database. In the attention mechanism, our input tokens $\mathbf{X} \in \mathbb{R}^{T \times d}$ are linearly projected to create our query tokens, $\mathbf{Q} \in \mathbb{R}^{T \times c_q}$, where T is the sequence length, d is the embedding dimension, and c_q is the channel dimension of the queries. The relevancy score is then computed via dot product with key tokens, $\mathbf{K} \in \mathbb{R}^{T \times c_q}$, also projections of the input tokens to a common space. The normalized relevance scores of any given query token against all other key tokens are now used to weight our value tokens, $\mathbf{V} \in \mathbb{R}^{T \times c_v}$, where c_v is the embedding dimension of the value tokens. This tells us what tokens in the input sequence are particularly relevant or "attend" to said query token. Typically c_q and c_v are the same size. The operations are as follows

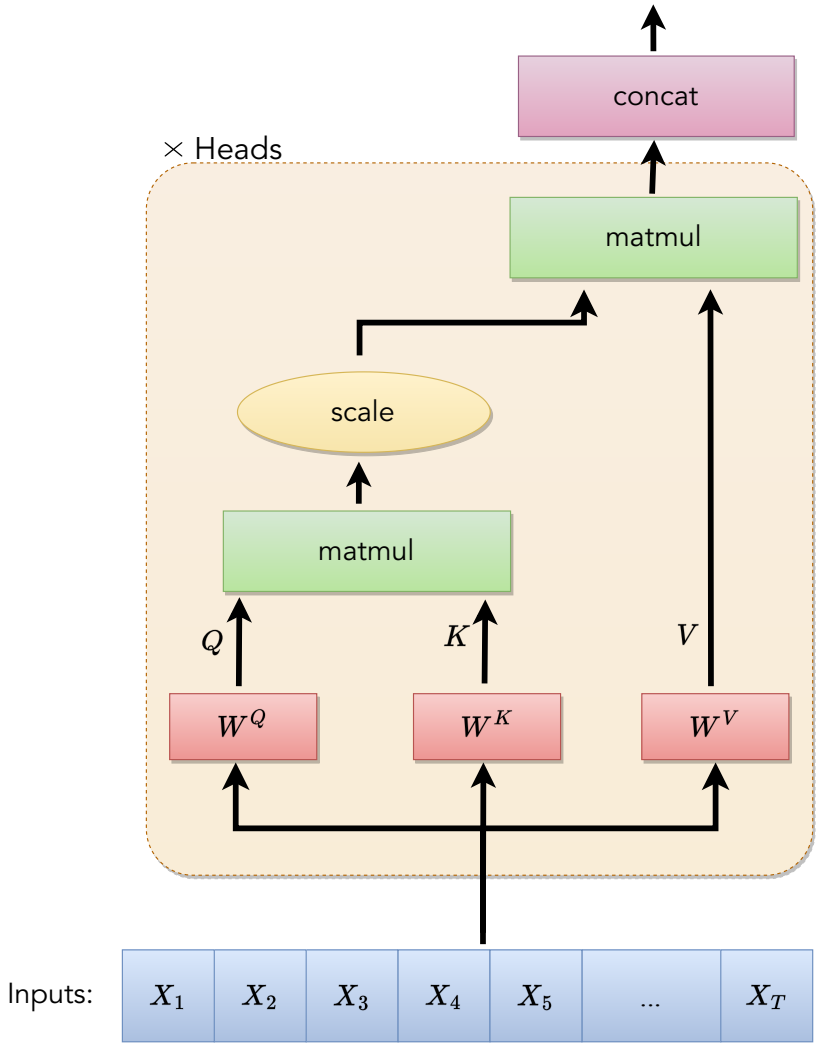


Figure 2.3: A multi-head attention block implemented in the transformer architecture. The input motion $X_{1:T}$ is projected into a common space by $\mathbf{W}^Q$, $\mathbf{W}^K$ and $\mathbf{W}^V$. This gives the queries, keys, and value tokens $\mathbf{Q}$, $\mathbf{K}$, and $\mathbf{V}$, respectively. Relevance scores are computed between all queries and keys via matrix multiplication. The scores are multiplied against $\mathbf{V}$ to determine the value tokens that most closely correlate to the given query tokens. This computation is done independently across the number of heads, and finally combined and projected with another set of learned weights $\mathbf{W}^O$.

$$\begin{aligned}\mathbf{Q} &= \mathbf{X} \cdot \mathbf{W}^{Q} \\ \mathbf{K} &= \mathbf{X} \cdot \mathbf{W}^{K} \\ \mathbf{V} &= \mathbf{X} \cdot \mathbf{W}^{V},\end{aligned} \tag{2.8}$$

where $\mathbf{W}^{\{Q,K\}} \in \mathbb{R}^{d \times c_q}$ and $\mathbf{W}^{V} \in \mathbb{R}^{d \times c_v}$ are learned weight matrices used to project our input tokens $\mathbf{X}$ to a common space. The full-attention operation is given by

$$\text{Attention} = \frac{\text{softmax}\left(\mathbf{Q}\mathbf{K}^{\top}\right)}{\sqrt{d_k}}\mathbf{V}. \tag{2.9}$$

In the traditional transformer architecture, the attention framework is extended to be hierarchical and multi-stream. Each layer may compute multiple attention operations independently depending on the number of attention "heads" that are defined per layer, as shown in Fig. 2.3. Each attention head can be thought of as a new stream with different independent weights for linearly projecting the query, key, and value tokens. The transformer block is then implemented by alternating multi-head attention blocks and linear layers with residual connections in between, as shown in Fig. 2.4.

Cross-attention. In its current formulation, multi-head self-attention lets us determine the soft relevance of tokens within a sequence with respect to other tokens within the same sequence. There are many situations where we may want to determine how one sequence of tokens may "attend" to a completely different sequence of tokens of a different modality and sequence length. A simple extension of self-attention for this purpose is cross-attention, as visualized in Fig. 2.5. Cross-attention is equivalent to self-attention when the sequence lengths of both signals are the same.

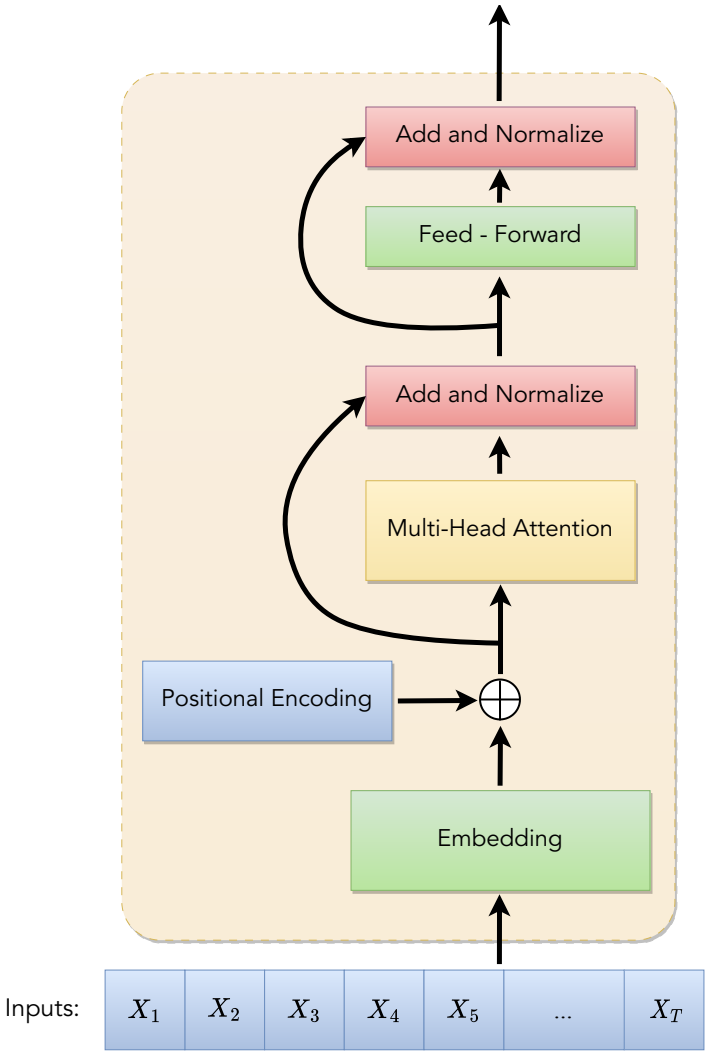


Figure 2.4: The series of operations that comprise a transformer block. The transformer architecture is a sequence modelling approach based on the attention mechanism without any notion of recurrence or convolutions. Given an input sequence, $X_{1:T}$, the sequence is first passed through an embedding layer, which is implemented as a multi-layer perceptron (MLP). The embedded sequence is then positionally encoded by a series of alternating sinusoidal functions of increasing frequency. This gives each token a unique encoding of its relative position in the full sequence. The encoded sequence is then passed through the multi-head attention block and an MLP. The number of tokens output from the final MLP is the same as the number of input tokens.

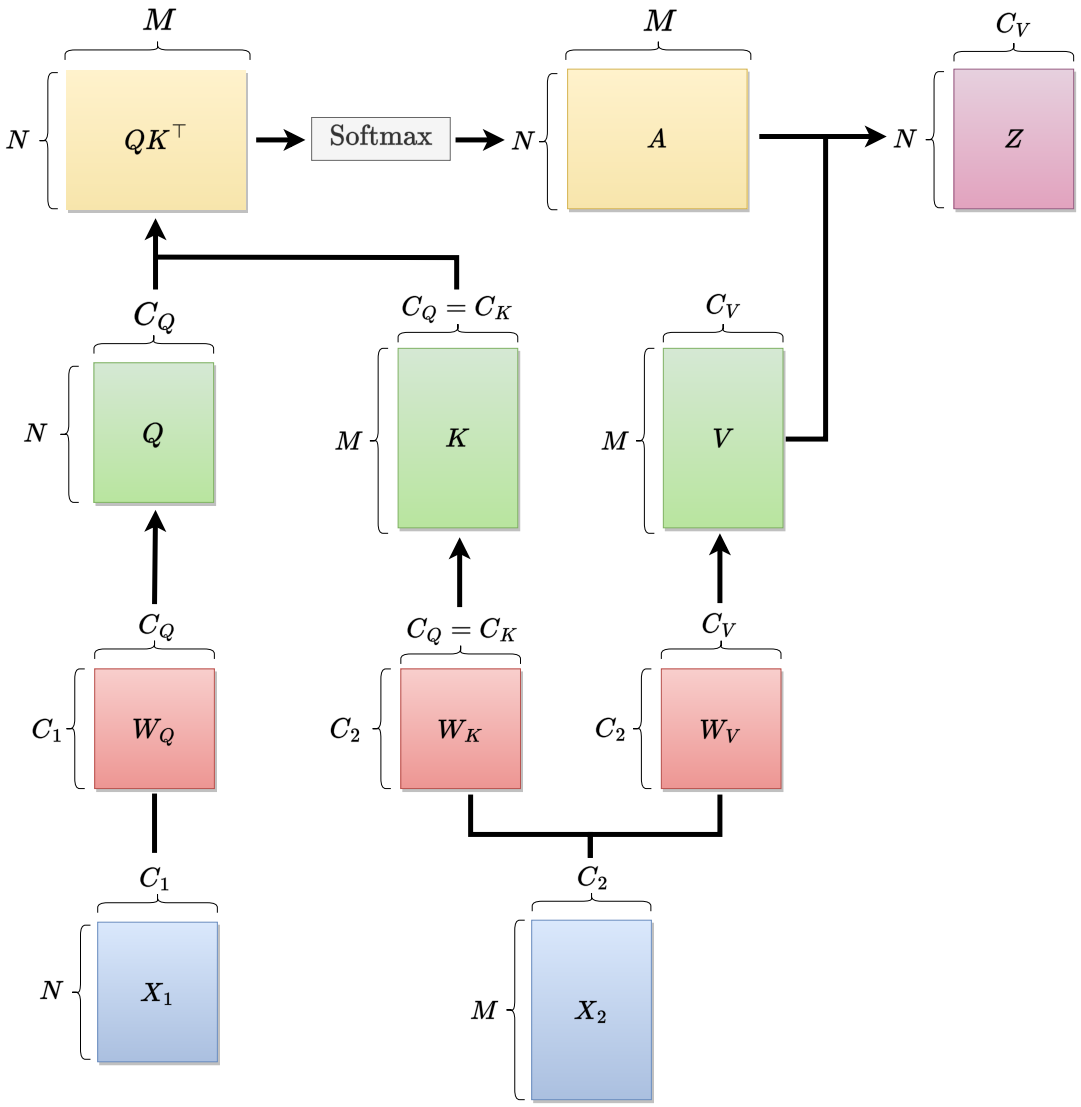


Figure 2.5: An example cross-attention operation between two sequences, X_1 and X_2. Cross-attention is an extension of self-attention that computes attention weights between sequences of different modalities that may have different dimensionality and sequence length. The overall computation is equivalent to self-attention when both sequences have the same length. Here, N and M are the length and C_1, C_2 the dimensionality of the two sequences X_1 and X_2, respectively. The queries are computed from X_1 while the keys and values are computed from X_2. To compute attention between two different sequences, the dimensionality between the learned weights W_Q and W_K remain the same.

2.3 Related work

To motivate our key-frame based representation for pose sequences, we present relevant adjacent work in tasks like motion forecasting to demonstrate the challenging nature of extending to long-term pose sequences. Specifically, these works consider long-term motions to be up to four seconds, and short-term to be one second or less. Our approach, which includes a discrete latent key-frame representation is naturally suited to extend to these challenging downstream tasks while preserving key qualities like multi-modality.

2.3.1 Unimodal motion forecasting

Previously, most efforts in this domain [10, 2, 19, 51] approached human motion forecasting as a sequence modelling task. Therefore, most of these methods use RNNs to predict a single motion at a time, given a past motion sequence. These early efforts were successful in predicting sequences up to just one or two seconds depending on the frame rate of the captured motions. While the resulting predictions were perceptually reasonable, they do not fully capture the underlying uncertainty of human motion. Given a past motion sequence, there is rarely a single plausible motion sequence, but multiple plausible motion futures. Due to the underlying assumption that future poses are sampled from a unimodal distribution, the converged solution of many of these works predict the average of possible motion futures [38, 15] from the constraint of seeking a single mode due to the multivariate Gaussian distributed output. Methods based on recurrent networks are also fundamentally bottlenecked by the fixed capacity of the recurrent hidden state, causing them to struggle with modelling longer-range dependencies necessary to predict longer motion sequences.

Error drift is also very common, as predicted poses are fed back as input to predict the following timesteps, small errors/noise tend to accumulate until complete deterioration.

2.3.2 Multi-modal motion forecasting

More recently, an emphasis has been placed on learning the stochastic nature of human motion [3, 6, 37, 55, 56], modelling the inherent uncertainty using techniques from generative modelling. Specifically, to address diversity when randomly sampling the latent prior of a pre-trained generative model, Yuan *et al*. [56] introduce a novel sampling strategy based on a set of learnable latent mapping functions. While this approach significantly improves diversity for generative sampling, they are still limited to short-term motion prediction of only 100 frames due to error accumulation and drift.

Mao *et al*. [37] approach the sample diversity issue by learning a separate latent representation of motion for the upper and lower body. To generate diverse future motion sequences, they first sample latent codes from the lower body model, and conditionally synthesize associated upper body motions per lower body latent sample. Again, they limit the duration of the predicted motion to between 60 and 100 frames amounting to about two seconds of motion.

While we do not explore motion forecasting directly in this thesis, our architecture is designed and motivated with the challenging downstream task of motion forecasting in mind. Specifically, we aim to design a flexible temporally compressed human locomotion model to eliminate the need for autoregressive methods and overcome their shortcomings for long-term sequences. By doing so our model for human locomotion could be utilized for downstream tasks that handle multi-modal long-term motion sequences.

2.3.3 Long-term motion forecasting

While there is no explicitly defined duration considered "long-term", based on the durations modelled in recent work, we consider it anything beyond four seconds. A few works have attempted to address long-term forecasting for human motion modelling by rethinking the sequential modelling framework [35, 12, 36]. The closest to key-framing is Diller *et al*. [12], who recast the problem in terms of only predicting what they define as "characteristic poses" which are only the critical poses that outline an action or a person's intent. By only predicting characteristic poses, they effectively decouple poses from time, instead generating poses in a goal-oriented fashion. While in essence these characteristic poses can be used to generate key-frame poses, they are generated in an iterative autoregressive fashion in a continuous space which can still be prone to error accumulation. Training such a model also requires an extra data labelling task to note the characteristic poses in the dataset, opening an avenue for possible human error. Ultimately our approach does not require novel data labelling and the network itself learns the placement of keyframes in an unsupervised fashion. Lastly, it is unclear how these characteristic poses can be used to synthesize full motion sequences when they are decoupled from time.

Lucas *et al*. [36] move away from continuous latent generative models in favour of learning a discretized latent representation through the adoption of vector-quantized variational autoencoders. This class of discrete generative models has recently shown promise in modelling image space through discrete tokens. Through a latent codebook learned by reconstructing images, new images can be sampled autoregressively from codebook indices directly via a separately trained network. For the task of long-term motion prediction, sampling codebook indices instead of a continuous latent representation prevents

noisy out-of-distribution samples allowing long-term predictions of up to 300 frames or about 10 seconds. However, they still rely on a fundamentally autoregressive architecture, sampling only one timestep at a time. While our key-frame based pose representation also builds on vector quantization to prevent drift, we also learn to decompose motions in terms of key-frames. This means downstream tasks that leverage our quantized motion representation would only need to sample a key-frame number of quantized latents which can be achieved without the need for autoregressive methods.

2.3.4 Human motion synthesis

Synthesizing human motion directly from modalities like text and video can be a useful tool for animation and a core component of augmented and virtual reality. Thus, a parallel direction of research aims to conditionally synthesize motion according to other data modalities, like action descriptions [11, 21, 1], video [29], music [33, 34] and more. In these works, the diversity of synthesized motion samples is again the primary interest of recent literature [16], and the duration of resulting synthesized motions is once again limited to mere seconds. The exception is methods that extract 3D poses directly from video [29]. Here, the extra modality can serve as a strong additional signal for conditional synthesis avoiding the need for long-term correlations to be learned directly from predicted poses. These adjacent tasks could benefit from our key-frame representation of pose sequences. Through latent key-frames, we reduce the computational burden of learning correlations between modalities by representing a longer motion sequence by a fraction of the original number of frames. Additionally, in our discrete generative formulation, sampling the codebook conditioned on a signal from a different modality can preserve a notion of

stochasticity in the synthesized motions.

2.4 Summary

In this chapter we presented relevant concepts in generative modelling, including variational autoencoders and their vector quantized counterparts, that form the foundation for our learned key-framing method. We also reviewed two common learning-based sequence modelling methods, recurrent neural networks and 1D convolutional neural networks, that have been used in related work to model human motion. We motivated the problem of long-term human motion modelling in the context of related downstream tasks, including motion forecasting and conditional motion synthesis. In these contexts, motion sequences are typically modelled autoregressively and can suffer from regression to the mean and error accumulation. In both instances, the durations of modelled motions are often limited to only four seconds. While we do not directly address these tasks in this thesis, our key-framing approach is motivated by the idea of learning a temporally compressed motion representation for these downstream tasks. Rather than rely on single time step autoregressive modelling for forecasting motions or multi-modal synthesis, we can instead learn these tasks in the space of learned latent key-frames and naturally extend to long-term motions.

Chapter 3

Technical Approach

In this chapter, we suggest that the core principles of animation, key-framing and interpolation, have a natural extension to sequence modelling using a two-stage generative framework that closely resembles the two-stage key-frame animation pipeline. In the first stage, we learn a latent motion representation of key-frame poses. We explore two different methods to accomplish this, one based on continuous autoencoding and the other based on learning a discrete library of key-frame poses. Additionally, we learn to predict their relative timestamps in an unsupervised fashion all within the proxy task of motion reconstruction. In the second stage, once key-frames are generated, we mimic the role of an in-between artist through the use of interpolation mechanisms in pose space.

3.1 Latent key-frames

To translate key-frame based animation principles to generate motion representations, we must first find an effective representation of key-frames. Inspired by real-time character

animation, where motion libraries for character control are common, we learn a discretized codebook of key-frames. We look to the recent success of discrete generative models, like Vector Quantized Variational Autoencoders (VQ-VAE) [48], that learn a latent representation encoded into a discrete codebook. Through such a representation, we can learn to summarize key poses within the entries of the codebook and sample these entries to compose a single pose or a sequence of poses that cover motion space, similar to the motion libraries in Holden *et al*. [26]. Additionally, we explore a non-quantized approach where our learned key-frames are represented by continuous latent vectors following traditional vanilla autoencoding.

Given a set of T motion observations, $X_{1:T} = (x_1, x_2, \cdots, x_T)$, where $x_i \in \mathbb{R}^{J\times 6}$ and J is the number of joints, we learn an encoding network, E, to temporally downsample the motion sequence to a continuous latent representation,

$$z_e, \Delta\hat{t}_{\mathcal{KF}} = E\left(X_{1:T}\right), \tag{3.1}$$

where $\Delta\hat{t}_{\mathcal{KF}} \in \mathbb{R}^{\mathcal{KF}}$ is our encoded one-dimensional representation of key-frame placement in time and $\mathcal{KF}$ is the temporal downsampling of the original sequence length T to the number of key-frames we choose to use, $z_e = (z_e^{(1)}, z_e^{(2)}, \cdots, z_e^{(\mathcal{KF})})$ and $z_e^{(i)} \in \mathbb{R}^c$ with c the channel size. In our continuous autoencoding approach for representing latent key-frames, the continuous latents z_e and $\Delta\hat{t}_{\mathcal{KF}}$ are now passed directly to the decoder, D.

3.1.1 Latent pose quantization

In our VQ approach, a quantization operator, $q(\cdot)$, maps each latent, $z_e^{(i)}$, to the closest entry in the codebook according to

$$z_q = q(z_e). \tag{3.2}$$

More specifically, the quantization of an individual continuous latent vector, $z_e^{(i)}$, by our operator, $q(\cdot)$, can be thought of as finding the index, $s^{(i)}$, of the codebook with the closest corresponding entry:

$$s^{(i)} = \arg\min_j \left\| z_e^{(i)} - e_j \right\|_2. \tag{3.3}$$

As the latent space is now represented as a series of fixed discrete codebook entries, an alternate equivalent way to represent z_q is as a set of integers, $S = \left(s^{(1)}, s^{(2)}, \cdots, s^{(\mathcal{KF})}\right)$, such that $\left\{s^{(i)} \in \mathbb{Z} \mid 1 \leq s^{(i)} < K\right\}$, where K is the number of codebook entries. The quantized latent vector, $z_q^{(i)}$, can then be constructed according to:

$$z_q^{(i)} = e_{s^{(i)}}, \tag{3.4}$$

where $e_{s^{(i)}}$ is the selected codebook entry at index $s^{(i)}$ that is closest to the continuous latent entry, $z_e^{(i)}$.

Furthermore, sampling this discrete space reduces error accumulation as each discrete entry is more likely to be in distribution compared to random samples of a continuous space [36]. This is highly beneficial for further avoiding drift common for previous long-term human motion forecasting methods [36].

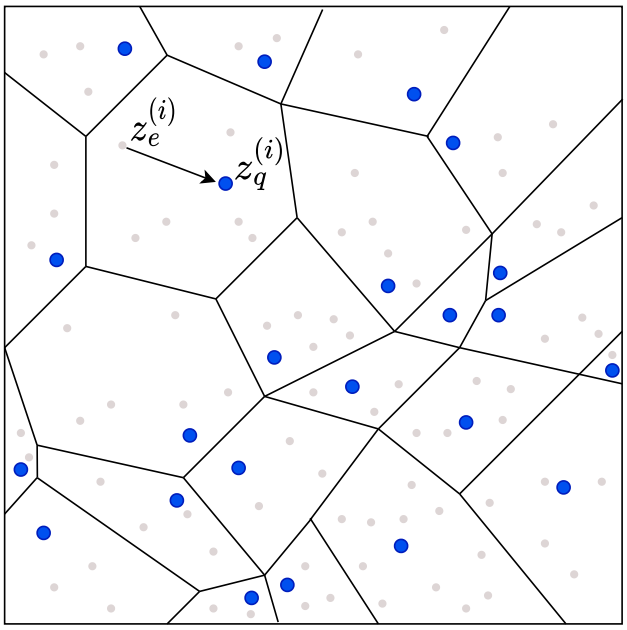


Figure 3.1: Voronoi tesselation depicting an example discretization of the continuous latent space, z_e, in two dimensions. Blue points represent learned codebook entries, $z_q^{(i)}$. Grey points represent examples of continuous latents, $z_e^{(i)}$, produced by our encoding of the motion sequence, $E(X_{1:T})$. The full discrete latent, z_q, is constructed by sampling the corresponding closest discrete entries for each $z_e^{(i)}$ for $i \in 1, 2, \cdots, \mathcal{KF}$.

3.2 Irregular key-framing

Now that poses can be efficiently represented as either a discretized codebook or continuous latents that represent individual poses or short pose sequences according to the choice of temporal downsampling factor, the question becomes how to naturally discover the key-frame structure of human motion, including the distance between frames in a completely unsupervised fashion. We choose to take advantage of the fact that many interpolation methods, including Spherical Interpolation (SLERP), are differentiable. Therefore, we can recast the traditional task of VAE-based reconstruction into a summarization task in which we only reconstruct a subset of the total frames of the original signal i.e., only the key-frames. Along with the key-frames themselves, we also predict the associated relative time-elapsed between all adjacent key-frames. This is all achieved in the decoding stage according to:

$$\hat{X}_{\mathcal{KF}}, \Delta t_{\mathcal{KF}}^{\text{logits}} = D\left(q\left(E\left(X_{1:T}\right)\right), \Delta\hat{t}_{\mathcal{KF}}\right), \tag{3.5}$$

where D is our decoder network, $\Delta t_{\mathcal{KF}}^{\text{logits}}$ is a one-dimensional sequence of logits of length $\mathcal{KF}-1$ representing the relative spacing of each key-frame from the previous key-frame, and $\hat{X}_{\mathcal{KF}}$ are the reconstructed key-frames in pose space. In the continuous autoencoding case, the decoding process is the same apart from the quantization operator, $q\left(\cdot\right)$.

We choose to use a continuous representation of time between $t=0$ and $t=1$ to represent the beginning and end of a full motion sequence. To ensure our key-frame time stamps sum to one and are strictly increasing, we choose to predict the *elapsed time* per key-frame and feed the logits from our decoder through a softmax:

$$\Delta t_{\mathcal{KF}} = \text{softmax}\left(\Delta t_{\mathcal{KF}}^{\text{logits}}\right). \tag{3.6}$$

The absolute times of each key-frame can then be computed through a cumulative sum according to:

$$t_{\mathcal{KF}}^{(i)} = \sum_{i=1}^{\mathcal{KF}} \Delta t_{\mathcal{KF}}^{(i)}. \tag{3.7}$$

In the first stage, the network is now encouraged to learn to accomplish two things to lower the reconstruction error. First, summarize the input motion sequence in terms of a fraction of the original frames. These frames are synthesized according to either a discrete compositional or continuous latent motion library and key-frame poses should be "characteristic" of the overall action. Since we output key-frame poses alone, our latent motion library only describes characteristic key poses of the learned motion space, instead of all plausible poses available in the dataset if we were directly reconstructing the entire sequence. Second, determine the elapsed time between key-frames, which ultimately determines where there will be variations in velocity in the output motion relating to the principle of "timing and spacing" in traditional animation.

We notice experimentally that our formulation for key-frame time can become unstable early in training. Specifically, the elapsed-time between key-frames can go to zero between some key-frames creating a local minima where the network can no longer produce an elapsed-time above zero at said key-frame. To stabilize the elapsed-time post initialization, we choose to employ entropy regularization on the elapsed-time softmax distribution,

$$H\left(\Delta t_{\mathcal{KF}}\right) = -\sum_{i=1}^{\mathcal{KF}-1} p\left(\Delta t_{\mathcal{KF}}^{(i)}\right) \log p\left(\Delta t_{\mathcal{KF}}^{(i)}\right). \tag{3.8}$$

This way, at the start of training, the network learns to effectively represent key-frames as if they were evenly spaced with each other. Over time, the regularization weight is decayed and the network can place key-frames more freely.

Given key-frames and their associated elapsed times, we can simply interpolate the key-frames to the final desired frame rate. For our experiments, we choose a linear interpolator like SLERP, although more sophisticated schemes are possible, including learned ones. Two quaternions, q_1 and q_2, can be interpolated by an arc length, u, by

$$\text{SLERP}\left(q_1, q_2, u\right) = \frac{\sin\left(1-u\right)}{\theta} q_1 + \frac{\sin\left(u\theta\right)}{\sin\theta} q_2. \tag{3.9}$$

The reconstructed motion sequence of length T is interpolated as follows:

$$\hat{X}_{1:T} = \text{SLERP}\left(\hat{X}_{\mathcal{KF}}, t_{\mathcal{KF}}\right). \tag{3.10}$$

The details of our SLERP function extended to irregular spacing are discussed in the appendix.

The final VQ-VAE summary-based reconstruction objective then becomes

$$\begin{aligned} L = \left\| \hat{X}_{1:T} - X_{1:T} \right\|^2 + \left\| \text{sg}\left[E\left(X_{1:T}\right)\right] - z_q \right\|_2^2 + \\ \beta \left\| E\left(X_{1:T}\right) - \text{sg}\left[z_q\right] \right\|_2^2 + \lambda H\left(\Delta t_{\mathcal{KF}}\right). \end{aligned} \tag{3.11}$$

Due to the quantization process being non-differentiable, the VQ-VAE objective involves the use of stop gradients, denoted as sg [], such that gradients from the decoder are copied

over the encoder. The first term represents the likelihood term, $\log p\left(X_{1:T} \middle| z_q\right)$, and the codebook entries are optimized by the middle term. The second last term ensures the continuous latent, z_e, is close enough to entries in the codebook. For our alternative continuous autoencoding approach, the loss objective is simply the reconstruction term summed with the entropy regularization term:

$$L = \left\| \hat{X}_{1:T} - X_{1:T} \right\|^2 + \lambda H\left(\Delta t_{\mathcal{KF}}\right), \tag{3.12}$$

which corresponds to the first and last terms in Eq. 3.11.

3.3 Temporal subsampling with transformers

The question now is how we learn to effectively decompose a full motion sequence into its constituent key-frames. In essence, our goal is to temporally subsample pose sequences to a pre-defined number of relevant poses. When naively using existing popular sequence modelling architectures, like 1D CNNs, there is not an obvious way to output the desired number of key-frames apart from adjusting hyperparameters, like kernel size, stride, and padding or pooling. Even with this approach, the captured temporal window for any given token in our sequence may be shallow depending on the size of the receptive field at a given layer. For the task of key-framing, any given key-frame may only be reconstructed according to its immediate neighbours. Depending on the complexity of the motion, considering the entire sequence could produce more optimal placements, especially due to our choice of softmax, any allocation of space between a pair of key-frames in one part of the motion results in less space between allocated between some other pair of key-frames

somewhere else in the sequence.

From this perspective, recent advances in attention mechanisms and transformers have useful properties that naturally handle this scenario. By definition, self-attention naturally "attends" to the full sequence at every layer, but the number of tokens output at every layer remains the same. To temporally subsample our full sequence while maintaining a global temporal context we can leverage cross-attention, visualized in Fig. 2.5, in tandem with self-attention. Initially introduced by Vaswani et al. [49], cross-attention is most commonly used to generate attention maps across signals from different modalities which often contain different sequence lengths. For our use case, we can apply cross-attention across our full sequence and a learned set of query tokens [9, 27]. The number of query tokens is equivalent to the number of key-frames we wish to learn.

The keys and values to our first cross-attention block is the input motion sequence whose channel dimension has been encoded by a feed-forward layer. Our first cross-attention block subsamples the full sequence (the keys and values), the output is then a key-frame number of poses. The following self-attention block globally contextualizes these resulting key-frames. Each successive self-attention block operates on our learned key-frames while the following cross-attention blocks now take our latent key-frames from our self-attention layer as queries, and the full input sequence as the keys and values. By alternating cross-attention and self-attention blocks within our key-framing transformer architecture, we effectively maintain a global temporal context while sub-sampling to our defined set of key-frames. This can be viewed as iterative sampling and refining sets of key-frames against our full sequence.

Our decoding stage is simply a 1D CNN decoder that takes as input our latent key-

frames, z_q, in our discrete formulation or z_e, in the continuous autoencoding case. The outputs are the reconstructed key-frames in pose space, $\hat{X}_{\mathcal{KF}}$, and the associated elapsed-time logits, $\Delta t_{\mathcal{KF}}^{\text{logits}}$. The complete motion sequence is then interpolated via SLERP. Our complete architecture is visualized in Fig. 3.2.

3.4 Summary

This chapter presented a method for reconstructing long-term motion sequences, which are defined as motions with durations beyond four seconds. Our approach circumvents the need for autoregressive methods by decomposing pose sequences into a series of key-frames and their associated placements in time. We explored a latent key-frame representation learned in a discrete latent codebook based on the VQ-VAE architecture [48] as well as learned via vanilla autoencoding. Key-frames are discovered in our cross-attention transformer architecture through the use of alternating cross-attention and self-attention blocks. Our first cross-attention block has a key-frame number of learnable query tokens which attend over the entire input sequence. This has the effect of temporally subsampling the original motion into a key-frame number of poses. Each successive cross-attention block then recontextualizes our key-frames against the original motion sequence while the successive self-attention block attends globally across selected key-frames. The complete reconstructed motion is then linearly interpolated to fill in missing in-between frames according to the decoded key-frames and their learned placements.

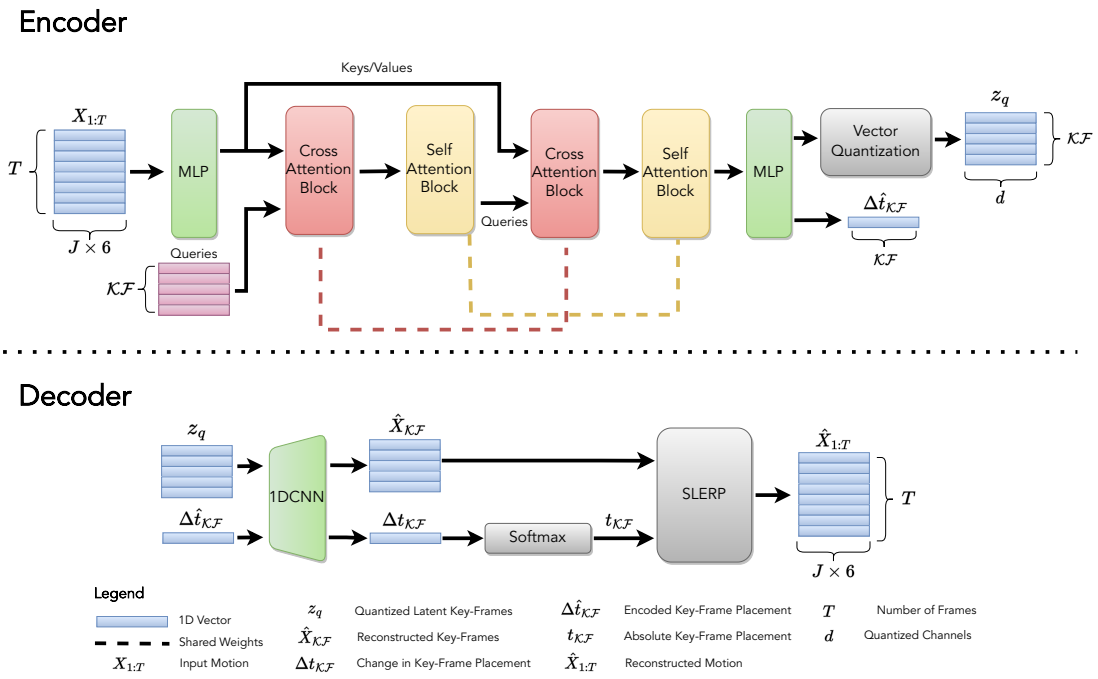


Figure 3.2: Key-framing transformer architecture. Our VQ-based framework for unsupervised key-frame discovery uses a transformer based architecture that alternates self-attention and cross-attention blocks. Starting with a learnable set of query tokens initialized randomly, the first cross-attention block can be viewed as learning to sub-sample the full motion sequence for only the most relevant key-frames. The output being a key-frame number of tokens that get recontexualized globally by the following self-attention block. In our discretized latent formulation, we then quantize the latent key-frame poses and ultimately decode them with a 1D CNN decoder that gives the key-frame poses and their placements. In the continuous case, latent key-frame poses are passed directly to the 1D CNN decoder.

Chapter 4

Experimental Evaluation

To validate our approach, we construct multiple baselines for both qualitative and quantitative comparisons. There are few directly related works that try to learn to summarize key-frames in a purely self-supervised fashion, those that do learn key-frame based representations require manually labelled ground truth [12]. The purpose of our approach is to determine if we can discover key-frames in the absence of explicit supervision. We therefore construct baselines that are naturally suited to validate our method.

4.1 Baselines

There are three primary axes of comparison which we use to better understand the efficacy of our key-framing transformer approach. The first is to understand the benefits of our use of attention, which we determine by implementing our learned key-framing scheme within an alternative sequence modelling architecture, the 1D CNN. Our 1D CNN baseline most closely resembles Holden et al. [24], one of the first works to learn a motion manifold with

variational autoencoders. The second is to compare the use of discrete versus continuous latent key-frames, which we explore within our proposed transformer architecture. The last is to determine how well our learned key-frame placements are by comparing them against the same set of models trained with key-frames placed regularly spaced apart. In each scenario we keep the relevant hyperparameters consistent as presented in the appendix.

We also evaluate our method by reconstructing sequences longer than the traditional one to four second intervals with varying numbers of key-frames, which we leave as a hyperparameter. We refer to the number of key-frames used to reconstruct the motion in terms of the key-frame sample rate in key-frames per second (kf/s). Specifically, we reconstruct sequences at two key-frames per second up to six key-frames per second for motion durations starting at four seconds, up to 12 seconds. For irregular spacing, the key-frame sample rate is not exact. If the network has the opportunity to allocate more key-frames in specific regions, there can be more key-frames in that particular second of motion than the key-frame sample rate. For consistency, we still use this nomenclature with irregular spacing and choose to think of it as the "average" key-frame sample rate in this setting.

4.2 Implementation and training

We implement our cross-attention transformer encoder with two sets of alternating cross-attention and self-attention blocks with shared weights. Our cross-attention blocks are comprised of a single layer with two heads and our self-attention block is comprised of

two layers with four heads. In our VQ case, the continuous latent representation from the encoder is discretized against a learned codebook with 1024 entries, with an individual entry containing 32 channels. In our non-VQ case, the continuous latent representation is passed directly to the decoder. Our decoder is implemented with two 1D CNN layers that output key-frame poses and the logits for elapsed time between key-frames. Empirically, we achieve better reconstruction results with a simple 1D CNN versus a transformer-based decoding scheme. Our choice of hyperparameters follows related work [9, 27] that utilize cross-attention and self-attention in tandem for multi-modal tasks. The input to the network are 3D pose sequences represented as a series of 3×3 rotation matrices per joint. The decoded key-frame poses are then interpolated with spherical interpolation over key-frame rotations converted into quaternions. For all datasets, we ensure there is no overlap between downsampled motions of the same sequence or subject between train and test sets. With these choices, our model is trained on a single Titan X GPU with a batch size of 32. Further details of our architecture are provided in the appendix.

4.3 Datasets and evaluation

MotSynth. Most publicly available datasets of human motion capture operate at 30 frames per second with few exceptions. Intuitively, datasets with motions captured at a high frame rate will contain adjacent frames whose poses are more similar than if the motion is captured at a lower frame rate. For modelling purposes, most datasets outright capture motions at 30 frames per second to enable enough variation in motion between adjacent frames for sequence modelling approaches to learn effectively. For our experiments mod-

elling key-frames, we effectively learn a temporal subsampling process that should ideally also handle higher frame rates since our learned representation is agnostic to the variations between directly adjacent frames, which are handled by our linear interpolator. To test this, we select MotionSynth [25] which contains six million frames of a single subject at the default frame rate of 120 frames per second. This dataset is twice the size of the CMU Motion Capture Database, which is another common benchmark dataset for human motion. MotionSynth also combines motions from other popular datasets [40, 54], including CMU, using a common inverse kinematics and skeleton retargeting procedure [25]. We reconstruct motions between four to twelve seconds at the original 120 Hz and at a downsampled version of the dataset at 30 Hz with the default skeleton definition in units of inches. At the longest duration and frame rate we reconstruct up to nearly 1440 frames for a single motion sequence. We use an 80-20 train-test split with the classes "walk", "run", "jog", and "transitions". Motion samples that are shorter than the target reconstruction duration are removed from both training and testing.

LAFAN1. Many popular human motion capture datasets contain relatively simplistic motions, with many variations on highly cyclic and predictable motions, such as "walking". For our purposes of discovering key-frames in a self-supervised manner, we choose to operate on more complex and unstructured motion datasets. LAFAN1 [22] is a recent dataset that contains a wide variety of highly dynamic movements expected of a video game character. Within individual motion samples, many motions are tagged with soft labels, i.e., the labels outline the overall predominant motion but they may contain other actions as well. Many motion samples contain motion transitions between walking, running, crouching, sitting and many more. As a result of our approach being self-supervised we can

fully leverage the complexity of the dataset without the need for specific labels. LAFAN1 contains five subjects with 77 sequences and over 400000 total frames at 30 frames per second. We use an 80-20 train-test split using the classes "walk", "run", "sprint", "dance", and "multiple actions" and use the default skeleton definition in units of centimetres. Motion samples that are shorter than the target reconstruction duration are removed from both training and testing, the same as for the MotionSynth dataset.

4.4 Metrics

Quantitative evaluations are based on measuring the displacement error between joint positions of our reconstructed motion sequence against the ground truth motion sequence. We evaluate this displacement according to two levels of granularity. Our more global error metric, the Average Displacement Error (ADE), measures the L2 distance between ground truth and reconstructed poses averaged across all frames:

$$\text{ADE} = \frac{1}{T}\sum_{t=1}^{T}\left\|\hat{X}_t - X_t\right\|_2. \tag{4.1}$$

To measure the error at the joint level, we use the Mean Per Joint Position Error (MPJPE) which measures the L2 distance between ground truth and reconstruction motions per joint per frame, and then averages across joints thereby giving the mean error per joint:

$$\text{MPJPE} = \frac{1}{N_j}\frac{1}{T}\sum_{j=1}^{N_j}\sum_{t=1}^{T}\left\|\hat{X}_t^{(j)} - X_t^{(j)}\right\|_2. \tag{4.2}$$

While both of these metrics can provide some intuition as to the overall reconstruction performance of our model across the test set, they do not necessarily provide direct information about how well key-frames are placed temporally. Our initial intuition is that the optimal key-frame placements should be related to the energy of the motion. Parts of a motion sequence with high energy due to large variations in movement could benefit from more key-frames being allocated in that region. To validate our intuition, we propose a series of qualitative analyses to better interpret the model's learned placement of key-frames.

We choose to interpret the velocity and acceleration of motions as an intuitive and highly correlated proxy for a motion's energy. Due to our choice of interpolator being linear, we can only expect to successfully reconstruct a limited range of the ground truth motion velocities. There are two ways we can increase the range of modelled velocities. The first is through increasing the key-frame sample rate, which increases the amount of non-linearly generated poses, the only source of velocity variation. The second, we hypothesize, is through irregular spacing of key-frames by learning to place them around regions of velocity variation. To validate this, we plot a velocity histogram over the entire test sets of both MotionSynth and LAFAN1 and compare them with the reconstructed test set velocity histogram using KL-Divergence. We construct our histograms by computing velocities across individual joints per test set motion for both reconstructed and ground truth motions. We then bin the per-frame velocities across 200 bins and compute the KL-Divergence:

$$D_{\mathrm{KL}}(P\|Q) = \sum_{x} P(x) \log\left(\frac{P\left(x\right)}{Q\left(x\right)}\right), \tag{4.3}$$

where $P(x)$ is our histogram of ground truth velocities and $Q(x)$ is our histogram of reconstructed velocities. We only compute and bin velocities for extremity joints, specifically the elbow, hand, knee, and heel.

4.5 Quantitative results

In order to determine the efficacy of our key-framing transformer, we compare our key-framing approach within an alternative sequence modelling architecture, we compare continuous versus discrete latent key-frame representations, and we compare our learned key-frame placements against uniform key-frame spacing. In Table 4.1 we see that our cross-attention transformer outperforms our 1D CNN baseline. Attention as a powerful tool for sequence modelling has been well documented. We see this as a positive indication that our specific use case of alternating cross-attention and self-attention for key-frame selection and temporal downsampling is beneficial. The continuous autoencoding variant of our key-framing transformer outperforms our complete VQ approach on MotionSynth by a small amount. As we scale to a more complex dataset like LAFAN1, continuous autoencoding outperforms VQ by a larger extent.

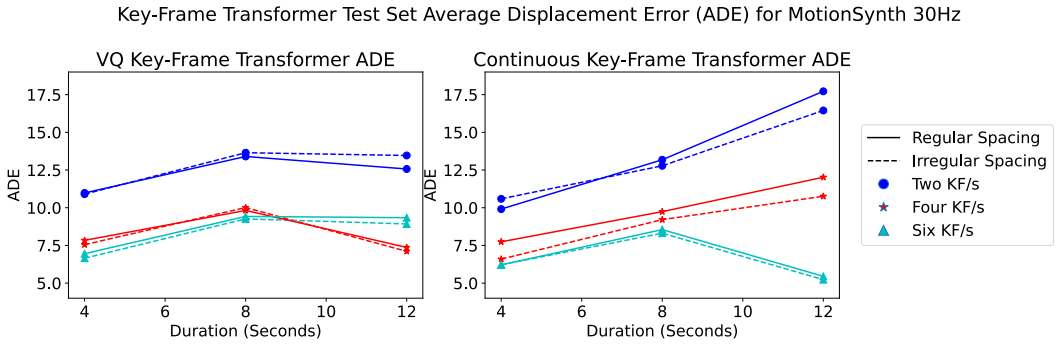
To validate the effectiveness of our irregular key-framing scheme, we plot the trends of our test set error metrics ADE and MPJPE across increasing reconstruction durations and increasing key-frame frequencies on both MotionSynth and LAFAN1 datasets. Figure 4.1 demonstrates that apart from the two kf/s variant, our VQ irregular key-framing approach seems to be marginally better than the regular spaced variant in most cases at 30 Hz. This trend holds when we move on to the LAFAN1 dataset as evident in Fig. 4.3, which is

Model	MotionSynth 4s				LAFAN1 4s			
	ADE ↓		MPJPE ↓		ADE ↓		MPJPE ↓	
	Reg	Irreg	Reg	Irreg	Reg	Irreg	Reg	Irreg
1DCNN	9.49	9.60	1.35	1.37	32.55	32.75	5.10	5.18
Ours (AE)	7.74	6.59	1.07	0.91	17.08	16.56	2.73	2.64
Ours (VQ)	7.83	7.55	1.09	1.05	30.78	30.40	4.81	4.71

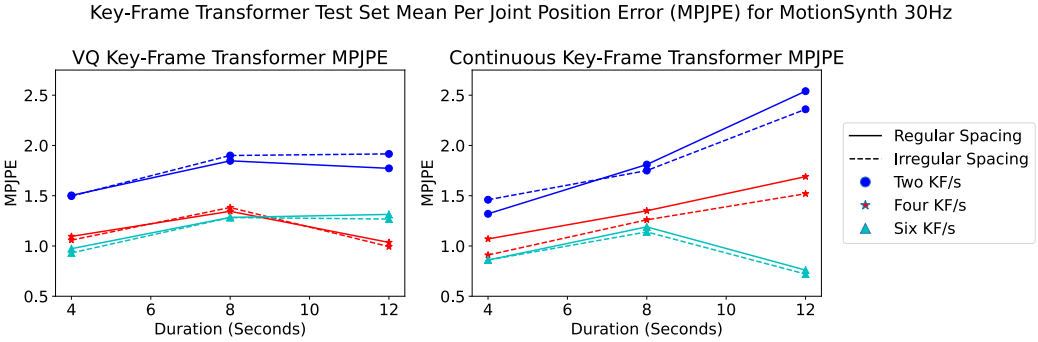
Table 4.1: Quantitative comparison of baselines for motions up to four seconds in the MotionSynth and LAFAN1 datasets. We compare a 1D CNN-based VQVAE (1D CNN) and our cross-attention-based transformer architecture without vector quantization (Ours AE) and with vector quantization (Ours VQ). The evaluation metrics include the Average Displacement Error (ADE) and Mean Per Joint Position Error (MPJPE) for both naive regular key-frame spacing (Reg) and learned irregular key-frame spacing (Irreg).

promising considering the more challenging nature of the dataset. When training on the MotionSynth dataset at 120 Hz, apart from four kf/s, regular spacing seems to reconstruct motions more effectively for VQ based key-framing in Fig. 4.2. Our irregular key-framing approach outperforms regular spacing by the largest margin when we use our key-framing transformer with a continuous latent space as evident in Fig. 4.3. This holds across each modelled duration and key-frame sample rate on LAFAN1, and in most conditions for MotionSynth. In most situations, the continuous autoencoding version of our key-framing transformer outperforms the discrete VQ variant.

Across both datasets, a minimum of four key-frames per second is required to achieve a decent reconstruction accuracy due to preserving sufficient non-linear elements of the ground-truth motion sequence. Due to our choice of linear interpolator, two key-frames per second are simply insufficient to capture variations in velocity, specifically velocity peaks, while simultaneously capturing the average velocity for that motion. At low key-

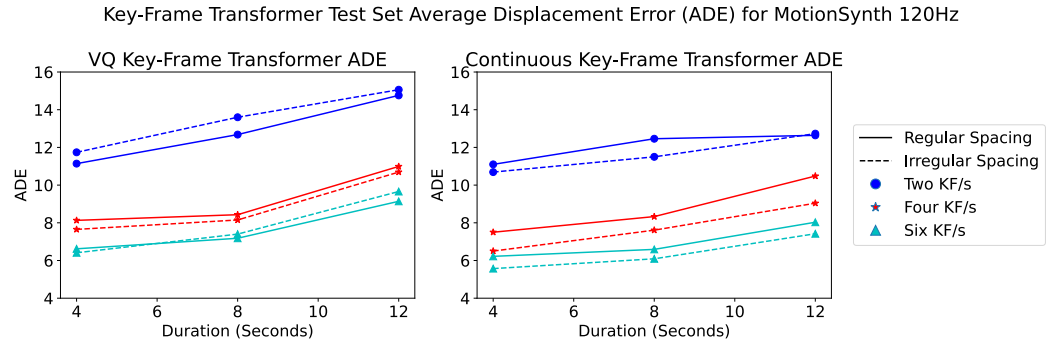


(a) Test set ADE of our key-framing transformer architecture with regular and irregular spacing trained on MotionSynth downsampled to 30 Hz

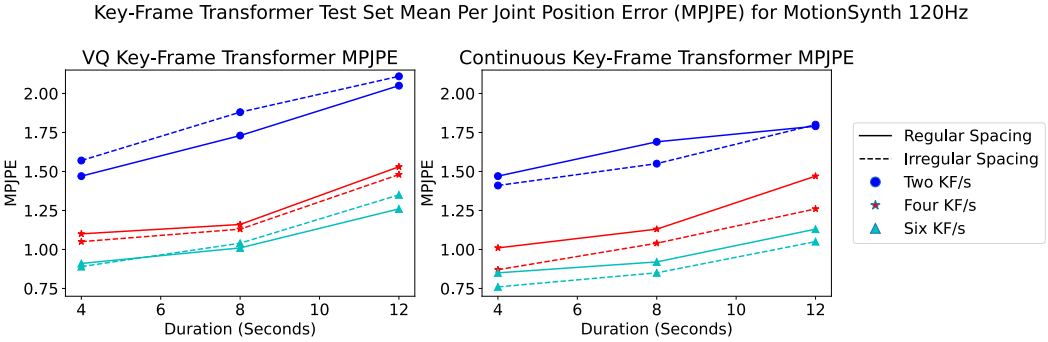


(b) Test set MPJPE of our key-framing transformer architecture with regular and irregular spacing trained on MotionSynth downsampled to 30 Hz

Figure 4.1: Average Displacement Error (Top) and Mean Per Joint Position Error (Bottom) of our cross-attention based transformer approach with and without vector quantization trained to reconstruct three different durations of motions on the MotionSynth dataset downsampled to 30 Hz. We compare regularly spaced and irregularly spaced variations of the model denoted by solid and dashed lines respectively.

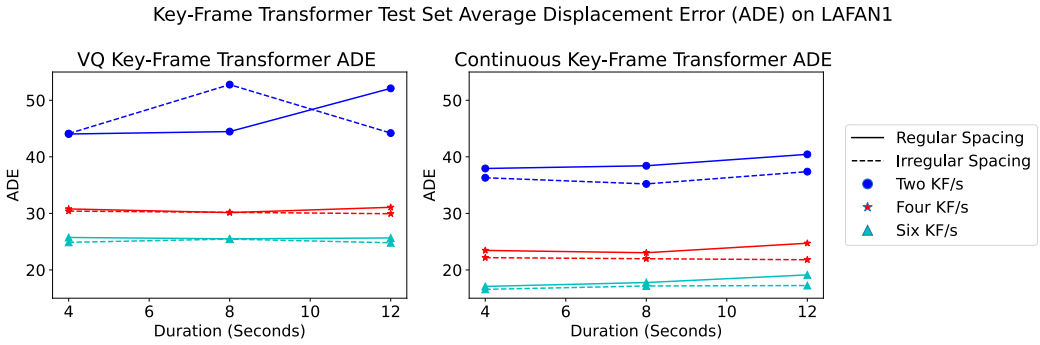


(a) Test set ADE of our complete key-framing transformer architecture with regular and irregular spacing trained on MotionSynth 120 Hz

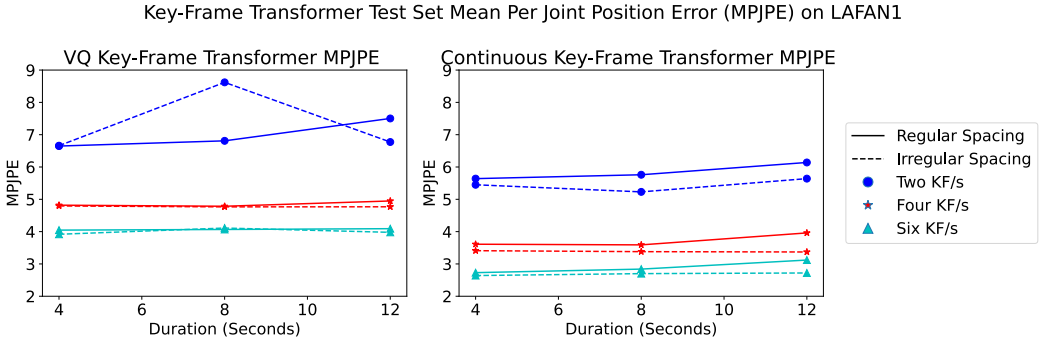


(b) Test set MPJPE of our complete key-framing transformer architecture with regular and irregular spacing trained on MotionSynth 120 Hz

Figure 4.2: Average Displacement Error (Top) and Mean Per Joint Position Error (Bottom) of our cross-attention based transformer approach with and without vector quantization trained to reconstruct three different durations of motions on the MotionSynth dataset at the original 120 Hz. We compare regularly spaced and irregularly spaced variations of the model denoted by solid and dashed lines respectively.



(a) Test set ADE of our key-framing transformer architecture with regular and irregular spacing trained on LAFAN1.



(b) Test set MPJPE of our key-framing transformer architecture with regular and irregular spacing trained on LAFAN1.

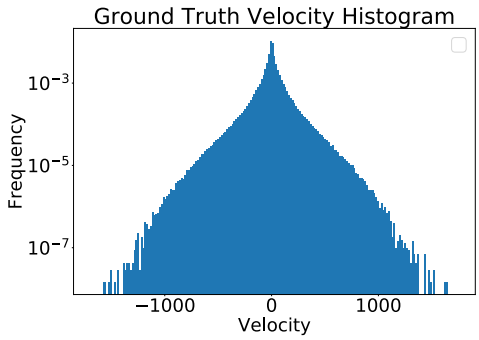
Figure 4.3: Average Displacement Error (Top) and Mean Per Joint Position Error (Bottom) of our cross-attention based transformer approach with and without vector quantization trained to reconstruct three different durations of motions on the LAFAN1 dataset. Our irregular spacing approach outperforms regular spacing when our key-framing transformer is implemented as a continuous autoencoder and at longer durations.

frame frequencies, the network is forced to only model the average velocities missing high frequency components due to this constraint.

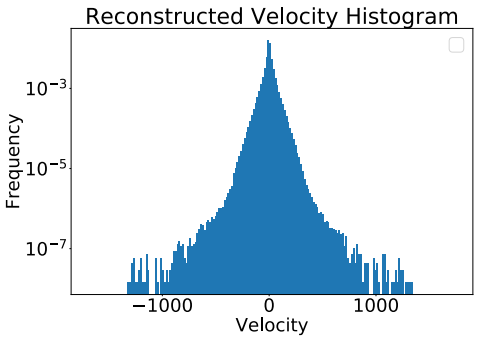
4.6 Qualitative results

Our current metrics model reconstruction error averaged across the entire test set. Due to the marginal improvements that we see in these quantities with VQ based key-framing, we further explore the impact of choosing key-frame placement on capturing variations in velocity. Specifically, we visualize the velocities of the test set ground truth motion as a histogram and see how closely we can capture this velocity distribution via reconstructing test set motions with both regular and irregular key-framing. Figure 4.4 visualizes the velocity histogram for a four second motion modelling two key-frames per second. When key-framing with regular spacing, the model struggles to capture the long-tail of the velocity distribution. Modelling the same number of key-frames, but with irregular spacing more effectively captures the distribution tail.

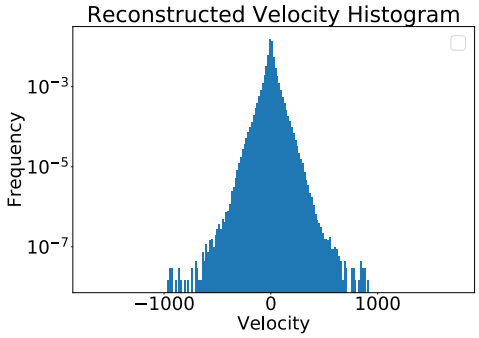
As we increase the number of key-frames we model, the range of velocities increases for both regular and irregular spacing. In the regularly spaced case, we now have more chances for key-frame poses to land close to high velocity regions. Although, even at four key-frames per second, Fig. 4.5 shows that we still are able to model the long-tail for angular velocity more effectively with irregular spacing. By increasing the number of key-frames, the qualitative differences between distributions begin to diminish. Figure 4.6 shows the KL-Divergence between velocity distributions at each of our modelled durations and key-frame frequencies for MotionSynth. In most cases, especially at 120 Hz,



(a) Ground truth test-set velocity histogram.

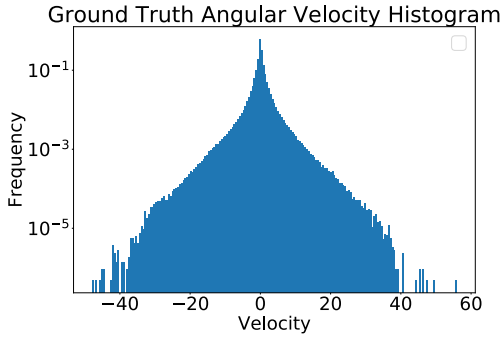


(b) Test-set velocity histogram of reconstructed motions with irregularly spaced key-frames.

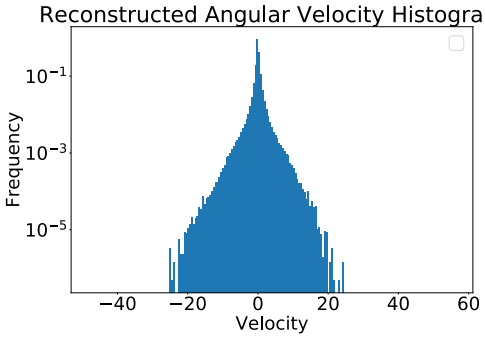


(c) Test-set velocity histogram of reconstructed motions with regularly spaced key-frames.

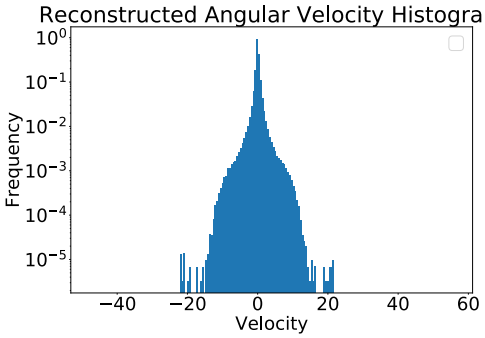
Figure 4.4: A comparison of log-scale test-set velocity histograms computed via finite difference of extremity joints. Irregular spacing and Regular spacing are trained to reconstruct two key-frames per second for a four second motion. At low key-frame rates regular spacing struggles to capture the long-tails of the ground truth velocity histograms. Irregular spacing is better able to capture these long-tails which is reflected by the lower KL-Divergence score.



(a) Ground truth test-set velocity histogram.

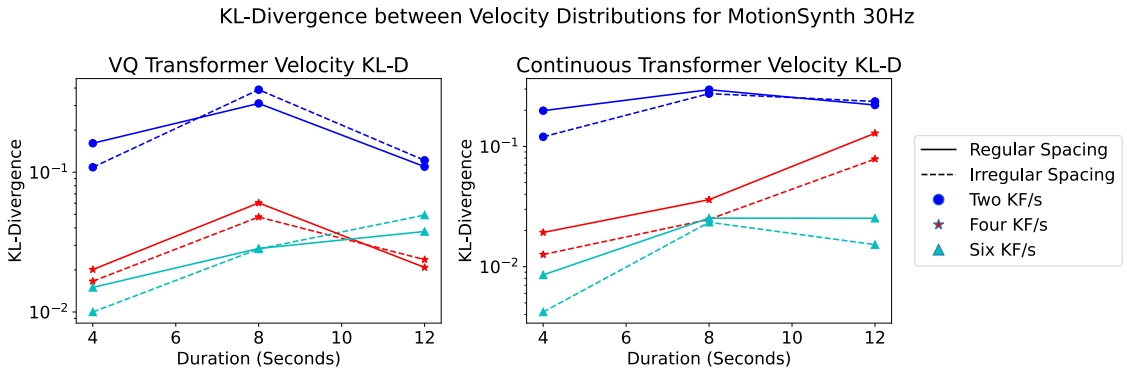


(b) Test-set velocity histogram of reconstructed motions with irregularly spaced key-frames.

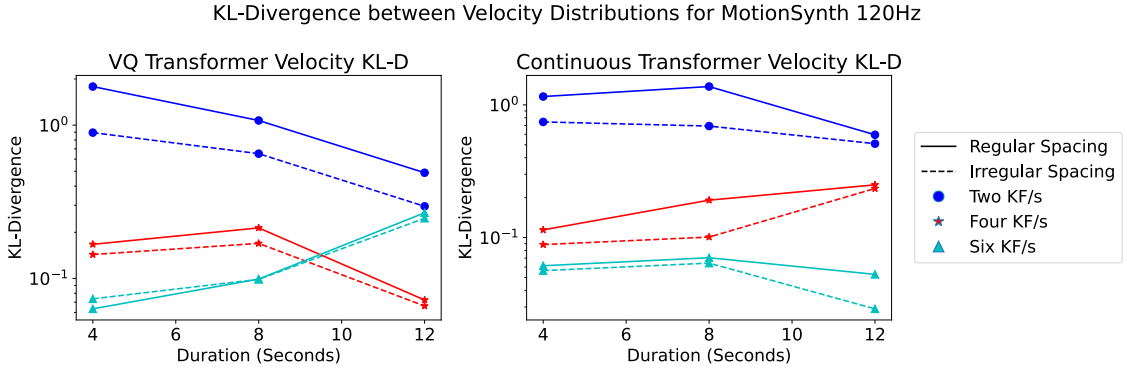


(c) Test-set velocity histogram of reconstructed motions with regularly spaced key-frames.

Figure 4.5: A comparison of log-scale test-set angular velocity histograms. Irregular spacing and Regular spacing are trained to reconstruct four key-frames per second for a four second motion. Regular spacing continues to struggle to capture the long-tails of the ground truth velocity histograms even with more key-frames. Irregular spacing once more has a lower KL-Divergence with the ground truth histogram.



(a) KL-Divergence comparison between discrete and continuous key-framing transformer for MotionSynth downsampled to 30 Hz



(b) KL-Divergence comparison between discrete and continuous key-framing transformer for MotionSynth at 120 Hz

Figure 4.6: We produce velocity histograms for regular and irregular key-frame-based reconstruction at increasing motion durations and key-frame rates for the MotionSynth dataset at 30 Hz and 120 Hz. The KL-Divergence is not as consistently better with irregular spacing on the MotionSynth dataset at 30 Hz in the discrete case, however we do see improvement across each key-frame sample rate and duration at 120 Hz. In our continuous variant, we see improvement with irregular spacing across each key-frame sample rate, duration, and frame rate.

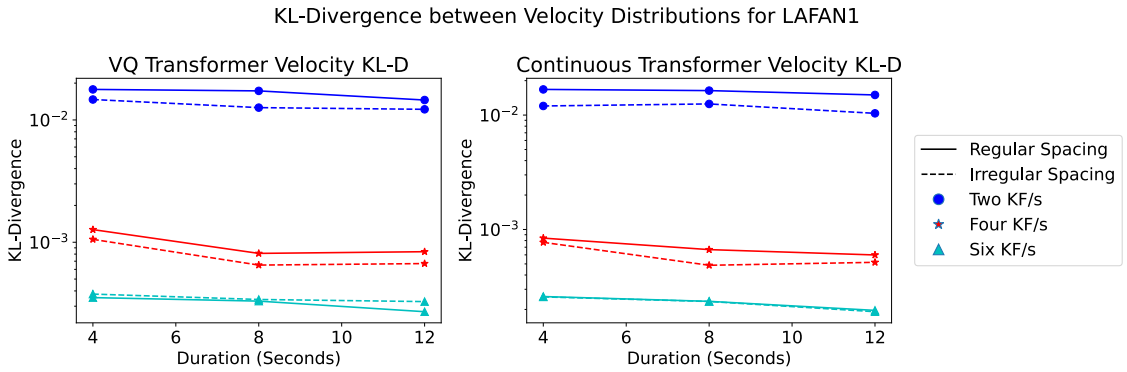


Figure 4.7: KL-Divergence between ground truth velocity histogram and reconstructed test-set motion histogram on the LAFAN1 dataset for increasing durations and key-frame sample rates. We compare both the discrete and continuous variants of our key-framing transformer architecture. We see that our irregular key-framing method consistently models the test-set velocity distribution more closely than regular spacing measured by KL-Divergence for the LAFAN1 dataset.

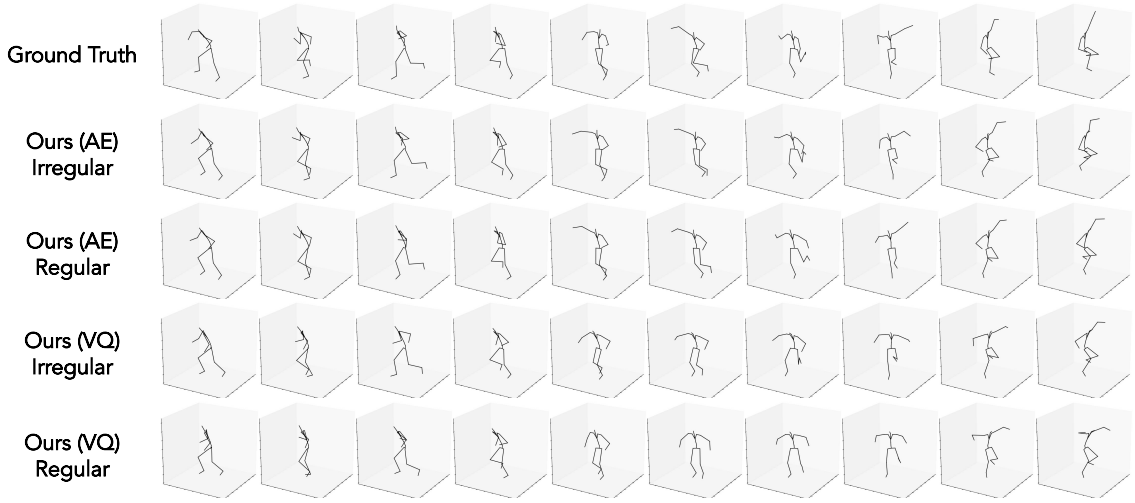


Figure 4.8: A test-set motion sample from the LAFAN1 dataset. We compare our continuous and VQ-based key-framing approach trained to reconstruct an eight-second motion with four key-frames per second. For both our continuous and VQ-based models, we present the reconstructed motion with learned irregular spacing as well as regular spacing.

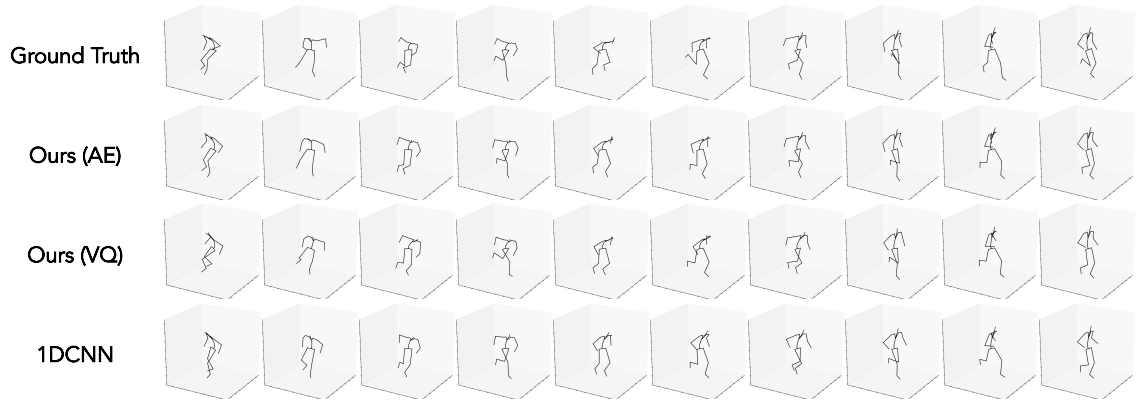


Figure 4.9: A 12-second test-set motion sample from the LAFAN1 dataset reconstructed at four kf/s. We compare each of our model classes with learned irregular spacing, including continuous attention-based key-framing, VQ attention-based key-framing, and 1DCNN baseline.

both VQ-based and continuous key-framing produce velocity distributions that are closer to the ground truth. For the LAFAN1 dataset, which contains the most energetic motions, Fig. 4.7 shows that irregular spacing of key-frames still produces velocity distributions that are closer to the ground truth for both VQ-based and continuous key-framing. The KL-Divergence becomes negligible as we reach our maximum number of key-frames. This trend is less evident in the MotionSynth dataset containing simpler motions and a smaller long-tail at 30 Hz. This implies that our irregular spacing approach is likely more effective when modelling motions with higher velocities. Figure 4.8 visualizes a test-set motion sample reconstructed with our attention-based key-framing approach with both irregular and regular spacing. Additionally, in Fig. 4.9, we visualize a test-set motion sample between our baseline 1DCNN and attention-based architecture with irregular spacing. Our continuous-autoencoding key-framing approach more closely reconstructs the ground truth motion with learned irregular spacing in both samples. Additional motion samples

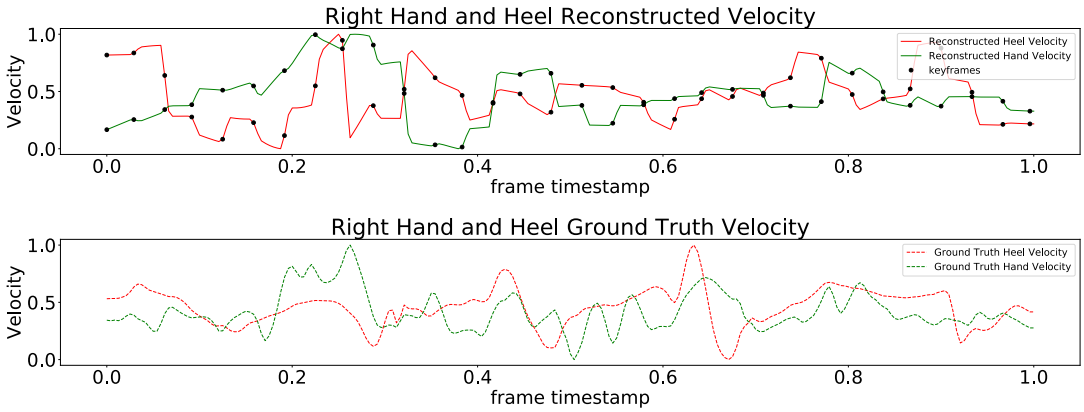


Figure 4.10: A challenge when representing an entire skeleton with a single key-frame is when different joints have different phases. We show the reconstructed velocities (top) of the heel (red) and hand (green) joints with associated key-frame placements (black) along with the ground truth velocities (bottom) for both joints.

are presented in the appendix.

4.7 Discussion

In the continuous autoencoding case, Fig. 4.1 and 4.3 shows that our irregular key-framing approach does improve over regular spacing across our quantitative metrics on the more challenging LAFAN1 dataset and in most conditions for MotionSynth. Additionally, when we compare our vector quantized key-framing transformer with our continuous baseline, continuous autoencoding outperforms our quantized autoencoder. It is important to note that one of the primary motivations for modelling a discrete latent space is to simplify downstream tasks like prediction. The difference is the need to predict continuous latent vectors in the case of continuous autoencoding versus a key-frame number of codebook indices represented as integers in the case of discretization using VQ. These indices could

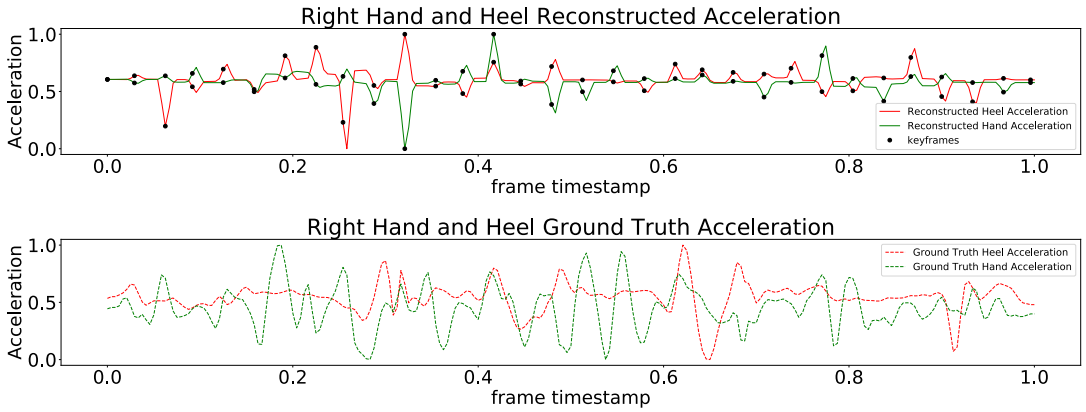


Figure 4.11: Visualization of the reconstructed acceleration of the same motion from 4.10. With linear interpolation, the only points where a change in velocity is possible is at a key-frame. Therefore, by modelling a whole pose with a single key-frame, the change in velocity across all joints is forced to occur at the same time.

be predicted as categorical distributions to preserve multi-modality. In our current continuous formulation, multi-modality is not as easily preserved if extended to prediction. The tradeoff for better reconstruction accuracy for the flexibility and simplicity of a discrete codebook should be considered.

Figures 4.7 and 4.6 show that variations in key-frame placements can help model a larger range of velocities present in ground truth motion samples. Consider for example a cyclic motion of an individual running. Each interpolated frame can be expected to have close to a constant velocity. When key-frames are regularly spaced, this constant velocity even between interpolated frames between two sets of different key-frames is more likely to be similar than if the key-frames were spaced irregularly. Regularly spaced key-frames would land at the same parts of different periods within the cycle while irregular spacing could potentially capture different parts of the cycle altogether. Since irregular spacing produces velocity distributions closer to the ground truth across both datasets, this is a

positive indication that our irregular spacing formulation successfully places key-frames at points where there is high variance in velocity. Due to our formulation producing a majority of poses through linear interpolation, it is beneficial to place key-frame poses, the only source of non-linearity, at points of high velocity variance. This is as opposed to regular spacing, in which regions with high velocity variance will not be captured unless they happen to roughly land at regularly spaced intervals.

While our velocity distributions show that there is some benefit to learning key-frame placement in an irregular manner, the benefits are perhaps not as significant as we may expect, especially for VQ-based key-framing. In Fig. 4.10, the heel joint seems to be more optimally reconstructed at the expense of the hand joints. This could mean that the network is biased towards key-frame placements that correspond to joints that contribute more towards the overall position loss. Although according to Fig. 4.12, our model seems to explore variations in key-frame placements, it is possible that finding the optimal placement when modelling all joints with a single set of key-frames is more challenging for our current formulation. Linear interpolation enforces constant velocity for interpolated frames between any two sets of key-frames. With this constraint, incorporating a strong learning signal based on a velocity reconstruction objective is infeasible. The only learning signal is implicitly through our position loss, without a strong learning signal for velocity, the network may further struggle to place optimal key-frames. This limitation is further demonstrated in our reconstructed accelerations in Fig. 4.11, changes in velocity can only be captured at the same time across all joints, even if optimal key-frame placement between joints may differ.

Similar to a Taylor series approximation, where a summation of non-linear polynomi-

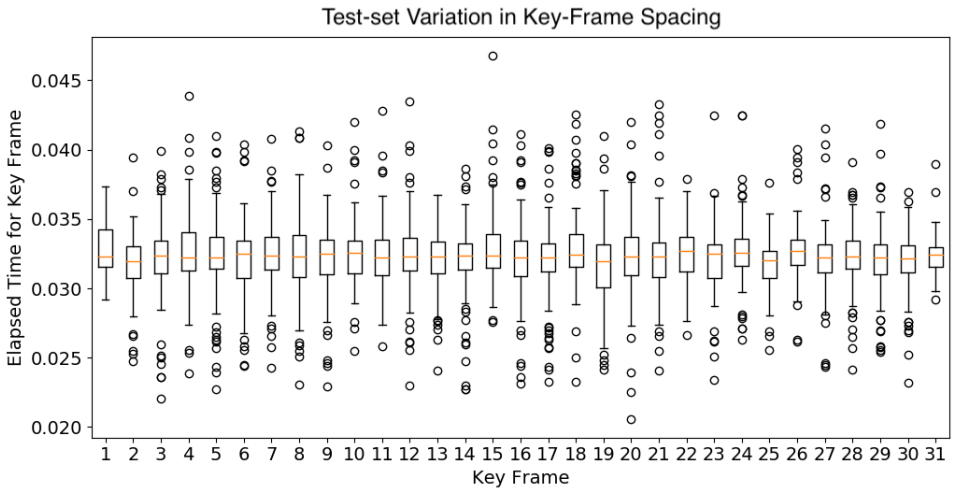
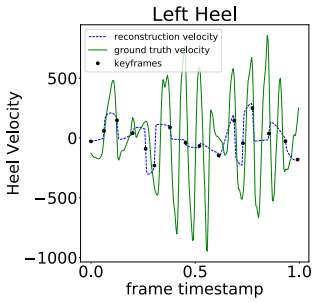
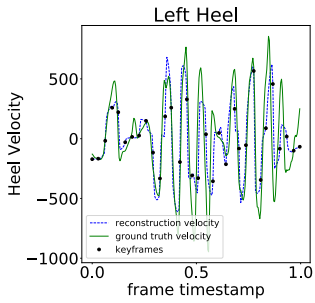


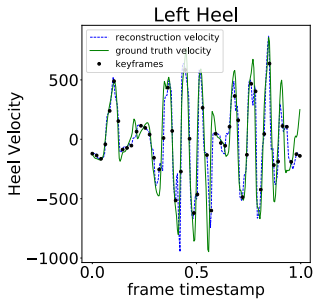
Figure 4.12: Box-plot of the variation in learned key-frame spacing for reconstructing test-set motions. This example is from our cross-attention key-framing transformer trained to reconstruct an eight second motion at four key-frames per second. Our representation of time elapsed is between 0 and 1, to determine the number of frames we multiply the elapsed time by the number of frames in the motion. For an eight second motion at 30 Hz, the key-frame spacings are between 4 to 10 frames.



(a) Two kf/s reconstructed (blue) and ground truth (green) velocity.



(b) Four kf/s reconstructed (blue) and ground truth (green) velocity.



(c) Six kf/s reconstructed (blue) and ground truth (green) velocity.

Figure 4.13: A visualization of reconstructed vs ground truth velocities of the heel joint for a test set motion using models trained with two, four, and six key-frames per second with irregular spacing. As we increase the number of key-frames we are able to better reconstruct velocity peaks.

als can be used to locally approximate a more complicated, non-polynomial function, in our current formulation the addition of each new key-frame can be viewed as a non-linear local approximation of the original motion sequence. At low key-frame sample rates, each key-frame is responsible for approximating a larger local window of the signal, in which case the converged solution tends to be the average of this region. The averaging effect at low key-frame sample rates can be seen in Fig. 4.13. At two kf/s, we can only reconstruct the average velocities, at four kf/s we can better reconstruct local regions around velocity peaks. As we increase the key-frame sample rate, each key-frame can be responsible for approximating an even more local region of the ground truth signal. Depending on the complexity of said signal, the number of key-frames to sufficiently reconstruct a local region could be as low as six kf/s. In our formulation, perhaps with enough key-frames the local regions are so well approximated with uniform spacing that learning the placement of key-frames does not improve the overall reconstruction accuracy as strongly, resulting in a weaker training signal for our self-supervised irregular spacing scheme.

4.8 Summary

In this chapter, we presented a series of quantitative and qualitative metrics for evaluating our attention-based approach for subsampling a motion into a set of key-frames and their placements in time. Our selection of baselines compare attention-based key-framing with an alternative sequence modelling choice, continuous versus discrete latent key-frame representations, and lastly learned irregular key-frame placements with naive uniform spacing. According to our quantitative metrics, attention-based key-framing outperforms our

Model	FPS	KF Freq.	4s		8s		12s	
			Reg	Irreg	Reg	Irreg	Reg	Irreg
1DCNN	30 Hz	2 kf/s	11.79	12.72	16.09	16.81	21.34	21.59
		4 kf/s	9.49	9.60	12.63	13.42	19.09	18.99
		6 kf/s	8.85	8.82	13.00	12.87	17.71	17.12
	120 Hz	2 kf/s	13.57	12.14	14.91	14.83	15.99	16.22
		4 kf/s	8.77	8.44	11.32	11.44	13.04	13.17
		6 kf/s	8.51	9.11	10.67	10.92	12.94	13.60
Ours (AE)	30 Hz	2 kf/s	9.91	10.59	13.18	12.78	17.72	16.45
		4 kf/s	7.74	6.59	9.74	9.21	12.02	10.76
		6 kf/s	6.21	6.22	8.55	8.31	5.45	5.23
	120 Hz	2 kf/s	11.10	10.69	12.46	11.5	12.64	12.73
		4 kf/s	7.50	6.50	8.33	7.61	10.48	9.04
		6 kf/s	6.22	5.57	6.59	6.09	8.03	7.42
Ours (VQ)	30 Hz	2 kf/s	10.98	10.91	13.39	13.64	12.56	13.46
		4 kf/s	7.83	7.55	9.81	10.01	7.36	7.10
		6 kf/s	6.95	6.65	9.42	9.26	9.33	8.92
	120 Hz	2 kf/s	11.14	11.74	12.68	13.6	14.76	15.06
		4 kf/s	8.13	7.65	8.43	8.15	10.99	10.69
		6 kf/s	6.62	6.41	7.18	7.40	9.14	9.66

Table 4.2: Average Displacement Error (ADE) across all joints averaged over all test set motion reconstructions from the MotionSynth dataset at the original 120 Hz and downsampled to 30 Hz. We explore varying the number of key-frames used for reconstruction, the duration of the motion to reconstruct, and irregular versus regular spacing variations of each model. Specifically, we examine the 1DCNN VQVAE (top), a continuous autoencoding cross-attention transformer (middle), and a VQVAE cross-attention transformer (bottom). For each modelled duration and key-frame sample rate, we highlight the best-performing reconstruction model between architectures and key-frame spacing method.

Model	FPS	KF Freq.	4s		8s		12s	
			Reg	Irreg	Reg	Irreg	Reg	Irreg
1DCNN	30 Hz	2 kf/s	1.65	1.81	2.29	2.43	3.16	3.20
		4 kf/s	1.35	1.37	1.78	1.92	2.83	2.81
		6 kf/s	1.27	1.27	1.85	1.85	2.60	2.48
	120 Hz	2 kf/s	1.85	1.64	1.98	2.10	2.33	2.36
		4 kf/s	1.22	1.17	1.61	1.64	1.90	1.92
		6 kf/s	1.19	1.28	1.52	1.56	1.89	1.99
Ours (AE)	30 Hz	2 kf/s	1.32	1.46	1.81	1.75	2.54	2.36
		4 kf/s	1.07	0.91	1.35	1.26	1.69	1.52
		6 kf/s	0.86	0.86	1.19	1.14	0.76	0.72
	120 Hz	2 kf/s	1.47	1.41	1.69	1.55	1.79	1.80
		4 kf/s	1.01	0.87	1.13	1.04	1.47	1.26
		6 kf/s	0.85	0.76	0.92	0.85	1.13	1.05
Ours (VQ)	30 Hz	2 kf/s	1.50	1.49	1.84	1.90	1.77	1.91
		4 kf/s	1.09	1.05	1.34	1.38	1.03	0.99
		6 kf/s	0.97	0.93	1.28	1.28	1.31	1.26
	120 Hz	2 kf/s	1.47	1.57	1.73	1.88	2.05	2.11
		4 kf/s	1.10	1.05	1.16	1.13	1.53	1.48
		6 kf/s	0.91	0.89	1.01	1.04	1.26	1.35

Table 4.3: Mean Per Joint Position Error (MPJPE) over test set reconstructions from the MotionSynth dataset at the original 120 Hz and downsampled to 30 Hz. For each modelled duration and key-frame sample rate, we highlight the best-performing reconstruction model between architectures and key-frame spacing method. The continuous autoencoding variant of our cross-attention key-framing transformer has the lowest MPJPE across increasing motion durations and key-frame sample rates.

Model	KF Freq.	4s		8s		12s	
		Reg	Irreg	Reg	Irreg	Reg	Irreg
1DCNN	2 kf/s	53.53	46.55	47.27	54.85	50.55	49.86
	4 kf/s	32.55	32.75	35.09	33.42	37.73	35.80
	6 kf/s	27.10	27.93	29.47	31.49	31.60	31.35
Ours (AE)	2 kf/s	37.94	36.30	38.42	35.21	40.44	37.38
	4 kf/s	23.45	22.17	23.04	21.98	24.72	21.80
	6 kf/s	17.08	16.56	17.77	17.16	19.12	17.24
Ours (VQ)	2 kf/s	44.03	44.11	44.46	52.77	52.10	44.20
	4 kf/s	30.78	30.40	30.13	30.18	31.07	29.84
	6 kf/s	25.75	24.87	25.49	25.44	25.63	24.80

Table 4.4: Average Displacement Error (ADE) across all joints averaged over all test set motion reconstructions from the LAFAN1 dataset. We compare against a 1DCNN VQ-VAE (top), a continuous autoencoding cross-attention key-framing transformer (middle), and our quantized cross-attention key-framing transformer (bottom).

Model	KF Freq.	4s		8s		12s	
		Reg	Irreg	Reg	Irreg	Reg	Irreg
1DCNN	2 kf/s	7.80	7.21	7.35	9.04	7.97	7.90
	4 kf/s	5.10	5.18	5.61	5.32	6.07	5.75
	6 kf/s	4.31	4.45	4.74	5.07	5.08	5.03
Ours (AE)	2 kf/s	5.64	5.45	5.76	5.23	6.14	5.64
	4 kf/s	3.61	3.41	3.59	3.38	3.96	3.37
	6 kf/s	2.73	2.64	2.84	2.70	3.12	2.72
Ours (VQ)	2 kf/s	6.64	6.65	6.80	8.61	7.50	6.77
	4 kf/s	4.81	4.79	4.78	4.76	4.94	4.76
	6 kf/s	4.04	3.91	4.06	4.11	4.09	3.97

Table 4.5: Mean Per Joint Position Error (MPJPE) over test set motion reconstructions from the LAFAN1 dataset. We compare our 1D CNN baseline (top) with our key-framing transformer that represents latent key-frames as both continuous vectors (middle) and discrete dictionary entries (bottom). For each modelled duration and key-frame sample rate, we highlight the best-performing method between architectures and key-frame spacing method.

1D CNN baseline, and our continuous latent key-frame representation outperforms discrete latent key-frames across a majority of baselines. It is also evident that these architectural choices allow irregular key-framing to outperform naive uniform spacing. Our qualitative metrics demonstrate that an advantage of learned key-frame placement is the ability to more effectively capture the long-tail of velocity ranges present in ground truth motion samples. This holds true for both continuous and discrete latent key-frame representations as measured by the KL-Divergence between ground truth and reconstructed velocity histograms. Ultimately, our choice of linear interpolation of full poses is a bottleneck for learning good key-frame placements. While linear interpolation enforces near-constant velocity between key-frames, modelling an entire pose with a single key-frame can make it challenging for the network to decide the optimal placement as defined by variations in velocity when different joints have velocity variations that occur at conflicting points in time.

Chapter 5

Conclusion and Future Work

In this thesis, we explored novel representations for encoding human motion sequences without the need for autoregressive methods based on the principles of key-framing and in-betweening. We construct a self-supervised learning objective to represent a human motion sequence in terms of only a small subset of poses, which we refer to as key-frames. The in-betweening is accomplished via linear interpolation to reconstruct an approximation of the original motion sequence based on our learned key-frames. We showed that a discretized latent representation based on Vector Quantized Variational Autoencoders is a good candidate for representing key-frame poses within a fixed size latent pose dictionary. This formulation naturally extends to downstream tasks, like motion prediction and retrieval, because we can now represent an encoded pose sequence as a series of integers that correspond to latent dictionary entries. Downstream tasks can now be trained on top of this discrete set of integers instead of a more challenging continuous latent space. However, for the task of reconstruction, our key-framing approach implemented in a continuous

autoencoding scheme gives the best quantitative results for irregular key-framing.

We introduced a cross-attention based transformer architecture that learns to sub-sample a motion sequence into a set of key-frames, the number of which is conveniently determined by the number of initialized query tokens. We showed that our cross-attention based key-framing transformer with a discretized latent space outperforms some of our baselines, including a 1D CNN VQVAE, while our continuous key-framing transformer outperforms all baselines. We also explored two methods for determining the placements of key-frames for reconstruction, one based on naively spacing key-frames evenly apart as well as placements learned directly from the cross-attention key-framing transformer in a self-supervised fashion. Quantitatively, our learned key-frame placements outperform regular spacing, especially in the continuous case. Lastly, we showed that the learned irregular key-frame spacing more effectively captures the long-tail of velocities, especially in the more challenging dataset.

Learning the optimal irregular placements of key-frames in our formulation is more challenging than initially expected. While we can intuitively say that the optimal key-frame placement occurs around the highest energy regions of a motion, the reality is that high energy regions occur at different times for different joints. The optimal key-frame placement when representing an entire skeleton with a single key-frame conflicts across joints, and the network may become biased to placing key-frames according to the joints that contribute most towards the position error like the heel joint. Additionally, if the optimal key-frame placement is based on variations in velocity, our choice of linear interpolator, which can only capture constant velocities between key-frames, may not provide a strong enough training signal to learn these placements according to our current self-

supervised formulation.

5.1 Broader impact

The rapid progress of generative modelling in the last few years raises important ethical concerns regarding their deployment in consumer products. A recurring theme in this line of work is the role of scaling data and compute for improving the representational capacity of existing models. The roll-out of models like stable diffusion [43] has led to widespread adoption, first in creative industries as novel tooling for artists and now to the general public. Yet, questions remain as to how much these models simply regurgitate training instances [45] and even perpetuate social biases that arise in web-scale datasets [8].

While our work does not directly deal with web-scale image data, pose sequences have been shown to encode gender information from 3D joint positions alone [47]. Given this information, it is critical for motion capture datasets to have an equal distribution of male and female subjects. In the case our datasets contain more subjects of a specific gender, it is possible that our method for representing pose sequences in terms of key-frames may be biased to more effectively represent the gender with more training samples at inference. While neither dataset we train on reveals the gender distribution of their subjects, this information should be sought out if we choose to deploy our approach.

Implementing generative models as tools for artists can help alleviate time-consuming and repetitive tasks that are an inevitable part of an artist's workflow. Our encoded key-frame representation has direct applications in motion retrieval systems and other tasks that can benefit from a temporally compressed representation of pose sequences. Currently, our

method does not synthesize new motion samples and instead summarizes existing ones. Therefore, our approach augments existing workflows without necessarily replacing them altogether. Beyond animation, our key-framing scheme can be used for surveillance applications to extend long-term tracking and motion prediction capabilities. As surveillance technologies improve, the capabilities of regimes seeking totalitarian control over the populace only increase.

5.2 Future work

To address our limitations, we propose a direction of future work to disentangle the skeleton from a single set of key-frames. One approach could be separating the upper and lower body with different learned discrete latent codebooks. We can then leverage two separate decoding streams which learn a unique set of key-frames and associated placements for the upper body, as well as a separate learned set of key-frames with placements for the lower body. The two sets of key-frames can then be interpolated independently and recombined to form the final reconstructed motion. Additionally, we can leverage non-linear interpolators that capture acceleration, or even interpolate directly in the latent space. Such architectural choices may provide a stronger learning signal for optimal key-frame placement. Nonetheless, it is evident from both our qualitative and quantitative results that a key-frame based representation for motion sequences can be effective for long-term temporal modelling beyond just four seconds.

One of the core motivations of our key-framing approach is the potential benefits of using this representation as a prior for downstream tasks like motion forecasting. With our

VQ-based method, multi-modality can be preserved for long-term motion prediction by learning to predict future motion sequences in terms of a categorical distribution over our pre-trained codebook indices representing latent key-frames. Long-term prediction is then cast as learning codebook indices represented as integers, instead of continuous values in a high-dimensional latent space, which can greatly reduce the complexity of the task. A similar framework has been used for image generation [14] and autoregressive motion prediction [36]. It is important to note that a parallel line of work leveraging continuous latent spaces as a prior for downstream image synthesis tasks has also been shown to achieve promising results [43]. Leveraging our latent key-frame prior learned as either a continuous latent space or discrete codebook for downstream long-term motion modelling tasks is now a promising line of future work.

Bibliography

[1] Chaitanya Ahuja and Louis-Philippe Morency. Language2Pose: Natural language grounded pose forecasting. In *Proceedings of the International Conference on 3D Vision (3DV)*, 2019. 22

[2] Emre Aksan, Manuel Kaufmann, and Otmar Hilliges. Structured prediction helps 3D human motion modelling. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2019. 19

[3] Sadegh Aliakbarian, Fatemeh Sadat Saleh, Mathieu Salzmann, Lars Petersson, and Stephen Gould. A stochastic conditioning scheme for diverse human motion prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 20

[4] David F. Armstrong and Sherman E. Wilcox. 5Grasping Language: Sign and the Evolution of Language. In *The Gestural Origin of Language*. 2007. 1

[5] Dzmitry Bahdanau, Kyung Hyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *ICLR*, 2015. 14

[6] Emad Barsoum, John Kender, and Zicheng Liu. HP-GAN: Probabilistic 3D human motion prediction via GAN. In *CVPRW*, 2018. 20

[7] Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013. 10

[8] Jannik Brinkmann, Paul Swoboda, and Christian Bartelt. A multidimensional analysis of social biases in vision transformers. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 64

[9] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 32, 37

[10] Hsu-kuang Chiu, Ehsan Adeli, Borui Wang, De-An Huang, and Juan Carlos Niebles. Action-agnostic human pose forecasting. In *Proceedings of the IEEE Workshop on Applications of Computer Vision (WACV)*, 2019. 19

[11] Ginger Delmas, Philippe Weinzaepfel, Thomas Lucas, Francesc Moreno-Noguer, and Grégory Rogez. PoseScript: 3D human poses from natural language. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022. 22

[12] Christian Diller, Thomas Funkhouser, and Angela Dai. Forecasting actions and characteristic 3D poses. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 21, 35

[13] Andrea Dittadi, Sebastian Dziadzio, Darren Cosker, Ben Lundell, Tom Cashman, and Jamie Shotton. Full-body motion from a single head-mounted device: Generating SMPL poses from partial observations. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021. 2

[14] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2021. 66

[15] Katerina Fragkiadaki, Sergey Levine, Panna Felsen, and Jitendra Malik. Recurrent network models for human dynamics. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2015. 2, 19

[16] Saeed Ghorbani, Calden Wloka, Ali Etemad, Marcus A Brubaker, and Nikolaus F Troje. Probabilistic character motion synthesis using a hierarchical deep latent variable model. In *Computer Graphics Forum*, 2020. 22

[17] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Neural Information Processing Systems (NeurIPS)*, 2014. 1

[18] Ian J. Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, Cambridge, MA, USA, 2016. `http://www.deeplearningbook.org`. 1

[19] Anand Gopalakrishnan, Ankur Mali, Dan Kifer, Lee Giles, and Alexander G Ororbia. A neural temporal model for human motion prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 19

[20] Caglar Gulcehre, Kyunghyun Cho, Razvan Pascanu, and Yoshua Bengio. Learned-norm pooling for deep feedforward and recurrent neural networks. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases (ECML PKDD)*, 2014. 12

[21] Chuan Guo, Xinxin Zuo, Sen Wang, Shihao Zou, Qingyao Sun, Annan Deng, Minglun Gong, and Li Cheng. Action2Motion: Conditioned generation of 3D human motions. In *Proceedings of the ACM International Conference on Multimedia* , 2020. 1, 22

[22] Félix G. Harvey, Mike Yurick, Derek Nowrouzezahrai, and Christopher Pal. Robust motion in-betweening. *ACM Transactions on Graphics*, 2020. 5, 38

[23] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 1997. 12

[24] Daniel Holden, Jun Saito, Taku Komura, and Thomas Joyce. Learning motion manifolds with convolutional autoencoders. In *SIGGRAPH Asia 2015 Technical Briefs*, 2015. 35

[25] Daniel Holden, Jun Saito, and Taku Komura. A deep learning framework for character motion synthesis and editing. *ACM Transactions on Graphics*, 2016. 5, 38

[26] Daniel Holden, Oussama Kanoun, Maksym Perepichka, and Tiberiu Popa. Learned motion matching. *ACM Transactions on Graphics*, 2020. 25

[27] Andrew Jaegle, Felix Gimeno, Andy Brock, Oriol Vinyals, Andrew Zisserman, and

Joao Carreira. Perceiver: General perception with iterative attention. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2021. 32, 37

[28] Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2014. 7

[29] Muhammed Kocabas, Nikos Athanasiou, and Michael J Black. VIBE: Video inference for human body pose and shape estimation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020. 22

[30] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Neural Information Processing Systems (NeurIPS)*, 2012. 1

[31] John Lasseter. Principles of traditional animation applied to 3D computer animation. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, 1987. 3

[32] Yann Lecun and Yoshua Bengio. Convolutional networks for images, speech, and time-series. In *The handbook of brain theory and neural networks*. MIT Press, 1995. 12

[33] Hsin-Ying Lee, Xiaodong Yang, Ming-Yu Liu, Ting-Chun Wang, Yu-Ding Lu, Ming-Hsuan Yang, and Jan Kautz. Dancing to music. *Neural Information Processing Systems (NeurIPS)*, 2019. 22

[34] Ruilong Li, Shan Yang, David A Ross, and Angjoo Kanazawa. AI Choreographer:

Music conditioned 3D dance generation with AIST++. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021. 22

[35] Zimo Li, Yi Zhou, Shuangjiu Xiao, Chong He, Zeng Huang, and Hao Li. Auto-conditioned recurrent networks for extended complex human motion synthesis. *ICLR*, 2017. 2, 21

[36] Thomas Lucas, Fabien Baradel, Philippe Weinzaepfel, and Grégory Rogez. PoseGPT: quantization-based 3D human motion generation and forecasting. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2022. 21, 26, 66

[37] Wei Mao, Miaomiao Liu, and Mathieu Salzmann. Generating smooth pose sequences for diverse human motion prediction. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021. 20

[38] Julieta Martinez, Michael J Black, and Javier Romero. On human motion prediction using recurrent neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 2, 19

[39] Lea Müller, Vickie Ye, Georgios Pavlakos, Michael Black, and Angjoo Kanazawa. Generative Proxemics: A prior for 3D social interaction from images. *arXiv preprint arXiv:2306.09337*, 2023. 1

[40] Ferda Ofli, Rizwan Chaudhry, Gregorij Kurillo, René Vidal, and Ruzena Bajcsy. Berkeley MHAD: A comprehensive multimodal human action database. In *Proceedings of the IEEE Workshop on Applications of Computer Vision (WACV)*, 2013. 38

[41] Georgios Pavlakos, Vasileios Choutas, Nima Ghorbani, Timo Bolkart, Ahmed AA Osman, Dimitrios Tzionas, and Michael J Black. Expressive Body Capture: 3D hands, face, and body from a single image. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 1

[42] Justin M Power, Guido W Grimm, and Johann-Mattis List. Evolutionary dynamics in the dispersal of sign languages. *Royal Society Open Science*, 7(1):191100, 2020. 1

[43] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 64, 66

[44] David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 1986. 11

[45] Gowthami Somepalli, Vasu Singla, Micah Goldblum, Jonas Geiping, and Tom Goldstein. Diffusion art or digital forgery? Investigating data replication in diffusion models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2023. 64

[46] Oksana Tkachman, Gracellia Purnomo, and Bryan Gick. Repetition preferences in two-handed balanced signs: Vestigial locomotor central pattern generators shape sign language phonetics and phonology. *Frontiers in Communication*, 5:612973, 2021. 1

[47] Nikolaus Troje. Decomposing biological motion: A framework for analysis and synthesis of human gait patterns. *Journal of vision*, pages 371–87, 2002. 64

[48] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Neural Information Processing Systems (NeurIPS)*, 2017. 9, 25, 33

[49] 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 *Neural Information Processing Systems (NeurIPS)*, 2017. 14, 32, 76

[50] Jacob Walker, Kenneth Marino, Abhinav Gupta, and Martial Hebert. The Pose Knows: Video forecasting by generating pose futures. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2017. 1

[51] Borui Wang, Ehsan Adeli, Hsu-kuang Chiu, De-An Huang, and Juan Carlos Niebles. Imitation learning for human pose prediction. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2019. 19

[52] Yingying Wang and Michael Neff. Deep signatures for indexing and retrieval in large motion databases. In *Proceedings of the 8th ACM SIGGRAPH Conference on Motion in Games*, 2015. 2

[53] Julie R. Williamson, Joseph O'Hagan, John Alexis Guerra-Gomez, John H Williamson, Pablo Cesar, and David A. Shamma. Digital Proxemics: Designing social and collaborative interaction in virtual environments. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*, 2022. 1

[54] Shihong Xia, Congyi Wang, Jinxiang Chai, and Jessica Hodgins. Realtime style transfer for unlabeled heterogeneous human motion. *ACM Transactions on Graphics*, 2015. 38

[55] Xinchen Yan, Akash Rastogi, Ruben Villegas, Kalyan Sunkavalli, Eli Shechtman, Sunil Hadap, Ersin Yumer, and Honglak Lee. MT-VAE: Learning motion transformations to generate multimodal human dynamics. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018. 20

[56] Ye Yuan and Kris Kitani. DLow: Diversifying latent flows for diverse human motion prediction. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020. 20

[57] Jingxiao Zheng, Xinwei Shi, Alexander Gorban, Junhua Mao, Yang Song, Charles R Qi, Ting Liu, Visesh Chari, Andre Cornman, Yin Zhou, et al. Multi-modal 3D human pose estimation with 2D weak supervision in autonomous driving. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2

[58] Yi Zhou, Jingwan Lu, Connelly Barnes, Jimei Yang, Sitao Xiang, et al. Generative Tweening: Long-term inbetweening of 3D human motions. *arXiv preprint arXiv:2005.08891*, 2020. 2

[59] Silvia Zuffi, Angjoo Kanazawa, David W Jacobs, and Michael J Black. 3D Menagerie: Modeling the 3D shape and pose of animals. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 1

Appendix

A.1 Additional implementation details

As presented in Chapter 3 of the thesis, our key-framing transformer learns to temporally subsample a motion sequence into a set of learned key-frames and their associated placements in time. The first step in this encoding process is our skeleton embedding layer applied to the channel dimension of our input pose sequence. We choose to embed the skeleton joint rotations from their original $J \times 6$ rotation matrix representation to a channel size of 32. Where J is 22 for LAFAN1 and 26 for Motionsynth. Our learnable set of query tokens are initialized randomly from a Gaussian distribution. We choose to set the channel dimension of our query tokens to 32 as well. Our cross-attention blocks are implemented with a single layer containing a single head, while our self-attention blocks contain two layers with four heads per layer. We find experimentally that sharing weights between cross-attention blocks and self-attention blocks respectively outperforms having two sets of separate weights per block. Before an attention block, we add a channel-wise positional encoding [49] of alternating sinusoids of increasing frequency according to:

$$\text{PE}_{(pos,2i)} = \sin\left(\frac{pos}{10000^{(2i/d)}}\right)$$
$$\text{PE}_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{(2i/d)}}\right)$$

Here i is the channel dimension index, d is the size of the channel dimension, and pos is the position in the sequence. Our decoder architecture is a simple 1D CNN with two layers with a kernel size of 15 and stride 1. The convolutions operate on the channels of the encoded sequence. In our experiments, we observe that a 1D CNN decoder outperforms transformer-based decoders.

In irregular key-framing, a batch of learned key-frames can have different placements in time per motion sample. In order to efficiently interpolate a batch of motions given differing endpoints during training, we first compute the absolute time of each frame in the complete motion sequences under the constraint that the first and last frame occurs at $t = 0$ and $t = 1$ respectively. Given the absolute time of each frame, we can determine the left and right predicted key-frame via binary search against the predicted key-frame placement times, $t_{\mathcal{KF}}$. Then for every frame, we can determine where along the arc it lies between its left and right key-frame, and therefore determine the interpolated quaternion at that frame.

A.2 Entropy regularization

During training, our formulation for learning key-frame placements can be unstable. We notice experimentally that early in training the network has a tendency to converge on a

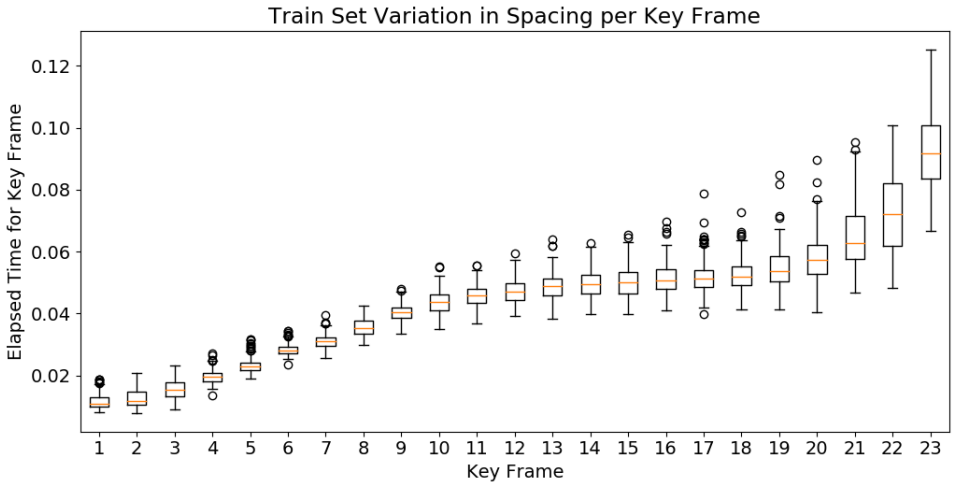


Figure A.1: Visualization of the variation in learned key-frame spacing without entropy regularization. Early in training the network can fall into a local minima such that some key-frames always have an elapsed time of zero.

local minima in which the elapsed time between the first few frames becomes close to zero, forcing the elapsed time between key-frames at other regions to be very far apart due to representing elapsed time as a distribution via a softmax operation. An example of this local minima is presented in Fig. A.1. In order to mitigate this, we employ an entropy regularization term in our training objective presented in Chapter 3 Eq. 3.8. Since our elapsed time between key-frames is represented as a distribution, high entropy is equivalent to uniform spacing between key-frames. Therefore early in training we maximize the entropy of the elapsed-time distribution and let the network learn relevant key-frame poses under uniform spacing. As this regularization term is decayed, the entropy is no longer maximized and the network starts to explore non-uniform key-frame placements.

	ADE ↓	
Model	**Reg**	**Irreg**
Ours (no regularizer)	——	17.49
Ours (VQ)	——	7.55

Table A.1: Comparison of Average Displacement Error (ADE) of our key-framing transformer with and without entropy regularization on the MotionSynth dataset trained to reconstruct four-second motions. Without entropy regularization, the model learns a degenerate solution in which the learned spacings between the first few key-frames are always near zero.

A.3 Additional samples

Additional reconstruction samples on the MotionSynth test-set at 30 Hz are shown in Figures A.2, A.3, and at 120Hz in Figure A.4. For the LAFAN1 test-set at 30Hz, some additional samples are shown in Figure A.5 and A.6. We include example velocity and corresponding acceleration reconstructions between the hand and heel joint in Figures A.7 and A.8, as well as Figures A.9 and A.10 respectively. We also visualize a subset of poses from reconstructed motions to compare irregular and regular spacing in Figure A.11 and against our baseline 1DCNN in Figure A.12.

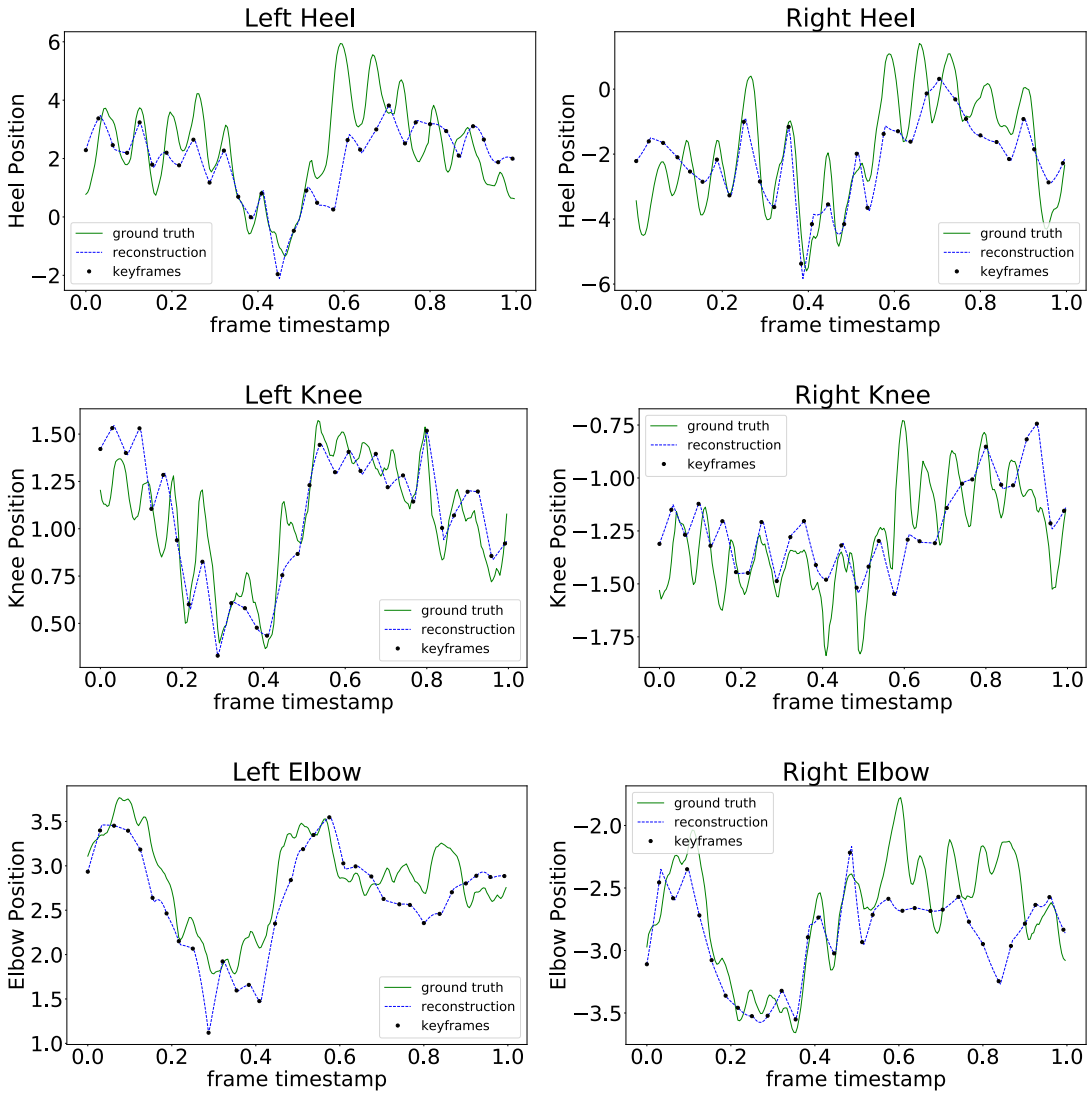


Figure A.2: MotionSynth 30 Hz reconstructed heel, knee, and elbow positions for an eight second motion at four kf/s with our continuous autoencoding key-framing transformer.

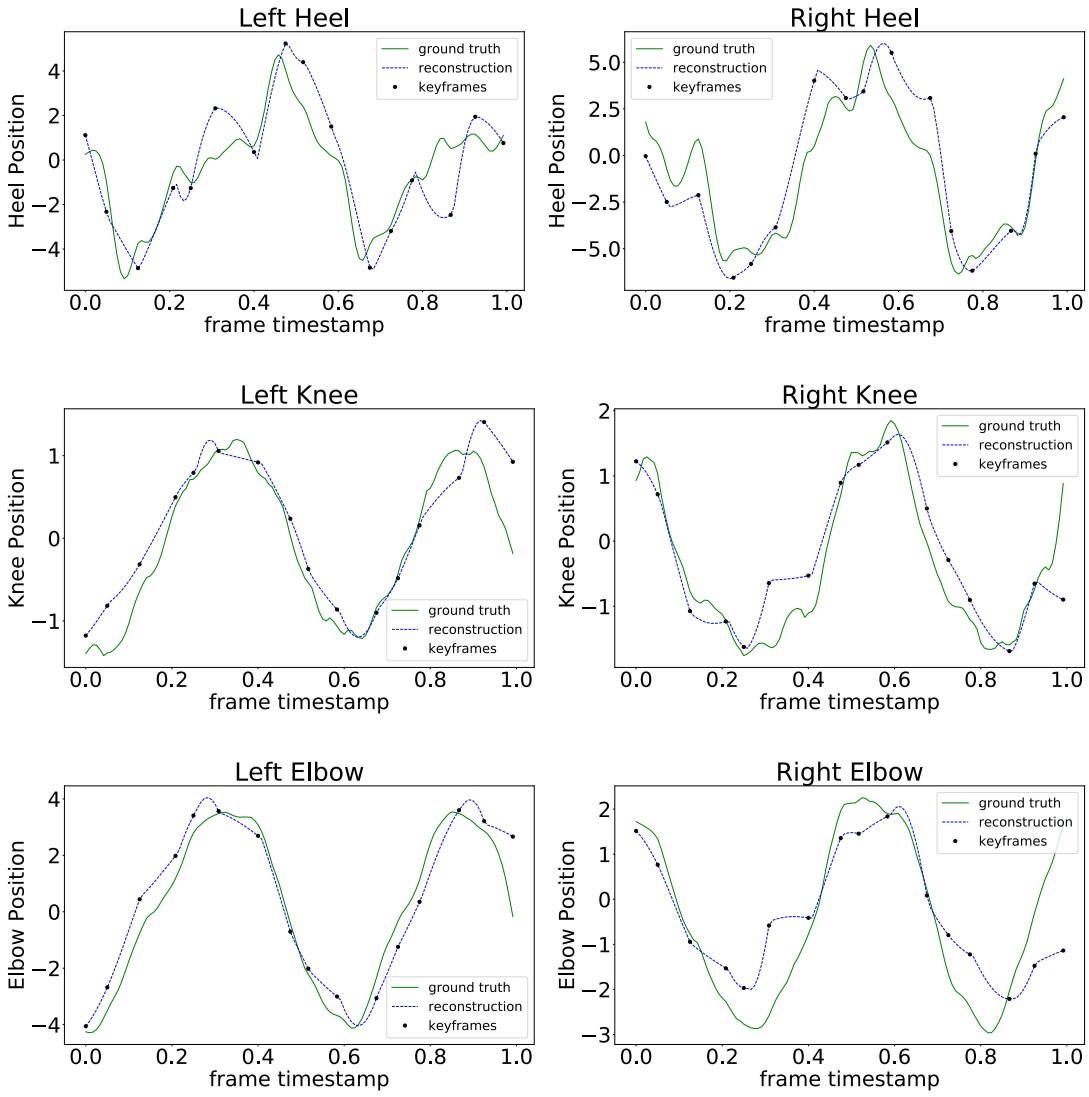


Figure A.3: MotionSynth 30 Hz reconstructed heel, knee, and elbow positions for a four second motion at four kf/s with our VQ-based key-framing transformer.

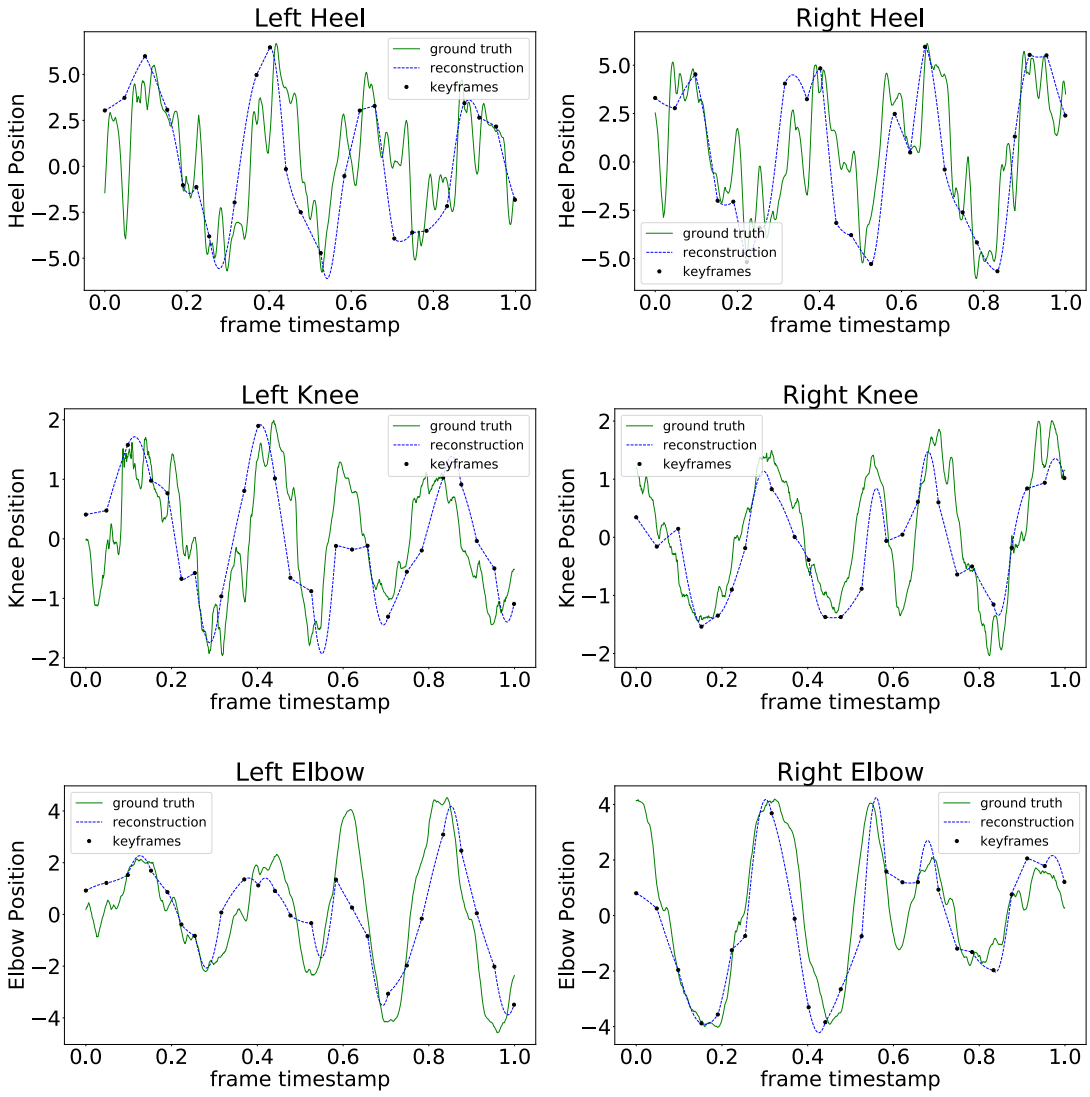


Figure A.4: MotionSynth 120 Hz reconstructed heel, knee, and elbow positions for a 12 second motion at two kf/s with our VQ-based key-framing transformer.

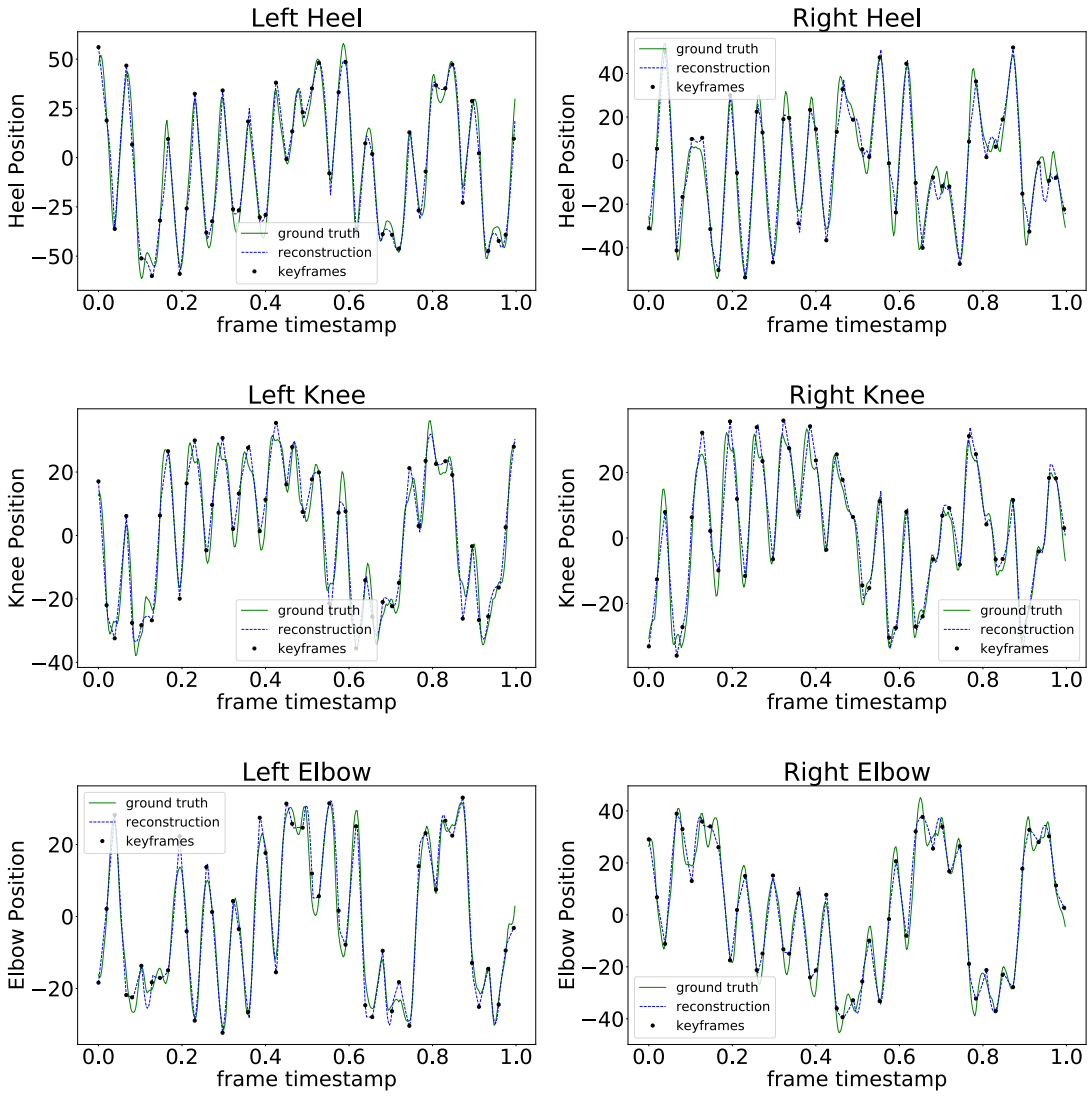


Figure A.5: LAFAN1 test-set reconstructed heel, knee, and elbow positions for a twelve-second motion at four kf/s with our continuous autoencoding key-framing transformer.

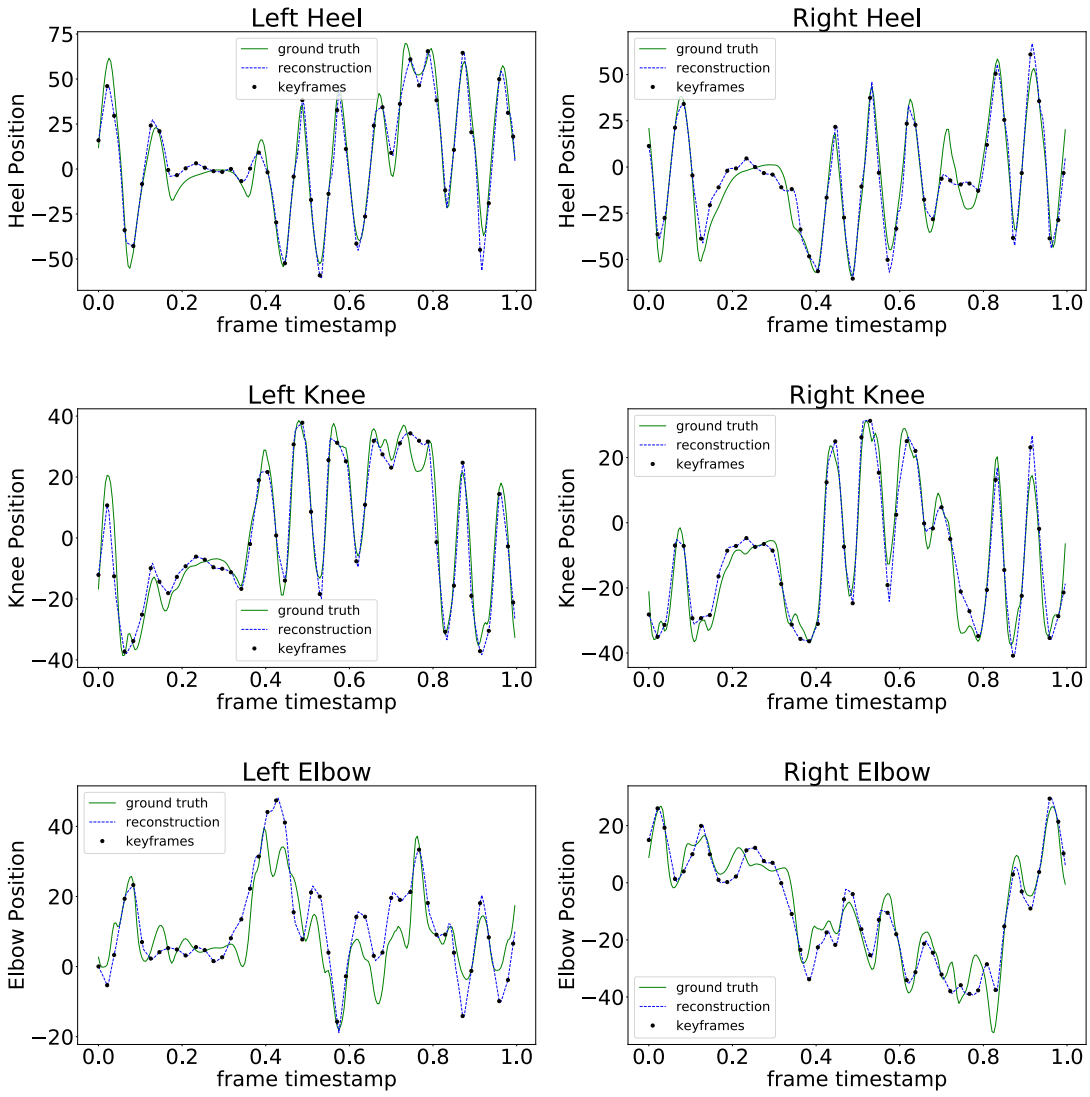


Figure A.6: LAFAN1 test-set reconstructed heel, knee, and elbow positions for an eight second motion at six kf/s with our VQ-based key-framing transformer.

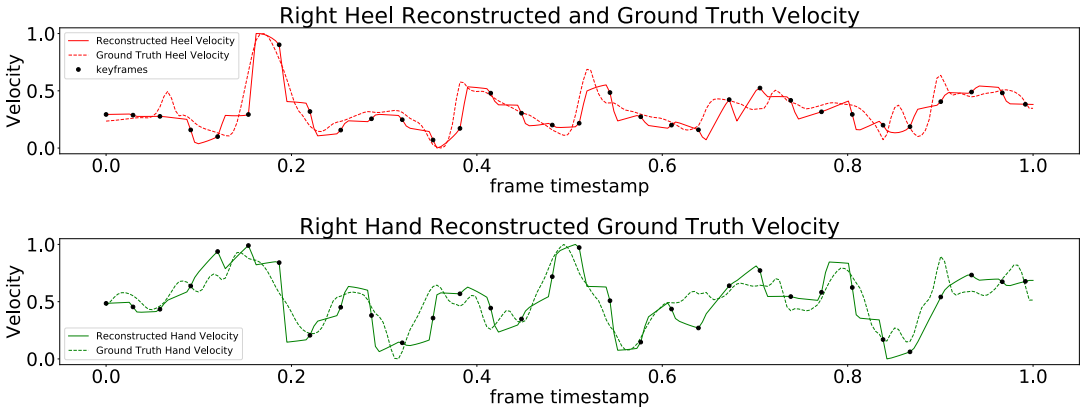


Figure A.7: Reconstructed velocities of the heel (top, red) and hand (bottom, green) joints with associated key-frame placements (black) along with the ground truth velocities for both joints. Velocities are reconstructed with our continuous key-framing approach trained to reconstruct an eight-second motion at four kf/s.

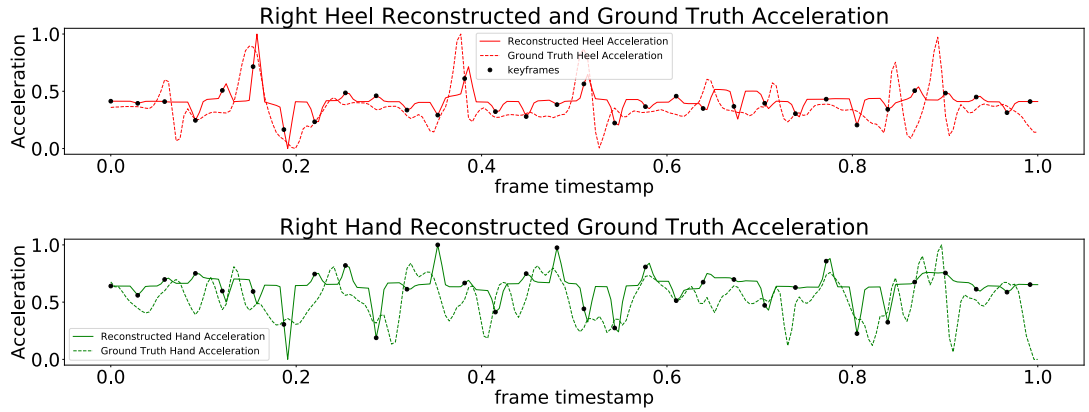


Figure A.8: Reconstructed accelerations of the heel (top, red) and hand (bottom, green) joints with associated key-frame placements (black) along with the ground truth acceleration for both joints. Accelerations are reconstructed with our continuous key-framing approach trained to reconstruct an eight-second motion at four kf/s.

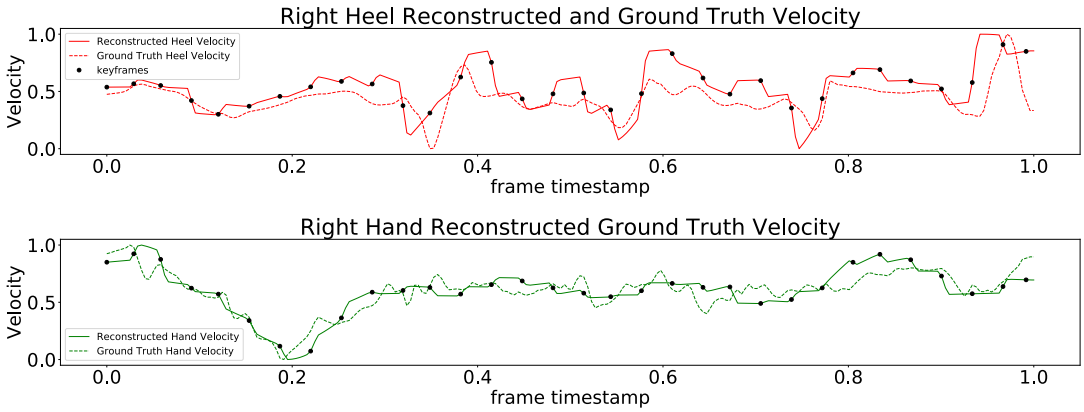


Figure A.9: Reconstructed velocities of the heel (top, red) and hand (bottom, green) joints with associated key-frame placements (black) along with the ground truth velocities for both joints. Velocities are reconstructed with our continuous key-framing approach trained to reconstruct an eight-second motion at four kf/s.

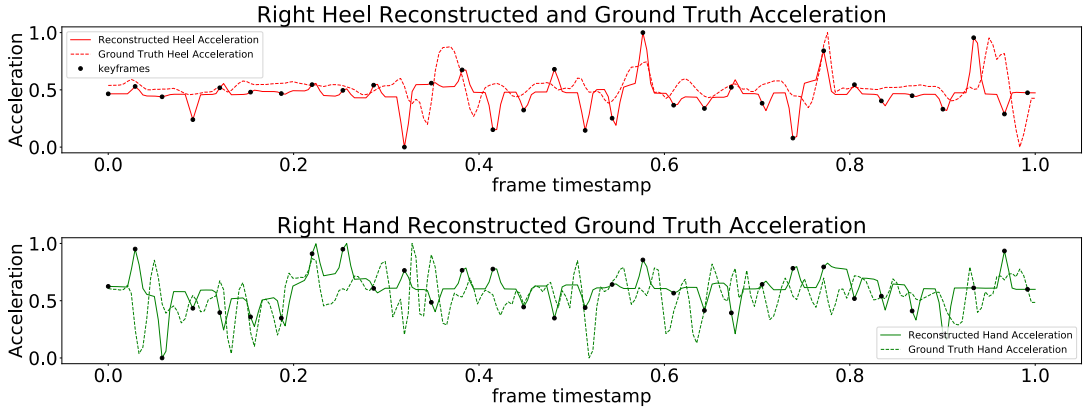


Figure A.10: Reconstructed accelerations of the heel (top, red) and hand (bottom, green) joints with associated key-frame placements (black) along with the ground truth acceleration for both joints. Accelerations are reconstructed with our continuous key-framing approach trained to reconstruct an eight-second motion at four kf/s.

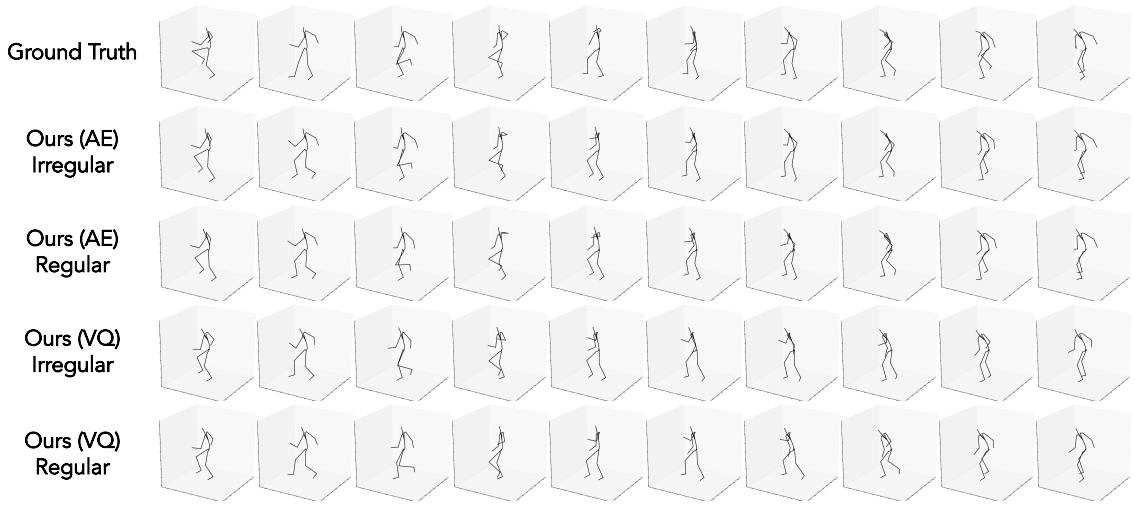


Figure A.11: A test-set motion sample from the LAFAN1 dataset. We compare our continuous and VQ-based key-framing approach trained to reconstruct a 12-second motion with four key-frames per second. For both our continuous and VQ-based models, we present the reconstructed motion with learned irregular spacing as well as regular spacing.

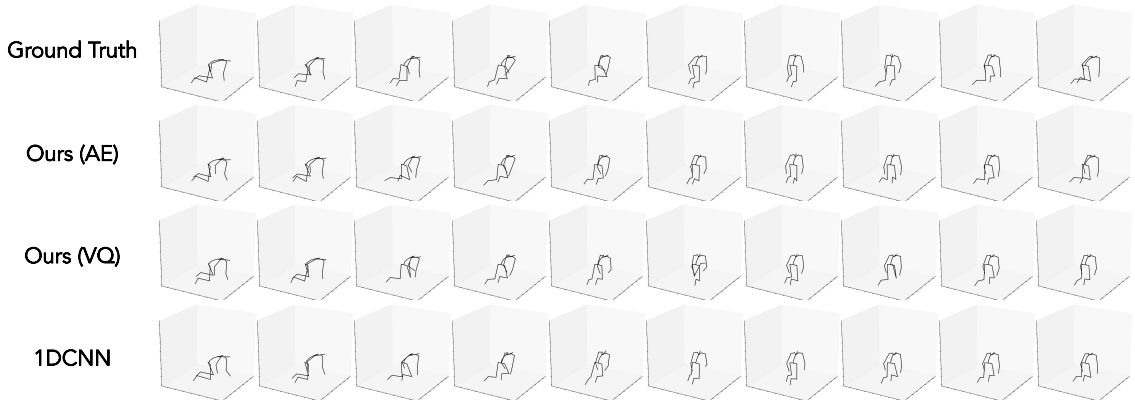


Figure A.12: A 12-second test-set motion sample from the LAFAN1 dataset reconstructed at six kf/s. We compare each of our model classes with learned irregular spacing, including our continuous attention-based key-framing, VQ attention-based key-framing, and 1DCNN baseline.