The Australian National University
2600 ACT | Canberra | Australia

Australian National University

School of Computing

College of Engineering, Computing and Cybernetics (CECC)

Seamless Human Motion Synthesis with Interactive Control Inputs

— Honours project (S1/S2 2024)

A thesis submitted for the degree
Bachelor of Advanced Computing

By:
Qianxuan Lin

Supervisors:
Prof. Hongdong Li

October 2024

Declaration:

I declare that this work:

- upholds the principles of academic integrity, as defined in the University Academic Misconduct Rules;
- is original, except where collaboration (for example group work) has been authorised in writing by the course convener in the class summary and/or Wattle site;
- is produced for the purposes of this assessment task and has not been submitted for assessment in any other context, except where authorised in writing by the course convener;
- gives appropriate acknowledgement of the ideas, scholarship and intellectual property of others insofar as these have been used;
- in no part involves copying, cheating, collusion, fabrication, plagiarism or recycling.

October, Qianxuan Lin

Acknowledgements

I would like to express my deepest gratitude to Professor Hongdong Li for his extraordinary supervision throughout this year. First, I want to thank him for accepting me as an honours student, a key milestone in my academic journey.

Despite his busy academic schedule, Professor Li has always been available whenever I needed guidance. I greatly appreciate the fact that I could find him in his office almost every time I sought help—something I know is not common with all supervisors. Each time I found him, I never took it for granted. His supportive approach, without being pushy, has been truly vital for me.

This year has passed by all too quickly, and I feel truly fortunate to have had him as my supervisor. This year has been a highlighted, happy year in my life. Lastly, thank myself, without whom this work would not be done.

Abstract

Realistic and responsive human motion synthesis is essential for interactive applications like gaming, virtual reality, and animation. Traditional methods relying on text inputs or predefined action categories are limited in flexibility and adaptability to dynamic scenarios. This thesis introduces a framework employing a transformer-based autoencoder that directly maps intuitive and interactive control inputs—such as joystick movements or touch gestures—to natural human motions without the need for predefined action labels or textual descriptions. Utilizing the AMASS dataset and the SMPL model for consistent and anatomically accurate motion representation, the proposed model leverages self-attention mechanisms to capture complex temporal and spatial dependencies in motion sequences. This results in smoother and more natural synthesized motions compared to conventional methods. The framework demonstrates enhanced naturalness and responsiveness in motion synthesis, offering a scalable and adaptable solution for interactive applications in gaming, virtual reality, and simulation.

Table of Contents

Chapter 1

Introduction

1.1 Motivation

Human motion synthesis is a critical field within computer vision and graphics, dedicated to generating realistic human movements in digital environments. It has wide-ranging applications in industries such as gaming, virtual reality, film, and simulation, where creating natural and believable human motion is essential for user engagement and immersion. Traditional approaches to motion synthesis often rely heavily on text inputs or predefined action categories, which can be limiting. These methods struggle to adapt to dynamic and interactive scenarios, making it challenging to achieve realistic motion, particularly when flexibility and responsiveness are required.

The significance of advancing human motion synthesis lies in its potential to overcome these limitations and create more adaptable, scalable solutions. The recent integration of deep learning, especially transformer-based models, has introduced new possibilities for addressing these challenges. By learning complex temporal and spatial patterns in motion data, transformer models enable the generation of more flexible and responsive human movements. This advancement allows for smoother transitions, real-time adaptability, and a broader range of motion synthesis, making it particularly relevant for applications requiring interactive or personalized experiences. The research presented in this thesis is motivated by the need to enhance the naturalness, responsiveness, and scalability of human motion synthesis, pushing the boundaries of what is achievable with current methodologies.

1.2 Problem Definition

Despite significant advancements, current human motion synthesis methods face several challenges that limit their effectiveness and applicability. One major issue is the reliance

on limited input modalities; many existing models depend on text descriptions or predefined action categories, which may not capture the nuances of user intent or allow for real-time interaction. This constraint hinders the ability to generate personalized or context-specific motions promptly. Additionally, there is a lack of responsiveness in generating smooth and natural movements in response to intuitive controls like joysticks or touch inputs. The complexity of human motion dynamics makes it difficult for models to produce seamless transitions and realistic movements based on such inputs. Moreover, adaptability remains a concern, as existing models often struggle to generalize to unseen actions or adapt to varying user inputs without extensive retraining. This limitation reduces the practicality of deploying these models in dynamic environments where user interactions are unpredictable.

This thesis addresses these challenges by proposing a transformer-based autoencoder framework that maps intuitive control inputs to realistic human motions. By leveraging the strengths of transformers in capturing temporal dependencies and modeling sequential data, the proposed method aims to enhance both the naturalness and responsiveness of synthesized motions.

1.3 Objectives

The primary objectives of this research are to develop a comprehensive framework that overcomes the limitations of current human motion synthesis methods. Specifically, the research aims to:

1. **Develop a Transformer-Based Autoencoder**: Design and implement an architecture capable of learning a motion manifold from unlabelled motion data, effectively capturing the underlying structure of natural human movements.
2. **Map Control Inputs to Motion**: Create a system that translates intuitive control inputs, such as joystick or mobile touch gestures, into smooth and responsive human movements, facilitating real-time interaction.
3. **Enhance Naturalness and Responsiveness**: Leverage attention mechanisms inherent in transformer models to capture temporal sequences and joint relationships in motion data, resulting in more natural and fluid motion synthesis.
4. **Evaluate Model Performance**: Propose evaluation metrics and conduct ablation studies to assess the effectiveness of the proposed method, providing insights into its strengths and areas for improvement.

1.4 Contributions

This thesis makes several contributions to the field of human motion synthesis:

Firstly, it introduces a novel transformer-based framework that effectively learns from and reconstructs human motion data. By utilizing an autoencoder architecture with

transformer components, the model captures complex temporal and spatial dependencies in motion sequences.

Secondly, it develops a control-to-motion mapping mechanism that translates intuitive user inputs into realistic human motions. This mechanism bridges the gap between user intent and motion generation, enabling more interactive and responsive applications.

Thirdly, the research demonstrates improved motion synthesis, showcasing enhanced naturalness and responsiveness in generated motions compared to existing methods. The use of attention mechanisms allows the model to produce more fluid and realistic movements.

Lastly, the thesis outlines an extensive evaluation plan, proposing strategies and metrics for future implementation and testing. This plan provides a foundation for assessing the model's performance and guiding further development.

1.5 Thesis Structure

This thesis contains five chapters. This chapter provides an overview of the importance of human motion synthesis, detailing its applications and the challenges posed by current approaches. Moreover, we discuss the motivations and objectives of our proposed transformer-based autoencoder framework for generating realistic human motion from interactive control inputs. Following this chapter, the remaining four chapters are summarised as follows:

Chapter 2 discusses the background and related work. This chapter provides an overview of the fundamental concepts and models, focusing on machine learning foundations and human motion data representation. It also covers the evolution of human motion synthesis, including problem settings, early approaches, and conditional motion synthesis methods. Methodologies employed in human motion synthesis are outlined, with a review of model architectures, kinematic constraints, and evaluation strategies.

Chapter 3 details the methodology developed in this research. It describes the problem definition, introduces the learning motion manifold via autoencoders, and explains the mapping from control inputs to seamless human motions. The chapter provides implementation details and a discussion of network and training hyperparameters.

Chapter 4 presents the evaluation of the proposed model. It outlines the experimental setup, benchmark datasets, baseline models, and evaluation metrics. The chapter also provides both quantitative and qualitative analysis of the results, including ablation studies.

Chapter 5 concludes the thesis by summarizing the contributions made and outlining potential areas for future work. Topics such as ...

Chapter 2

Background and Related Work

In this chapter, we present the foundational concepts, models, and methodologies that underpin the research conducted in this thesis. Section 2.1 begins by introducing essential machine learning models, such as Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), Autoencoders, and Transformers. This section lays the groundwork for understanding their applications in human motion synthesis, highlighting the strengths and limitations of each model, as well as key advancements like Vision Transformers, Masked Autoencoders, and Generative Adversarial Networks (GANs).

Next, in Section **??**, we explore the nature of human motion data and its representation. This includes a discussion of motion capture technologies, the SMPL model for 3D human body representation, and prominent datasets used in human motion synthesis research. These datasets and models form the foundation of the data-driven approaches employed in the field.

Section 2.2 focuses on problem settings within human motion synthesis, reviewing early approaches and progressively covering tasks such as motion prediction, action-based synthesis, text-conditioned motion generation, and style and scene-aware motion synthesis. This section demonstrates how the field has evolved from basic probabilistic models to more complex and context-aware methodologies.

In Section 2.3, we delve into the key methodologies utilized in human motion synthesis, discussing models such as RNN-based approaches, Variational Autoencoders (VAEs), GANs, and the growing influence of Transformer-based models and diffusion models. We also cover multimodal learning and cross-modal representations, emphasizing their role in integrating different input modalities like text and audio for motion generation.

Finally, Sections 2.4 and 2.5 address the challenges and future directions in human motion synthesis, as well as the evaluation metrics used in this field. By addressing issues like data scarcity, physical plausibility, and computational efficiency, we identify key

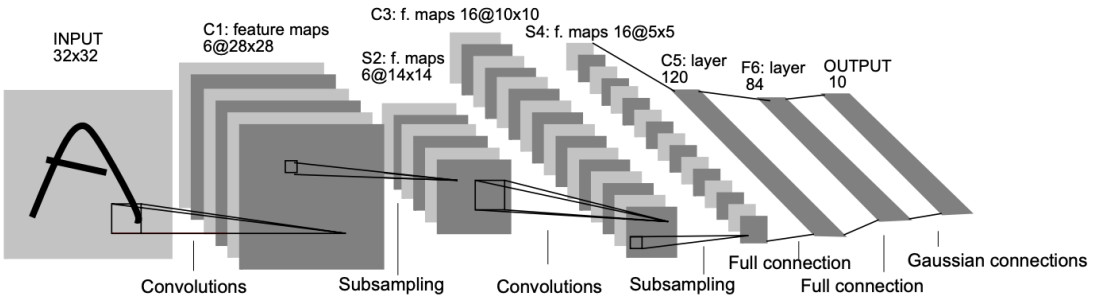


Figure 2.1: Architecture of LeNet-5, a Convolutional Neural Network, here for digits recognition. Figure from (LeCun et al., 1998)

areas for future research. These sections provide the necessary foundation for understanding the novel contributions presented in subsequent chapters.

2.1 Fundamental Concepts and Models

2.1.1 Machine Learning Foundations

Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNNs) are a class of deep learning models that have revolutionized the field of computer vision due to their exceptional ability to automatically and adaptively learn spatial hierarchies of features from input data. Introduced by LeCun et al. (1998), CNNs were initially applied to document recognition tasks, where they significantly outperformed traditional machine learning algorithms by effectively capturing local patterns and spatial dependencies in images. A visualization of the CNN architecture is provided in Figure 2.1.

The foundational architecture of CNNs consists of several key components:

- **Convolutional Layers**: These layers apply learnable filters (kernels) that convolve across the input's spatial dimensions, enabling the network to detect various features such as edges, textures, and shapes at different levels of abstraction.
- **Pooling Layers**: By performing operations like max pooling or average pooling, these layers reduce the spatial dimensions of the data, which helps in decreasing computational complexity and provides a form of spatial invariance.
- **Fully Connected Layers**: Positioned towards the end of the network, these layers interpret the features extracted by previous layers to perform tasks such as classification or regression.

The innovation of CNNs lies in their ability to automatically learn feature representations directly from raw data, eliminating the need for manual feature engineering. This

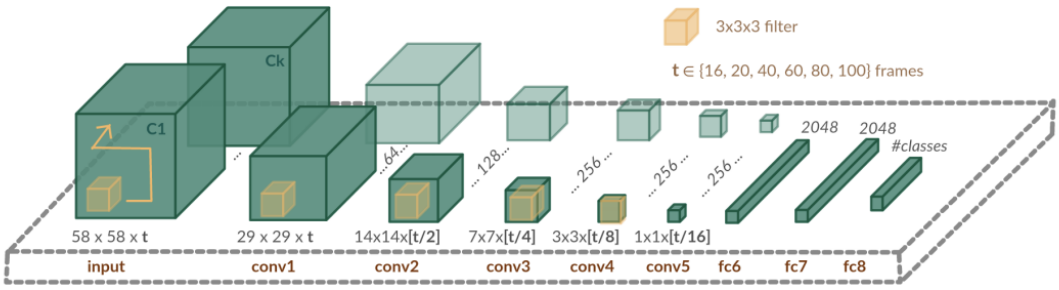


Figure 2.2: Network architecture of Spatio-temporal convolutions. Figure from (Varol et al., 2017a)

capability has made CNNs the de facto standard for a wide range of computer vision applications.

Building upon the success of CNNs in image recognition, Varol et al. (2017a) extended the concept to action recognition in videos by introducing Long-Term Temporal Convolutions (LTCs). Their work addressed the challenge of modeling temporal dynamics over extended periods, which is essential for understanding complex human actions in video sequences. A visualization of the LTCs is provided in Figure 2.2.

The key contributions of Varol et al. (2017a) include:

- **Temporal Convolutions**: By employing 3D convolutional kernels, the network captures spatiotemporal features across both spatial and temporal dimensions, effectively modeling motion and temporal dependencies.

- **Long-Term Modeling**: The network architecture is designed to process long sequences of frames, allowing it to learn from extended temporal contexts, which is critical for recognizing actions that unfold over time.

The integration of temporal convolutions into CNNs enables the extraction of motion cues and temporal patterns that are not accessible through standard spatial convolutions alone. This advancement has significant implications for tasks such as human motion synthesis, where understanding the temporal evolution of motion is crucial.

In summary, CNNs and their temporal extensions have profoundly impacted the field of computer vision by providing powerful tools for feature extraction and pattern recognition in both spatial and temporal domains. Their ability to learn hierarchical representations makes them particularly well-suited for the complex task of human motion analysis and synthesis, which is a central focus of this thesis.

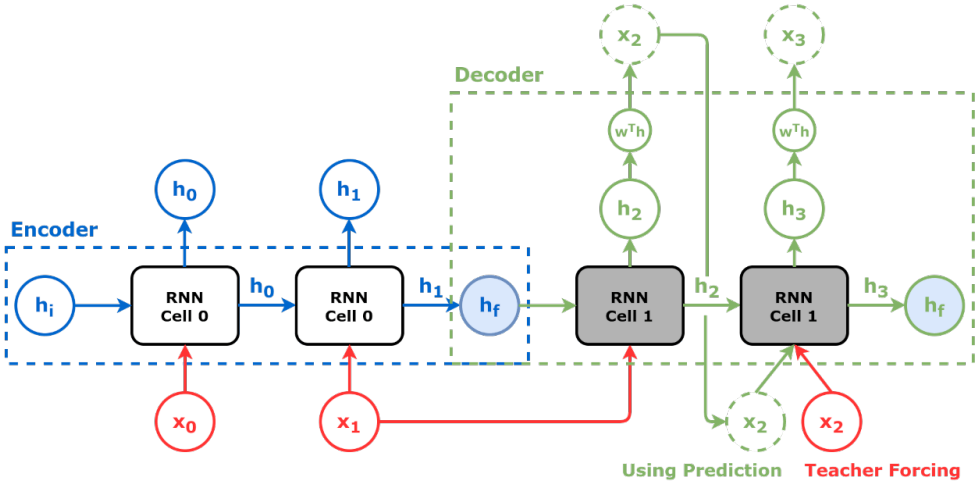


Figure 2.3: Seq2seq RNN encoder-decoder with attention mechanism. Image by Daniel Voigt Godoy, licensed under CC BY 4.0 (Godoy, 2021).

Recurrent Neural Networks (RNNs)

Recurrent Neural Networks (RNNs) are a class of neural networks designed to handle sequential data by maintaining a hidden state that captures information from previous time steps. This architecture allows RNNs to exhibit temporal dynamic behavior, making them suitable for tasks involving sequences, such as language modeling, speech recognition, and time-series prediction.

The foundational work on RNNs was presented by Elman (1990) in his seminal paper *Finding Structure in Time* Elman (1990). Elman introduced a simple recurrent network where the hidden state from the previous time step is fed back into the network along with the current input. This feedback loop enables the network to maintain a form of memory, capturing temporal dependencies in sequential data.

Despite their potential, traditional RNNs faced significant challenges in learning long-term dependencies due to issues like vanishing and exploding gradients. Bengio et al. (1994) analyzed these difficulties in their work *Learning Long-Term Dependencies with Gradient Descent is Difficult* Bengio et al. (1994). They showed that as sequences become longer, the gradients propagated back through time either vanish or explode, making training unstable and inefficient.

To overcome these limitations, Hochreiter and Schmidhuber (1997) introduced the Long Short-Term Memory (LSTM) network Hochreiter and Schmidhuber (1997). LSTMs address the vanishing gradient problem by incorporating memory cells and gating mechanisms—input, output, and forget gates—that regulate the flow of information. This architecture allows LSTMs to capture both short-term and long-term dependencies effectively.

Further advancements were made by Sutskever et al. (2014) with the development

of Sequence-to-Sequence (Seq2Seq) models Sutskever et al. (2014). Seq2Seq models utilize RNNs to map input sequences to output sequences of varying lengths. This framework has been particularly influential in machine translation and conversational modeling, where the length of the input and output sequences may differ significantly. The two Recurrent Neural Networks (RNNs) can be employed in an encoder-decoder architecture, where the encoder processes an input sequence into a series of hidden vectors, and the decoder generates an output sequence from these hidden vectors, with an optional attention mechanism. This approach was pivotal in building state-of-the-art neural machine translation models between 2014 and 2017, and it laid the groundwork for the development of Transformer models. A visualization of Seq2Seq is provided in Figure 2.3.

In summary, RNNs and their extensions like LSTMs have been pivotal in advancing the modeling of sequential data. They enable networks to maintain temporal context, capture long-term dependencies, and have significantly impacted various domains in machine learning. However, challenges remain in terms of computational efficiency and parallelization, which have led to the exploration of alternative architectures such as Transformers.

Autoencoders

Autoencoders are a type of neural network designed for unsupervised learning of efficient data encodings. They aim to learn a compressed representation (encoding) of input data by training the network to reconstruct the input from the encoding. This process forces the network to capture the most salient features of the data, effectively performing dimensionality reduction.

The concept was significantly advanced by Hinton and Salakhutdinov (2006) in their paper *Reducing the Dimensionality of Data with Neural Networks*. They demonstrated that deep autoencoders could learn low-dimensional representations of high-dimensional data, outperforming traditional methods like Principal Component Analysis (PCA). The architecture comprises an encoder that maps the input data to a latent space and a decoder that reconstructs the data from this latent representation.

Building upon this foundation, Kingma and Welling (2014) introduced the Variational Autoencoder (VAE), which incorporates probabilistic elements into the autoencoder framework. VAEs learn a distribution over the latent space, typically assuming a Gaussian prior, enabling them to generate new data samples by sampling from the latent space. This probabilistic approach makes VAEs powerful generative models.

In the realm of human motion synthesis, Holden et al. (2015) applied autoencoders to learn motion manifolds. By training convolutional autoencoders on motion capture data, they were able to capture the complex, nonlinear structures inherent in human motion. The learned manifolds facilitate tasks such as motion interpolation, denoising, and synthesis, providing a compact and expressive representation of motion data.

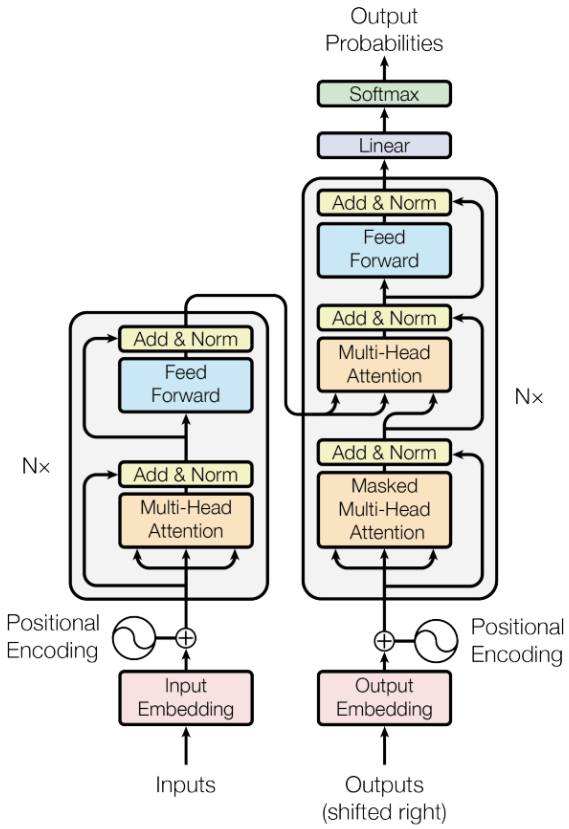


Figure 2.4: The Transformer model architecture. Figure from (Vaswani et al., 2017a)

Autoencoders, particularly VAEs, have become fundamental tools in representation learning and generative modeling. They enable the extraction of meaningful features from high-dimensional data and serve as building blocks for more complex architectures in various machine learning applications.

Transformers

Transformers have fundamentally reshaped the landscape of sequence modeling and transduction tasks across various domains. Introduced by Vaswani et al. (2017a) in their groundbreaking paper *Attention Is All You Need*, the Transformer architecture departed from traditional sequential processing models like Recurrent Neural Networks (RNNs) and Convolutional Neural Networks (CNNs). Instead, it leveraged a novel mechanism known as self-attention to capture global dependencies within input sequences efficiently. The Transformer model architecture is visualized in Figure 2.4.

The key innovation of the Transformer lies in its ability to process all elements of a sequence in parallel, thanks to the self-attention mechanism. This mechanism computes

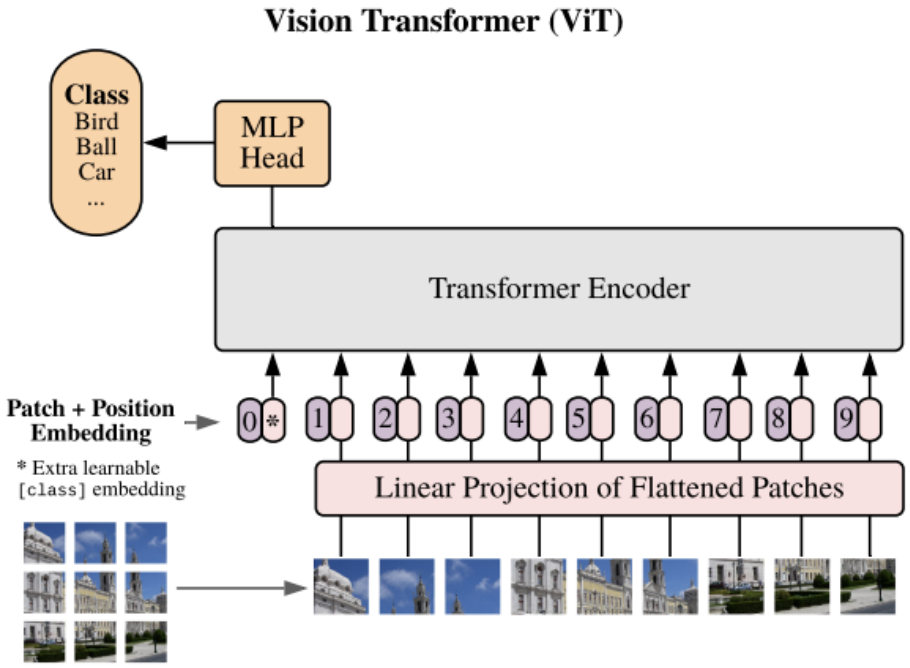


Figure 2.5: The ViT architecture. Figure from (Dosovitskiy et al., 2021a)

attention scores between all pairs of positions in the sequence, allowing the model to weigh the relevance of each element relative to others. As a result, Transformers can model long-range dependencies without the limitations imposed by sequential computations in RNNs or the locality constraints in CNNs.

Building upon the success of Transformers in natural language processing, particularly in machine translation, Dosovitskiy et al. (2021a) extended the Transformer architecture to the field of computer vision with the introduction of the Vision Transformer (ViT). In their work *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*, they demonstrated that Transformers could achieve state-of-the-art performance on image classification tasks. This model architecture is visualized in Figure 2.5.

The ViT adapts the Transformer to handle image data by dividing images into a sequence of fixed-size, non-overlapping patches (e.g., 16×16 pixels). Each patch is flattened and linearly projected into an embedding space, analogous to word embeddings in NLP. Positional embeddings are added to retain spatial information, and the resulting sequence of patch embeddings is fed into a standard Transformer encoder.

One of the significant findings of the ViT is that Transformers can learn image representations without relying on convolutional inductive biases, provided they are trained on sufficiently large datasets. However, this requirement for large-scale data is a lim-

itation, as training on smaller datasets often results in inferior performance compared to CNNs. The ViT's success underscores the potential of self-attention mechanisms to model spatial relationships in images, challenging the long-held dominance of convolutional architectures in computer vision.

To address the data efficiency limitations and further enhance the capabilities of vision Transformers, He et al. (2021) proposed the Masked Autoencoder (MAE) in their paper *Masked Autoencoders Are Scalable Vision Learners*. The MAE introduces a self-supervised pre-training strategy inspired by the masked language modeling objectives used in NLP models like BERT.

In the MAE framework, a high proportion of the input image patches (e.g., 75%) are randomly masked out. The encoder processes only the visible patches, significantly reducing computational cost. A lightweight decoder is then tasked with reconstructing the original image from the latent representations of the unmasked patches and mask tokens. This reconstruction objective forces the encoder to learn rich, context-aware representations that capture both local and global structures in the data.

The MAE's masked reconstruction approach enhances data efficiency and enables effective learning from smaller datasets. By pre-training the model in a self-supervised manner, the MAE can leverage vast amounts of unlabeled data, making it highly scalable. This methodology is particularly relevant to our work, where we employ Transformer-based autoencoding architectures for human motion synthesis.

The integration of Transformers with autoencoding techniques offers several advantages:

- **Efficient Representation Learning**: By focusing on reconstructing missing information, the model learns compact and meaningful representations of complex data.
- **Scalability**: The reduced computational burden during encoding allows for the training of larger models or processing of longer sequences.
- **Applicability to Multimodal Data**: The flexibility of Transformers makes them suitable for handling various data modalities, including sequential motion data.

In summary, the evolution from the original Transformer to ViT and MAE demonstrates the versatility of self-attention mechanisms beyond NLP. The MAE, in particular, addresses critical challenges in vision Transformers, paving the way for more efficient and effective models. Our work builds upon these advancements to develop novel approaches for human motion synthesis, leveraging the strengths of Transformers and masked autoencoding to model the intricate dynamics of human movement.

Generative Adversarial Networks (GANs)

Generative Adversarial Networks (GANs), introduced by Goodfellow et al. (2014) in their seminal paper *Generative Adversarial Nets*, have established themselves as a cornerstone of generative modeling in deep learning. GANs consist of two competing neural

Algorithm 1 Minibatch stochastic gradient descent training of generative adversarial nets. The number of steps to apply to the discriminator, k, is a hyperparameter. We used $k = 1$, the least expensive option, in our experiments.

for number of training iterations **do**
  for k steps **do**
    • Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
    • Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.
    • Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D\left(\boldsymbol{x}^{(i)}\right) + \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right) \right].$$

  end for
  • Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
  • Update the generator by descending its stochastic gradient:

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log\left(1 - D\left(G\left(\boldsymbol{z}^{(i)}\right)\right)\right).$$

end for
The gradient-based updates can use any standard gradient-based learning rule. We used momentum in our experiments.

Figure 2.6: Minibatch stochastic gradient descent training algorithm for Generative Adversarial Networks (GANs), adapted from (Goodfellow et al., 2014). The discriminator is updated by maximizing its ability to distinguish real from generated data, while the generator is updated to minimize the discriminator's accuracy on generated data.

networks—the generator and the discriminator—that engage in a minimax game. The generator aims to produce data indistinguishable from real data, while the discriminator endeavors to differentiate between real and generated (fake) data.

The adversarial training process can be formally described as:

$$\min_G \max_D V(D, G) = \mathbb{E}_{\mathbf{x} \sim p_{\text{data}}(\mathbf{x})}[\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z} \sim p_{\mathbf{z}}(\mathbf{z})}[\log(1 - D(G(\mathbf{z})))]$$

where G is the generator, D is the discriminator, p_{data} is the data distribution, and $p_{\mathbf{z}}$ is the prior over the latent space. The training algorithm for GANs is shown in Figure 2.6.

This framework enables GANs to model complex, high-dimensional data distributions without explicitly defining a likelihood function. The generator learns to map samples from a simple prior distribution (e.g., Gaussian noise) to the data space, effectively capturing the underlying data distribution.

GANs have achieved remarkable success in various domains, including image synthesis, style transfer, and data augmentation. Their ability to generate high-fidelity samples has made them invaluable for tasks requiring realistic data generation. In the context of human motion synthesis, GANs can be employed to produce plausible motion sequences, aiding in animation, virtual reality, and simulation applications.

Despite their success, GANs present several challenges:

- **Training Instability**: The adversarial nature of GANs can lead to unstable training dynamics, requiring careful tuning of hyperparameters and network architectures.
- **Mode Collapse**: The generator may converge to producing limited varieties of outputs, failing to capture the full diversity of the data distribution.
- **Sensitivity to Hyperparameters**: GAN performance is often sensitive to the choice of learning rates, batch sizes, and network designs.

Various extensions and techniques have been proposed to address these issues, such as Wasserstein GANs (Arjovsky et al., 2017), feature matching (Salimans et al., 2016), and spectral normalization (Miyato et al., 2018). These advancements have improved the stability and robustness of GAN training, expanding their applicability across different domains.

Regularization and Optimization Techniques

Effective training of deep neural networks necessitates strategies to prevent overfitting and ensure convergence. Regularization and optimization techniques play a crucial role in enhancing model generalization and training efficiency.

Dropout

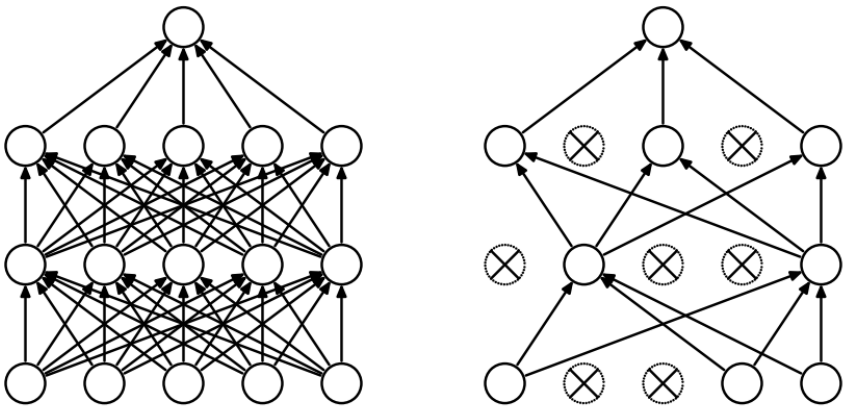
Srivastava et al. (2014) introduced *Dropout*, a simple yet powerful regularization technique that mitigates overfitting in neural networks. During training, Dropout randomly sets a fraction of the input units to zero at each update, which prevents units from co-adapting excessively. Mathematically, for each neuron, the output becomes:

$$\mathbf{y} = \mathbf{W}(\mathbf{x} \odot \mathbf{r}) + \mathbf{b}$$

where $\odot$ denotes element-wise multiplication, $\mathbf{r}$ is a binary mask vector with entries drawn from a Bernoulli distribution, and $\mathbf{W}, \mathbf{b}$ are the weights and biases. Dropout encourages the network to learn redundant representations, improving robustness and generalization on unseen data. A visualization of dropout is in Figure 2.7

Batch Normalization

Training deep networks can be hindered by internal covariate shift—the change in the distribution of network activations due to updates in preceding layers. Ioffe and Szegedy



(a) Standard Neural Net (b) After applying dropout.

Figure 2.7: Dropout Neural Net Model. Left: A standard neural net with 2 hidden layers. Right: An example of a thinned net produced by applying dropout to the network on the left. Crossed units have been dropped. Figure from (Srivastava et al., 2014)

(2015) addressed this issue with *Batch Normalization*, which normalizes the inputs of each layer to have zero mean and unit variance within a mini-batch:

$$\hat{\mathbf{x}}^{(k)} = \frac{\mathbf{x}^{(k)} - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}}$$

where $\mu_{\mathcal{B}}$ and $\sigma_{\mathcal{B}}^2$ are the mean and variance of the mini-batch $\mathcal{B}$, and ϵ is a small constant for numerical stability.

Batch Normalization accelerates training, allows for higher learning rates, and acts as a form of regularization, reducing the need for Dropout in some cases.

Layer Normalization

While Batch Normalization is effective, it depends on batch statistics, which can be problematic for recurrent networks or small batch sizes. Ba et al. (2016) proposed *Layer Normalization*, which normalizes across the features of each data sample rather than across the batch:

$$\hat{\mathbf{x}} = \frac{\mathbf{x} - \mu}{\sqrt{\sigma^2 + \epsilon}}$$

where μ and σ^2 are the mean and variance computed over the features of a single sample.

Layer Normalization is particularly beneficial for sequence modeling tasks, as it ensures consistent normalization regardless of batch size and sequence length.

Other Models

While deep learning architectures dominate current research, classical machine learning models remain relevant for specific applications. Pons-Moll et al. (2015) explored the use of *Metric Regression Forests* for human pose estimation. Regression forests are ensemble models consisting of multiple decision trees that perform regression by averaging the outputs of individual trees.

In their work, they leveraged regression forests to map input features derived from depth images to 3D human poses. The advantages of regression forests include:

- **Computational Efficiency**: Trees can be trained and evaluated quickly, enabling real-time applications.
- **Interpretability**: Decision paths within trees can provide insights into the feature importance and decision-making process.
- **Robustness**: Ensemble methods reduce variance and improve generalization compared to single-tree models.

Although surpassed by deep learning methods in many benchmarks, regression forests offer a viable alternative in scenarios where data is limited or interpretability is crucial.

2.1.2 Human Motion Data and Representation

Understanding and modeling human motion requires comprehensive datasets and robust representation models. This section discusses the fundamental components of human motion data acquisition and representation, focusing on motion capture technologies, the SMPL model for body representation, and prominent datasets utilized in human motion synthesis research.

Motion Capture Data

Motion capture (mocap) is a technology used to record the movements of human subjects, providing precise and high-fidelity data essential for analyzing and synthesizing human motion. Mocap systems often employ optical markers, inertial sensors, or depth cameras to capture the kinematics of the human body.

A pivotal contribution in this domain is the HumanEva dataset introduced by Sigal and Black (2010). The HumanEva dataset provides synchronized video and motion capture data, offering a standardized benchmark for evaluating articulated human motion algorithms. It includes multiple subjects performing a variety of actions, captured using a multi-camera system and a marker-based mocap setup. The dataset's synchroniza-

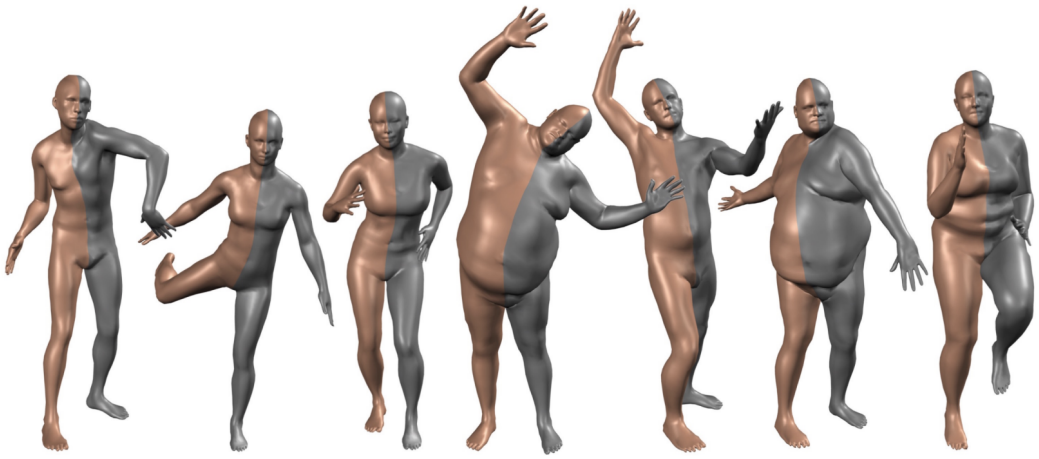


Figure 2.8: SMPL is a realistic learned model of human body shape and pose that is compatible with existing rendering engines, allows animator control, and is available for research purposes. Figure from ()

tion between 2D video and 3D motion data enables researchers to develop and validate methods for pose estimation, action recognition, and motion synthesis.

SMPL Model

Accurate and expressive representation of the human body is crucial for realistic motion synthesis. The Skinned Multi-Person Linear (SMPL) model, proposed by Loper et al. (2015a), is a widely adopted parametric model that represents the human body shape and pose in a unified framework. SMPL models the human body as a mesh with a fixed topology, parameterized by shape and pose parameters, as shown in Figure 2.8.

Key features of the SMPL model include:

- **Shape Parameters**: A set of coefficients that capture individual body shape variations derived from a training corpus of 3D body scans.
- **Pose Parameters**: Joint rotations that define the body pose, articulated through a kinematic skeleton.
- **Linear Blend Skinning**: A deformation technique that combines the effects of skeletal motion and surface skinning to produce realistic deformations.

The SMPL model facilitates the synthesis of diverse human shapes and motions, enabling applications in computer graphics, virtual reality, and human-computer interaction. Its compatibility with mocap data and ease of integration into learning frameworks make it an essential tool in human motion research.

Datasets

High-quality datasets are indispensable for training and evaluating models in human motion synthesis. While numerous datasets exist, we focus on three foundational datasets closely related to this thesis: Human3.6M, CMU Motion Capture Database, and AMASS.

Human3.6M Human3.6M (Ionescu et al., 2014) is one of the largest and most comprehensive datasets for 3D human pose estimation and motion analysis. It contains approximately 3.6 million 3D human poses captured from seven professional actors performing 15 daily activities, such as walking, eating, and discussing.

Key characteristics:

- **Multimodal Data**: Includes synchronized video, depth maps, 3D joint positions, and time-of-flight data.
- **High Resolution**: Provides high-resolution imagery and accurate 3D joint annotations.
- **Standard Benchmark**: Widely used as a benchmark for evaluating pose estimation and motion synthesis algorithms.

Human3.6M's extensive coverage of actions and precise annotations make it a cornerstone dataset for developing data-driven models in human motion analysis.

CMU Motion Capture Database (CMU Mocap) The CMU Motion Capture Database (Carnegie Mellon University, 2003) is a seminal resource containing thousands of motion sequences recorded using a Vicon optical motion capture system. It encompasses a diverse range of human motions, including locomotion, sports, interactions, and complex activities.

Key characteristics:

- **Diversity**: Features motions from 144 subjects, covering a vast array of actions.
- **Accessibility**: Freely available to the research community, fostering widespread use.
- **Format**: Provides data in standard formats compatible with various animation and analysis tools.

The CMU Mocap dataset serves as a foundational dataset for animation, biomechanics, and machine learning research, enabling studies on motion synthesis, style transfer, and character animation.

AMASS (Archive of Motion Capture as Surface Shapes) AMASS (Mahmood et al., 2019) is a comprehensive dataset that aggregates 18 motion capture datasets into a unified format using the SMPL model. By parameterizing the mocap data with SMPL,

AMASS provides a consistent and high-quality representation of human motions across different datasets.

Key characteristics:

- **Large Scale**: Contains over 40 hours of motion data, amounting to millions of frames.
- **Unified Representation**: Uses SMPL parameters for consistent shape and pose representation.
- **Versatility**: Enables seamless integration and comparison of models trained on varied motion data.

AMASS facilitates large-scale learning of human motion models and supports research in motion synthesis, prediction, and understanding.

Other Notable Datasets In addition to the aforementioned datasets, several others contribute valuable resources to the field are shown in Table 2.1. These datasets enhance the diversity and richness of data available for human motion research, supporting advancements in various subfields such as pose estimation, action recognition, and multimodal learning.

Table 2.1: Other notable datasets contributing valuable resources to the field.

Dataset	Description
MPI-INF-3DHP (Mehta et al., 2017)	Provides 3D human pose data in both controlled and in-the-wild scenarios.
PoseTrack (Andriluka et al., 2018)	Focuses on multi-person pose estimation and tracking in videos.
KIT Motion-Language Dataset (Plappert et al., 2016)	Pairs motion capture data with natural language descriptions for action-conditioned motion synthesis.
SURREAL Dataset (Varol et al., 2017b)	Offers synthetic RGB videos and depth maps with 3D human motion data.
MoVi Dataset (Siddiqui et al., 2020)	Contains multimodal data, including motion capture, video, and audio.
3DPW (3D Pose in the Wild) (von Marcard et al., 2018)	Provides 3D pose annotations in natural outdoor environments.
BABEL (Punnakkal et al., 2021)	Enriches AMASS motions with action labels and temporal segmentation, bridging language and motion.

In conclusion, the combination of sophisticated motion capture data, expressive human body models like SMPL, and extensive datasets provides a robust foundation for modeling and synthesizing human motion. These resources are instrumental in driving forward

the research and applications discussed in this thesis.

2.2 Human Motion Synthesis: Problem Settings

Human motion synthesis is a pivotal area in computer graphics and animation, aiming to generate realistic and natural human movements through computational models. The complexity of human motion, characterized by high dimensionality, non-linear dynamics, and intricate temporal dependencies, presents significant challenges in modeling and synthesis. This section explores the evolution of motion synthesis methodologies, starting from early approaches to recent advancements in motion prediction, critically analyzing key contributions, comparing different methodologies, discussing limitations, and providing insights into future directions.

2.2.1 Early Approaches and Motion Prediction

Early Approaches to Motion Synthesis

The foundational work of Taylor et al. (2007) introduced a probabilistic model for human motion synthesis using binary latent variables. They proposed the use of Conditional Restricted Boltzmann Machines (CRBMs) to model temporal dependencies in motion capture data. By employing binary latent variables, the CRBM captured the stochastic nature of human motion, enabling the generation of plausible sequences by sampling from the learned distribution.

This approach was significant in demonstrating the potential of probabilistic models for motion synthesis. The CRBM effectively learned the dynamics of motion sequences without relying on explicit temporal modeling techniques like Hidden Markov Models. However, the reliance on binary latent variables introduced limitations in capturing the continuous and complex variations inherent in human motion. The model also faced challenges in scalability and computational efficiency, especially when dealing with high-dimensional motion data over long sequences.

Advancing beyond probabilistic models, Holden et al. (2016) pioneered the application of deep learning to character motion synthesis and editing. They developed a deep learning framework that utilized convolutional neural networks (CNNs) to learn motion representations directly from raw motion capture data. Their approach involved encoding motion sequences as two-dimensional images, where one dimension represented time, and the other represented joint rotations or positions. The CNNs then learned hierarchical features from these representations.

Holden et al. (2016)'s framework excelled in synthesizing high-quality, realistic motions and allowed for various motion editing tasks such as style transfer, interpolation, and blending. By leveraging the capabilities of deep learning, the model captured complex spatio-temporal patterns without the need for manual feature engineering. However, the approach had limitations in modeling long-range temporal dependencies due to the

inherent locality of CNNs. Additionally, training the model required large amounts of labeled motion capture data, which may not always be readily available.

Motion Prediction

Motion prediction focuses on forecasting future human movements based on observed motion sequences, which is crucial for applications in animation, robotics, and human-computer interaction. A significant challenge in motion prediction is capturing the temporal dynamics and variability of human motion while ensuring real-time performance.

Ren et al. (2020) addressed these challenges by introducing a self-supervised neural network for real-time motion prediction. Their model employed an encoder-decoder architecture, where the encoder extracted features from past motion data, and the decoder predicted future motions. By utilizing self-supervised learning, the model learned temporal dependencies without requiring labeled data, setting up a pretext task of predicting future frames from past observations.

This approach effectively reduced the reliance on large labeled datasets and enabled real-time inference, making it suitable for interactive applications. However, the model faced limitations in handling complex and diverse motion patterns, particularly those involving abrupt changes or non-repetitive movements. The self-supervised framework also depended heavily on the representativeness of the training data; insufficient variability could lead to poor generalization to unseen motions.

Robust motion in-betweening involves generating intermediate frames between key poses to produce smooth and natural motion sequences. This task is essential for animation and requires maintaining temporal coherence and physical plausibility. Techniques for motion in-betweening range from traditional interpolation methods to learning-based approaches that model the underlying motion dynamics.

Although specific references were not provided for robust motion in-betweening, the challenges in this area include ensuring consistency in movement dynamics, avoiding artifacts during interpolation, and capturing the nuances of human motion styles. Integrating physical constraints and biomechanical principles can enhance the realism of the synthesized motions.

Zhou et al. (2017) contributed to the field by proposing a weakly-supervised method for 3D human pose estimation in the wild. Their approach aimed to estimate 3D poses from 2D observations in unconstrained environments, leveraging weak supervision to overcome the scarcity of labeled 3D data. By combining 2D keypoint detections with geometric constraints and a reprojection loss, the model inferred 3D poses without explicit 3D annotations.

This method expanded the applicability of pose estimation models to real-world scenarios with diverse backgrounds and occlusions. However, the reliance on weak supervision introduced limitations in accuracy, particularly in complex scenes or with significant occlusions. The approach focused on static pose estimation rather than dynamic motion

prediction, indicating a need for further development to handle full motion synthesis tasks.

Comparative Analysis and Insights

The evolution from probabilistic models to deep learning frameworks in human motion synthesis reflects the ongoing pursuit of capturing the complexities of human motion accurately and efficiently. Taylor et al.'s work demonstrated the feasibility of modeling motion with binary latent variables, but it was constrained by the discrete nature of the representation and computational challenges.

Holden et al. (2016)'s deep learning framework leveraged the power of CNNs to learn from raw data, achieving high-quality motion synthesis and supporting motion editing tasks. However, the model's limitations in capturing long-range temporal dependencies and the requirement for extensive labeled data highlighted areas for improvement.

Ren et al. (2020)'s self-supervised approach addressed data scarcity by eliminating the need for labeled datasets and achieved real-time performance. Nonetheless, the effectiveness of the model depended on the diversity and representativeness of the training data, and it faced challenges with complex motion patterns.

Zhou et al. (2017)'s weakly-supervised method contributed to pose estimation in unconstrained environments, but the focus on static poses and the limitations in accuracy due to weak supervision indicated the need for integrating dynamic modeling for motion synthesis.

Overall, the advancements suggest that future research should focus on models that can capture long-range temporal dependencies, handle diverse and complex motion patterns, and reduce reliance on large labeled datasets. Incorporating attention mechanisms, recurrent architectures, or transformers may enhance the ability to model extended sequences. Additionally, integrating physical constraints and biomechanical knowledge can improve the realism and plausibility of synthesized motions.

2.2.2 Conditional Motion Synthesis

Action-Conditioned Motion Synthesis

Action-conditioned motion synthesis focuses on generating human motion sequences conditioned on specific action labels or descriptions. This problem setting is crucial for applications in animation, virtual reality, and human-computer interaction, where producing realistic and diverse human motions based on desired actions enhances user experience and enables more natural interactions.

One of the fundamental challenges in action-conditioned motion synthesis is generating motions that are semantically consistent with the given action labels while exhibiting realistic, physically plausible movements. Additionally, the synthesized motions should capture the diversity of possible movements within the same action category, reflecting

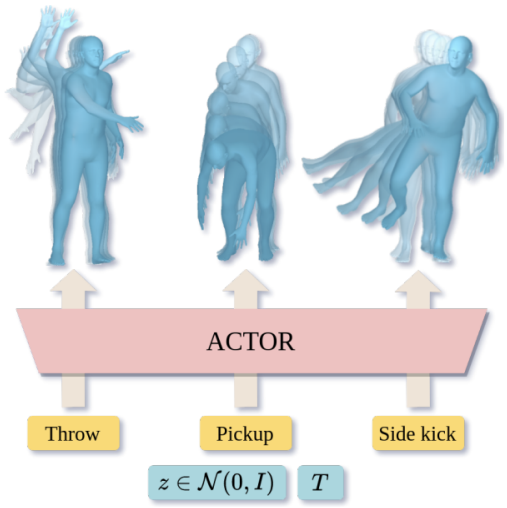


Figure 2.9: Action-Conditioned Human Motion Synthesis. Figure from (Petrovich et al., 2021)

variations in style and execution. An illustration of action-conditioned motion synthesis is show in Figure 2.9

Petrovich et al. (2021) addressed these challenges by introducing the ACTOR model (Action-Conditioned Transformer VAE), a novel framework that combines Transformer architectures with variational autoencoders (VAEs) for action-conditioned 3D human motion synthesis. The key contributions of ACTOR include its use of a *Transformer-based architecture*, which allows for capturing long-range temporal dependencies in motion sequences, significantly improving the coherence and fluidity of the generated motions. Transformers, as sequence models, excel at handling complex motion patterns over extended time frames. Additionally, the model incorporates a *variational autoencoder framework*, which introduces stochastic sampling from the learned latent space. This approach promotes variability in the synthesized motions, enabling the generation of multiple distinct motion samples for the same action label.

Furthermore, ACTOR integrates *action conditioning* into the motion generation process through a dedicated mechanism, ensuring that the motions are semantically aligned with the given actions. This conditioning mechanism is crucial for generating realistic and relevant action-specific motions. The *high-quality motion generation* demonstrated in experiments reveals that ACTOR produces smooth transitions and physically plausible motions, often outperforming prior methods in both qualitative and quantitative evaluations. While Transformers address some limitations of recurrent neural networks (RNNs) by effectively modeling long-term dependencies, the VAE introduces a degree of stochasticity, beneficial for generating diverse samples. However, the model's reliance on large datasets and substantial computational resources may restrict its accessibility

for certain applications.

Guo and Joo (2020) proposed *Action2Motion*, another prominent framework for generating 3D human motions conditioned on action labels. Unlike ACTOR, Action2Motion employs a *generative adversarial network (GAN)* to model the distribution of human motions within each action category. The adversarial training of the GAN encourages the generator to produce realistic motions that closely resemble real-world data, while the discriminator is trained to distinguish between generated and real motions. In this framework, *action embeddings* are used to condition the motion generation process, ensuring that the synthesized motions are relevant to the given action label. Action2Motion also promotes diversity in the generated motions by introducing randomness during generation, thereby capturing different styles and executions within the same action category. To enhance realism, the model incorporates mechanisms for *temporal coherence*, which ensure smooth transitions between frames and improve the overall naturalness of the synthesized motions. Despite these advantages, GAN-based models like Action2Motion can face challenges such as *mode collapse* and training instability, and their scalability to handle a broader set of action categories remains a limitation.

The *TEACH* model (Temporal Action Composition for 3D Humans)(Athanasiou et al., 2022) extends action-conditioned motion synthesis by addressing the generation of complex motion sequences composed of multiple action segments. This model allows for *temporal action composition*, where multiple actions are synthesized sequentially, such as generating a motion where a character walks, sits, and then stands in one continuous sequence. TEACH's contribution lies in its *sequence-level conditioning*, which enables fine-grained control over the order and duration of the actions in the synthesized motion. This capability is particularly useful in scenarios requiring long, varied action sequences. Furthermore, TEACH leverages *attention mechanisms* to focus on the most relevant temporal segments during the motion generation process, thus improving the coherence of transitions between different actions. The resulting motion sequences are both realistic and varied, maintaining physical plausibility throughout the sequence.

A comparative analysis of these models reveals significant differences in their architectures and approaches. ACTOR (Petrovich et al., 2021) employs *Transformers* and *VAEs*, leveraging sequence modeling and generative mechanisms to produce diverse and realistic motions. In contrast, Action2Motion (Guo and Joo, 2020) uses *GANs*, focusing on adversarial training to generate plausible motions, though with potential challenges like mode collapse. TEACH(Athanasiou et al., 2022) stands out for its ability to handle sequences of multiple actions, making use of attention mechanisms to manage the complexity of transitions between different actions.

In terms of *action conditioning*, while ACTOR and Action2Motion focus on generating motions based on single actions, TEACH expands this to sequences of actions, offering more granular control over the motion generation process. Both ACTOR and TEACH explicitly address *motion diversity* through their respective use of stochastic sampling and attention mechanisms, while Action2Motion relies on the adversarial training frame-

work to promote diversity.

Nevertheless, there are notable challenges associated with these models. GAN-based models like Action2Motion may struggle with *training stability*, while Transformer-based models such as ACTOR require substantial computational resources and large datasets. The increased complexity of models like TEACH necessitates more sophisticated training procedures, and capturing the nuances of *action transitions* remains a difficult problem. Moreover, ensuring the *physical plausibility* of synthesized motions, particularly across complex action sequences, is an ongoing area of research.

In conclusion, action-conditioned motion synthesis has made significant progress with the development of models such as ACTOR, Action2Motion, and TEACH. These models contribute to generating realistic and diverse human motions based on action labels, addressing challenges in temporal modeling, diversity, and action composition. Future research may focus on improving scalability, enhancing the handling of complex action sequences, ensuring physical plausibility through biomechanics, and reducing reliance on large datasets.

Text-Conditioned Motion Synthesis

Text-conditioned motion synthesis is an emerging field that aims to generate realistic human motion sequences based on textual descriptions. This problem setting bridges the gap between natural language and human motion, enabling applications in animation, virtual reality, and human-computer interaction where users can generate or control character motions through intuitive textual commands. The complexity of the task arises from the need to understand and interpret diverse linguistic inputs and to map them to corresponding plausible motions, which involves handling high-dimensional motion data, temporal dependencies, and ensuring physical plausibility.

One of the foundational works in this area is TEMOS (Text-Embedded Motion Synthesis) by Petrovich et al. (2022). TEMOS introduces a model that generates diverse human motions from textual descriptions. The authors propose a method that learns a shared embedding space for text and motion, allowing the model to interpret textual inputs and generate corresponding motion sequences.

TEMOS employs a variational autoencoder (VAE) framework, where both textual descriptions and motion sequences are encoded into a shared latent space. This shared space enables the model to learn the associations between language and motion, facilitating the generation of motions that are semantically aligned with the input text. The use of a VAE allows the model to capture the stochastic nature of human motion, enabling the generation of diverse motions from the same textual description.

Moreover, TEMOS incorporates Transformer architectures to model the temporal dependencies in motion sequences. Transformers have demonstrated effectiveness in capturing long-range dependencies in sequential data, which is crucial for generating coherent and fluid motions over time. The model is trained on a dataset of paired textual descriptions

and motion sequences, learning to generate motions that correspond to the given text.

While TEMOS makes significant contributions in mapping textual descriptions to motion sequences, it faces limitations. The model's performance depends heavily on the quality and diversity of the training data. Handling ambiguous or abstract textual descriptions remains challenging, and the reliance on large datasets and computational resources may limit its applicability in some settings.

Another notable work is T2M-GPT (Zhang et al., 2023a), which focuses on generating human motion from textual descriptions using discrete representations. T2M-GPT treats motion generation as a sequence-to-sequence problem, leveraging advances in language models, particularly GPT-like architectures. By discretizing motion data into tokens, the model applies language modeling techniques to motion sequences, allowing it to generate motions conditioned on textual inputs.

The discrete representation of motions in T2M-GPT simplifies the mapping between text and motion but may lead to a loss of fine-grained motion details. Moreover, training large language models requires substantial computational resources, and the approach may struggle with complex or lengthy textual descriptions.

Zhang and Wang (2022) introduced MotionDiffuse, applying diffusion models to text-driven human motion generation. Diffusion models have shown promise in generating high-quality data by modeling complex distributions through iterative denoising processes. MotionDiffuse leverages this capability to generate motion sequences that are both realistic and aligned with textual descriptions.

In MotionDiffuse, the model starts from random noise and iteratively refines it into coherent motion data, guided by the textual input. The text conditioning is integrated into the diffusion process, influencing the denoising steps to produce motions that correspond to the input descriptions. While diffusion models excel in capturing complex data distributions, their iterative nature results in slow generation times, which may not be suitable for real-time applications.

Building upon the principles of MotionDiffuse, the Human Motion Diffusion Model advances the application of diffusion models in motion synthesis (Tevet et al., 2022b). This model focuses on generating human motions conditioned on textual descriptions, improving upon previous methods by implementing state-of-the-art diffusion techniques and optimizing the process for motion data.

The Human Motion Diffusion Model enhances the quality and diversity of generated motions, achieving superior performance in both qualitative and quantitative evaluations. However, similar to MotionDiffuse, the computational intensity and slow generation times pose challenges for practical applications.

MotionCLIP represents another significant advancement in text-conditioned motion synthesis by leveraging the Contrastive Language-Image Pre-training (CLIP) model (Radford et al., 2021). MotionCLIP uses CLIP as a text prior to guide motion generation,

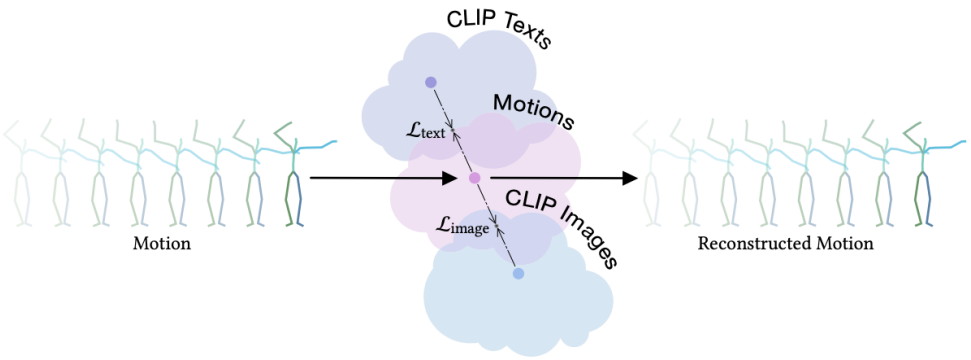


Figure 2.10: MotionCLIP overview. Figure from (Tevet et al., 2022a).

aligning motion data with text embeddings in a shared latent space. By benefiting from CLIP's extensive pre-training on diverse text-image pairs, MotionCLIP inherits rich semantic representations, enabling it to generate motions that correspond to textual descriptions (Tevet et al., 2022a). The overview of MotionCLIP is shown in Figure 2.10.

One of the key advantages of MotionCLIP is its ability to perform zero-shot generalization, generating motions for unseen textual descriptions without explicit training on specific action labels. This is particularly valuable in applications where new or rare actions need to be synthesized. However, the reliance on CLIP introduces potential limitations, such as inheriting biases present in the pre-trained model and the challenge of aligning different data modalities (text and motion).

The models discussed exhibit various approaches to bridging the semantic gap between text and motion. While TEMOS and MotionCLIP focus on learning shared embedding spaces, MotionDiffuse and the Human Motion Diffusion Model leverage diffusion probabilistic models to capture complex motion distributions. T2M-GPT applies language modeling techniques to discrete motion representations.

Despite the progress, several challenges persist in text-conditioned motion synthesis. Handling ambiguous or abstract textual descriptions requires models to have a deeper understanding of language semantics and context. Ensuring diversity and realism in generated motions involves balancing the stochasticity and coherence of motions. Computational efficiency remains a concern, especially for models like diffusion models that are computationally intensive.

In comparing these models, MotionCLIP and the Human Motion Diffusion Model represent different approaches with their respective strengths and limitations. MotionCLIP leverages a pre-trained model (CLIP) to bridge the gap between text and motion, enabling zero-shot generalization. The Human Motion Diffusion Model focuses on modeling the complex distribution of motion data through diffusion techniques, achieving

high-quality and diverse motion generation. MotionCLIP benefits from the extensive knowledge encoded in CLIP but may inherit biases and face challenges in aligning different data modalities. The Human Motion Diffusion Model excels in generating realistic motions but is computationally intensive and may not be practical for applications requiring fast generation times.

Future research directions may focus on combining the strengths of these approaches. Integrating pre-trained language models or embeddings into diffusion models could enhance their textual understanding while leveraging the high-quality generation capabilities of diffusion techniques. Additionally, ensuring that models can handle diverse and complex textual descriptions, generate physically plausible motions, and operate efficiently remains an ongoing challenge.

In conclusion, text-conditioned motion synthesis has advanced significantly with the introduction of models like TEMOS, MotionCLIP, MotionDiffuse, and others. These models have enhanced the ability to generate realistic and diverse human motions from textual inputs, addressing some of the challenges in semantic alignment and motion generation. Future work may involve improving computational efficiency, enhancing the understanding of complex textual descriptions, reducing dependency on large datasets through techniques like transfer learning, and addressing ethical considerations such as bias and cultural sensitivity in generated motions.

Music-Conditioned Human Motion Synthesis

Music-conditioned human motion synthesis is a specialized field that generates human movements, particularly dance, in response to musical inputs. This area combines elements of audio processing, motion synthesis, and sometimes style transfer to create animations where characters move in synchronization with music. The inherent challenge lies in interpreting the temporal and emotive features of music and translating them into coherent and expressive motions.

Early efforts in this domain often relied on rule-based systems or handcrafted mappings between musical features and motion parameters. However, these approaches struggled to capture the complexity and diversity of both music and dance. Recent advancements leverage deep learning to model the intricate relationships between auditory and motion data.

One significant contribution is the development of models that utilize large datasets of paired music and motion data. The AIST++ dataset, introduced by Li et al., provides a comprehensive collection of 3D dance motions synchronized with music (Li et al., 2021). This dataset has enabled the training of models that can learn direct mappings from music to motion. The "AI Choreographer" model builds upon this dataset, employing a sequence-to-sequence architecture with attention mechanisms to generate dance motions conditioned on music (Li et al., 2021). By extracting features from both music and motion sequences, the model captures temporal dependencies and correlations between

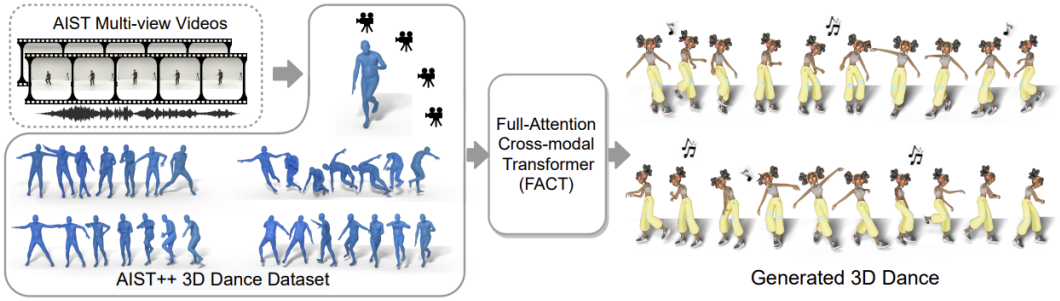


Figure 2.11: AI Choreographer. Figure from (Li et al., 2021)

the two modalities. This approach allows for the generation of dance movements that are synchronized with the rhythm and mood of the music, as shown in Figure 2.11.

Another advancement is the use of Transformer architectures in models that learning to generate dance motions with Transformer(Huang et al., 2020). Transformers excel at modeling long-range dependencies, making them suitable for capturing the temporal structure of music and motion. By incorporating variational techniques, these models also address the need for diversity in generated motions, producing multiple plausible dances for a single piece of music.

Bailando introduces an innovative approach for 3D dance generation by integrating reinforcement learning with generative modeling (Ren et al., 2022). The model utilizes an actor-critic framework where the actor generates dance motions and the critic evaluates their quality and adherence to the music. The inclusion of a choreographic memory enhances the model's ability to produce diverse and contextually appropriate dance sequences.

Despite these advancements, music-conditioned motion synthesis faces several challenges. One major limitation is the dependency on large datasets like AIST++, which may not cover all dance styles or musical genres. Models trained on such datasets might not generalize well to unseen styles or complex musical compositions. Additionally, ensuring that generated motions are physically plausible and emotionally expressive remains difficult.

The translation of musical features into motion is inherently subjective, as different individuals may interpret the same piece of music differently in terms of movement. Capturing this variability requires models to account for personal and cultural differences in musical interpretation.

Future research may explore unsupervised or semi-supervised learning techniques to reduce the reliance on extensive labeled data. Incorporating higher-level musical features, such as emotional content or genre-specific characteristics, could improve the expressiveness of generated motions. Cross-modal representations that jointly model audio and

motion data may also enhance the model's ability to capture the nuanced relationships between music and movement.

Style and Scene-Aware Motion Synthesis

The synthesis of human motion that is both stylistically rich and aware of the surrounding scene is a critical area in computer graphics and animation. This domain focuses on generating motions that not only exhibit specific stylistic attributes but also interact naturally with the environment, enhancing realism and immersion in virtual worlds. The complexity arises from the need to model the intricate relationships between a character's movements, their stylistic nuances, and the physical constraints imposed by the environment.

Scene-aware motion synthesis addresses the challenge of generating motions that are coherent with the environment. Early approaches often treated motion generation and scene understanding as separate problems, leading to animations where characters might collide with objects or move unrealistically within the scene. Recent advancements aim to integrate scene context directly into the motion synthesis process.

Wang et al. (2021c) made significant contributions in this area by proposing a scene-aware generative network for human motion synthesis. Their model integrates environmental information into the motion generation pipeline, enabling the synthesis of motions that respect scene constraints such as obstacles and terrain variations. By encoding the scene using convolutional neural networks and conditioning the motion generation on this encoding, the model produces motions that are contextually appropriate.

This integration of scene context into motion synthesis allows for more realistic character interactions within complex environments. However, challenges remain in generalizing to highly varied scenes and ensuring that the model can handle dynamic environments where the scene changes over time. The reliance on accurate scene representations also means that any errors in scene perception can lead to unrealistic motions.

Motion style transfer focuses on altering the stylistic aspects of a motion sequence while preserving its fundamental structure and intent. This is particularly useful for animators who wish to apply different styles to a base motion without recreating it from scratch. Style transfer in motion synthesis involves learning representations of style that can be manipulated and applied to different motions.

Wang et al. (2021b) explored motion style transfer between humans and robots, highlighting the complexities of transferring styles across domains with different physical characteristics. Their work demonstrates that by learning a shared latent space, it is possible to transfer human motion styles to robotic motions, enabling robots to exhibit more natural and expressive movements.

The primary challenge in motion style transfer lies in defining and capturing the essence of "style" in a quantifiable manner. Styles can be subtle and multifaceted, encompassing variations in speed, amplitude, and fluidity. Models must disentangle style from content

to manipulate it effectively. Furthermore, transferring styles between entities with different kinematics, such as humans and robots, introduces additional complexities due to differences in joint structures and movement capabilities.

Combining style transfer with scene awareness leads to the synthesis of motions that are both stylistically rich and environmentally coherent. The work by Wang et al. (2022) advances this integration by exploring techniques that jointly model style and scene context. This approach aims to generate a diverse set of natural motions that are appropriate for the given scene and desired style, enhancing the versatility and applicability of motion synthesis models.

Despite these advancements, several limitations persist in style and scene-aware motion synthesis. Models often require large datasets that cover a wide range of styles and scenes to generalize effectively. Data scarcity in certain styles or environments can lead to biased models that do not perform well outside the training distribution. Additionally, the computational demands of processing complex scenes and styles can hinder real-time applications.

Future research directions may focus on improving the efficiency of these models and developing methods to learn from limited data. Techniques such as transfer learning and domain adaptation could help models generalize better across different styles and scenes. Incorporating physical constraints and biomechanics could also enhance the realism of synthesized motions, ensuring that generated movements are not only visually plausible but also physically feasible.

2.3 Methodologies in Human Motion Synthesis

The development of human motion synthesis models has been significantly influenced by advancements in machine learning methodologies. Various approaches have been explored to capture the complex dynamics of human motion, each leveraging different aspects of sequential data modeling, latent variable representation, and adversarial training. This section delves into the predominant methodologies employed in human motion synthesis, critically analyzing recurrent neural network-based models, variational autoencoders, and generative adversarial networks. The discussion highlights the evolution of these methods, their contributions to the field, comparative strengths and weaknesses, and insights into potential future directions.

2.3.1 Model Architectures for Human Motion Synthesis

Human motion synthesis has progressed significantly, driven by the development of diverse model architectures that address the inherent complexity, variability, and temporal dependencies of human movement. Early neural network-based approaches, such as Recurrent Neural Networks (RNNs), Variational Autoencoders (VAEs), and Generative Adversarial Networks (GANs), laid the groundwork by modeling sequential motion data,

but have since been complemented by more advanced techniques. Transformers and diffusion models, for instance, have emerged as powerful alternatives that effectively capture long-range dependencies and generate higher-quality motions. Additionally, multimodal learning—leveraging multiple input modalities like text, audio, and visual data—has further enriched the field, allowing models to synthesize motions that are contextually aligned with diverse inputs. This section delves into the principal model architectures shaping human motion synthesis today, focusing on neural networks, Transformers, diffusion models, multimodal approaches, and other emerging innovations. By analyzing these models, we aim to uncover their individual contributions, strengths, and limitations, illustrating how they collectively push the boundaries of motion generation.

Neural Network-Based Approaches

Neural network-based approaches have made significant contributions to the development of human motion synthesis, particularly in tackling the challenges posed by high-dimensional and temporally dependent motion data. These models, including Recurrent Neural Networks (RNNs), Variational Autoencoders (VAEs), and Generative Adversarial Networks (GANs), provided foundational techniques for sequential data modeling and motion generation. RNNs, despite limitations in capturing long-range dependencies, were among the first to address temporal continuity in human motion. VAEs, on the other hand, remain widely used due to their ability to model complex motion distributions and generate diverse, realistic motion sequences through latent space representations. GANs introduced an adversarial framework that has been valuable in generating realistic motion, but they can be challenging to train and often suffer from instability. While neural network-based approaches are now supplemented by newer architectures like Transformers and diffusion models, they continue to play a critical role, especially in hybrid frameworks that combine their strengths with more advanced methodologies. In this subsection, we explore the key methodologies of RNNs, VAEs, and GANs, examining their impact on human motion synthesis and their roles in addressing challenges such as temporal dependencies, motion variability, and generative realism.

Recurrent Neural Network-Based Models

Recurrent Neural Networks (RNNs) have been fundamental in modeling sequential data due to their ability to capture temporal dependencies. In human motion synthesis, RNNs have been utilized to model the sequential nature of motion capture data, enabling the generation of realistic and coherent motion sequences.

Chung et al. (2015) introduced a recurrent latent variable model for sequential data, integrating stochastic latent variables into RNNs to capture the variability in sequences. Their model, known as the Variational RNN (VRNN), extends the traditional RNN by incorporating a variational autoencoder (VAE) framework at each time step. This integration allows the model to capture complex temporal dependencies and generate diverse sequences by sampling from the latent space.

The VRNN addresses limitations in standard RNNs, which often struggle with modeling

sequences that exhibit high variability due to their deterministic nature. By introducing stochastic latent variables, the VRNN captures both the temporal dynamics and the inherent uncertainty in human motion. However, training such models can be challenging due to the complexities involved in optimizing variational objectives and the potential for vanishing gradients over long sequences.

Building on the strengths of recurrent models, Harvey et al. (2020) proposed Recurrent Transition Networks (RTNs) for character locomotion. Their approach focuses on modeling the transitions between different locomotion states, such as walking, running, and jumping. By structuring the model to learn transitions explicitly, RTNs capture the nuances of motion changes and generate smoother and more realistic locomotion sequences.

The RTN architecture leverages gated recurrent units (GRUs) to manage temporal dependencies while incorporating transition functions that model the probabilistic transitions between states. This design allows the model to handle varying motion lengths and complex dynamics. Nonetheless, RTNs may face difficulties in scaling to motions with a large number of distinct states or in capturing long-term dependencies beyond immediate transitions.

Zhang, Cao, and Pons-Moll (2021) further advanced RNN-based methodologies by learning motion manifolds with sequential networks **?**. They proposed a framework that models human motion as trajectories on a low-dimensional manifold, capturing the intrinsic structure of motion data. By utilizing sequential networks, the model learns to represent motions in a latent space where interpolation and manipulation become more straightforward.

This manifold learning approach enables the generation of smooth and continuous motions by interpolating within the learned latent space. It also facilitates style transfer and motion editing by navigating the manifold. However, accurately learning the motion manifold requires extensive and diverse motion data, and the model might struggle with extrapolating beyond the regions covered by the training data.

Variational Autoencoders (VAEs)

Variational Autoencoders have been instrumental in modeling complex data distributions by learning latent representations that capture the underlying structure of the data. In human motion synthesis, VAEs enable the generation of diverse and realistic motions by sampling from the learned latent space.

Ling et al. (2020) employed motion VAEs for character controllers, integrating VAEs into the control mechanisms of animated characters. Their approach involves training a VAE on motion capture data to learn a latent space of motions. The character controller then operates within this latent space, allowing for the generation of motions that are both coherent and responsive to control inputs.

By leveraging the VAE's ability to model the variability in motion data, the character

controller can produce a wide range of motions while maintaining physical plausibility. This methodology enhances the flexibility and expressiveness of animated characters. However, VAEs can suffer from issues such as blurry outputs or mode collapse if not properly regularized. Additionally, ensuring that the latent space aligns well with control inputs requires careful design and training.

Generative Adversarial Networks (GANs)

Generative Adversarial Networks have gained prominence in generating high-fidelity data by framing the generation process as a game between a generator and a discriminator. In human motion synthesis, GANs have been used to produce realistic motions conditioned on various inputs.

Guo and Joo (2020) introduced Action2Motion, a GAN-based model for the conditioned generation of 3D human motions. Their model generates motion sequences conditioned on action labels, enabling the synthesis of motions corresponding to specific activities. The generator produces motion sequences, while the discriminator evaluates their realism and adherence to the conditioned action.

Action2Motion addresses the challenge of generating diverse motions for the same action by incorporating a stochastic component in the generator. This allows the model to produce multiple plausible motions for a given action label. The adversarial training ensures that the generated motions are realistic and action-consistent. However, GANs are known for training instability and mode collapse, where the generator produces limited variations. Balancing the adversarial loss and ensuring diversity in outputs can be challenging.

Comparative Analysis and Insights

The methodologies discussed—RNN-based models, VAEs, and GANs—each offer unique strengths in modeling human motion synthesis. RNN-based models excel in capturing temporal dependencies and handling sequential data, making them suitable for modeling the dynamics of motion sequences. The integration of latent variables in models like VRNN enhances their ability to capture variability but introduces training complexities.

VAEs provide a powerful framework for learning latent representations that capture the underlying structure of motion data. They facilitate diversity in generated motions and enable manipulation within the latent space. However, VAEs may produce less sharp outputs compared to GANs and require careful regularization to prevent issues like mode collapse.

GANs are adept at generating high-fidelity data and ensuring realism through adversarial training. In the context of motion synthesis, GANs like Action2Motion can produce realistic and diverse motions conditioned on specific inputs. Nonetheless, GANs can be unstable during training and may struggle with generating sufficient diversity without careful design.

Combining these methodologies may offer pathways to overcome individual limitations.

For instance, integrating VAE frameworks with GANs (e.g., VAE-GANs) can leverage the strengths of both models, capturing variability and ensuring output realism. Additionally, incorporating recurrent structures into GANs can enhance their ability to model temporal dependencies in motion sequences.

Future research directions may focus on developing hybrid models that integrate the strengths of RNNs, VAEs, and GANs. Addressing training challenges, such as stabilizing GAN training and optimizing variational objectives in sequential contexts, remains critical. Exploring attention mechanisms and transformer architectures may also enhance the modeling of long-term dependencies and complex motion patterns.

Transformer-Based Models

The advent of Transformer architectures has revolutionized sequence modeling in various domains, notably in natural language processing and computer vision. Transformers, introduced by Vaswani et al. (2017b), rely entirely on attention mechanisms to model dependencies within sequences, eschewing recurrence and convolution. Their capability to capture long-range dependencies and model complex sequential data has made them particularly suitable for human motion synthesis, where capturing temporal dynamics and spatial correlations is crucial.

In human motion synthesis, Transformers have been leveraged to address the challenges of modeling high-dimensional motion data with intricate temporal and spatial dependencies. By employing attention mechanisms, these models can focus on relevant parts of the input sequence when generating each part of the output, allowing for more accurate and coherent motion generation.

Petrovich et al. (2021) extended the Transformer architecture by integrating it with a Variational Autoencoder (VAE) in their work on action-conditioned 3D human motion synthesis. Their model, known as Transformer VAE, combines the strengths of Transformers in capturing long-range dependencies with the ability of VAEs to model data distributions and generate diverse outputs. By conditioning on action labels, the model generates motion sequences that are not only realistic but also aligned with the specified actions.

The Transformer VAE operates by encoding motion sequences into a latent space using the Transformer encoder and then decoding from this space to reconstruct or generate new motion sequences. The use of a VAE allows for sampling from the latent space, enabling the generation of diverse motions for the same action. The attention mechanisms within the Transformer facilitate the capture of both temporal and spatial dependencies in motion data.

Aksan et al. (2020) introduced a spatio-temporal Transformer for 3D human motion prediction. Their model focuses on predicting future motions based on past observations, a task that requires understanding both the temporal evolution and the spatial configuration of human poses. By utilizing a Transformer architecture, the model attends

to relevant time steps and joint relationships, effectively modeling the spatio-temporal dynamics of human motion.

Their approach addresses the limitations of recurrent neural networks (RNNs) and convolutional neural networks (CNNs) in capturing long-term dependencies and complex spatial correlations. The attention mechanism allows the model to weigh the importance of different joints and time steps dynamically, leading to more accurate motion predictions. However, Transformers require substantial computational resources and large amounts of data to train effectively, which can be a limitation in motion synthesis tasks where data may be scarce.

Li and Zhang (2019) explored the use of Transformers in generating diverse dance motions conditioned on music. By treating dance motion generation as a sequence-to-sequence problem, they employed a Transformer architecture to model the relationship between music and dance movements. The model captures the temporal alignment between musical beats and dance motions, generating realistic and synchronized dance sequences.

Their work demonstrates the ability of Transformers to handle multimodal inputs and outputs, leveraging attention mechanisms to align features from different domains. The challenge lies in capturing the nuances of dance styles and ensuring diversity in the generated motions. The model addresses this by incorporating stochasticity in the generation process, allowing for the creation of multiple plausible dance sequences for the same musical input.

Furthermore, the work "Bailando: 3D Dance Generation by Actor-Critic GPT with Choreographic Memory" further extends the application of Transformer-based models in motion synthesis (Ren et al., 2022). The authors utilize a Generative Pre-trained Transformer (GPT) architecture within an actor-critic framework to generate dance motions conditioned on music. The GPT model, pre-trained on large motion datasets, acts as the generator (actor), while the critic evaluates the quality and coherence of the generated motions.

The integration of choreographic memory allows the model to recall and incorporate complex dance patterns, enhancing the diversity and realism of the generated motions. The actor-critic setup facilitates learning through reinforcement, with the critic providing feedback to the generator. This approach highlights the adaptability of Transformer models in different training paradigms, including reinforcement learning.

In T2M-GPT, the authors leverage GPT-based models to generate human motions from textual inputs (Zhang et al., 2023a). By treating motion sequences as discrete tokens similar to words in language modeling, the model applies techniques from natural language processing to motion synthesis. The Transformer architecture captures the relationships between textual descriptions and motion tokens, generating motions that correspond to the input text.

This methodology treats human motion as a "foreign language," as further explored in

MotionGPT (Jiang et al., 2023). By framing motion generation as a language modeling problem, the model benefits from the advancements in Transformer architectures and pre-training strategies in NLP. This perspective allows for the utilization of large-scale pre-training on motion data, improving the model's ability to generate coherent and realistic motions.

The use of Transformers in human motion synthesis brings several advantages. The attention mechanism enables the modeling of long-range dependencies and complex relationships within motion data. Transformers are inherently parallelizable due to their non-recurrent structure, allowing for efficient training on large datasets. Additionally, their flexibility in handling different types of input and output sequences makes them suitable for various motion synthesis tasks, including action-conditioned generation, motion prediction, and multimodal synthesis involving text or music.

However, Transformers also present challenges. Their reliance on large amounts of data for effective training can be a limitation in domains where annotated motion data is limited. The computational resources required for training and inference are substantial, potentially hindering real-time applications. Overfitting can occur if the model complexity is not adequately managed, especially when dealing with high-dimensional motion data.

Future research may focus on addressing these limitations by developing more data-efficient training methods, such as transfer learning and unsupervised pre-training. Incorporating domain-specific inductive biases into the Transformer architecture could improve performance on motion synthesis tasks with limited data. Exploring hybrid models that combine Transformers with other architectures, such as convolutional or recurrent networks, may also enhance the modeling of spatial and temporal dependencies.

In conclusion, Transformer-based models have significantly advanced the field of human motion synthesis by providing powerful tools for modeling complex sequential data. Their ability to capture intricate dependencies and handle various input modalities has enabled the development of models that generate realistic and diverse human motions. Continued research in this area promises further improvements in motion synthesis methodologies, with the potential to overcome current limitations and expand the applicability of these models in real-world applications.

Diffusion Models

The application of diffusion models in human motion synthesis represents a significant advancement in generative modeling, offering a powerful framework for capturing the complex, high-dimensional distributions inherent in human motion data. Diffusion models, initially introduced in the context of image generation, have been adapted to motion synthesis, leveraging their iterative denoising processes to generate realistic and diverse motion sequences.

Diffusion models operate by progressively transforming a simple noise distribution into

the target data distribution through a series of learned denoising steps. This process enables the model to capture intricate data structures and temporal dependencies, making them particularly suitable for modeling the stochastic nature of human motion.

Zhang and Wang (2022) pioneered the application of diffusion models to text-driven human motion generation with their work *MotionDiffuse*. They demonstrated that diffusion probabilistic models could effectively generate human motion sequences conditioned on textual descriptions. By integrating text conditioning into the diffusion process, the model produces motions that are semantically aligned with the input text, capturing both spatial configurations and temporal dynamics.

The diffusion process in MotionDiffuse begins with random noise and iteratively refines it towards a coherent motion sequence. At each denoising step, textual information guides the model, ensuring that the generated motions correspond to the semantic content of the input description. This approach allows for the synthesis of high-quality motions that are both realistic and contextually appropriate.

Building upon this foundation, *MoFusion* introduced a framework for denoising-diffusion-based motion synthesis (Dabral et al., 2023). MoFusion extends the application of diffusion models by focusing on generating diverse and physically plausible motions. The framework emphasizes modeling the multimodal nature of human motion, where multiple plausible motions can correspond to the same conditioning input. By capturing this variability, MoFusion enhances the diversity of generated motions, which is crucial for applications requiring a range of motion options.

The *Human Motion Diffusion Model* further advances the field by exploring diffusion models as a generative prior in motion synthesis tasks **?**. By treating the diffusion model as a prior, this approach facilitates the incorporation of additional constraints or conditioning information, such as physical constraints or user inputs. This perspective allows for greater flexibility and control in the motion generation process, enabling the model to generate motions that adhere to specific requirements while maintaining realism.

Incorporating physics into diffusion-based motion models, *PhysDiff* introduces a physics-guided human motion diffusion model (Yuan et al., 2023). By integrating physical constraints directly into the diffusion process, PhysDiff ensures that the generated motions comply with the laws of physics, resulting in physically plausible and realistic movements. This integration addresses a common limitation in motion synthesis, where generated motions may appear visually plausible but violate physical principles, such as gravity or momentum conservation.

Chen et al. (2023) explores generating motions based on user commands using diffusion in latent space. By operating in the latent space of the diffusion model, the approach efficiently generates motions that satisfy specified commands while maintaining coherence and realism. This method enhances user interaction with the model, allowing for more intuitive control over the generated motions.

ReMoDiffuse introduces a retrieval-augmented motion diffusion model (Zhang et al.,

2023b). By integrating a retrieval mechanism, the model leverages a database of real motion sequences to inform the diffusion process. This approach grounds the generation in actual motion data, enhancing the diversity and quality of the generated motions. The retrieval component helps the model to produce motions that are both novel and consistent with real human movements.

The adoption of diffusion models in human motion synthesis offers several advantages. Firstly, they are capable of capturing complex data distributions, enabling the generation of high-fidelity and diverse motions. The iterative denoising process refines coarse motion representations into detailed and realistic sequences. Secondly, the flexibility of the diffusion framework allows for the incorporation of various conditioning information, such as text descriptions, physics constraints, or user commands, facilitating controlled motion generation.

However, diffusion models also present challenges. A primary limitation is the computational intensity of the iterative denoising process, which can result in slow generation times unsuitable for real-time applications. Training diffusion models requires significant computational resources and large datasets to effectively learn the intricate data distributions. Additionally, ensuring that generated motions are both physically plausible and semantically aligned with conditioning inputs remains a complex task, necessitating careful model design and integration of domain knowledge.

Comparatively, diffusion models offer a distinct approach compared to other methodologies like RNN-based models, VAEs, GANs, or Transformer-based models. While RNNs and VAEs focus on sequential modeling and latent variable representations, and GANs employ adversarial training to produce realistic outputs, diffusion models directly model the data distribution through a Markov chain of reversible transformations. This direct modeling allows diffusion models to capture the full complexity of the data distribution without relying on adversarial training, which can be unstable.

The incorporation of physics into diffusion models, as seen in PhysDiff, represents a significant advancement in addressing the physical plausibility of generated motions. By embedding physics-based constraints into the generative process, the model produces motions that adhere to physical laws, enhancing the realism and applicability of the synthesized motions in practical applications such as animation and simulation.

Future research directions in diffusion-based motion synthesis may focus on improving computational efficiency to enable real-time applications. Techniques such as model distillation, acceleration of the denoising process, or approximation methods could reduce generation times. Additionally, integrating diffusion models with other methodologies, such as combining Transformer architectures with diffusion processes, may enhance the modeling of complex temporal dependencies and improve the quality of generated motions.

Moreover, expanding the conditioning capabilities of diffusion models to incorporate multimodal inputs, user interactions, or contextual information could further enhance

their versatility. Addressing data requirements through techniques like data augmentation, unsupervised learning, or transfer learning may mitigate reliance on large datasets, making diffusion models more accessible and practical in various settings.

In summary, diffusion models have emerged as a powerful methodology in human motion synthesis, offering the ability to generate high-quality, diverse, and controllable motions. Their application has advanced the field by addressing key challenges in modeling complex data distributions and incorporating various conditioning information. Continued research and development in diffusion-based motion synthesis hold promise for further advancements in generating realistic, physically plausible human motions that meet the demands of increasingly sophisticated applications.

Multimodal Learning and Cross-Modal Representations

Multimodal learning and cross-modal representations have significantly advanced human motion synthesis by enabling models to interpret and generate motions conditioned on various input modalities such as text, audio, and visual cues. By integrating information from multiple sources, these approaches aim to bridge semantic gaps between different data types, allowing for the synthesis of realistic and contextually appropriate human motions.

Radford et al. (2021) introduced Contrastive Language-Image Pre-training (CLIP), a model trained on a vast dataset of image-text pairs using a contrastive learning objective. CLIP learns a shared embedding space where images and their corresponding textual descriptions are mapped closely together, facilitating cross-modal retrieval and understanding. This shared space has been leveraged in human motion synthesis to align language and motion modalities.

Building on CLIP, Tevet et al. (2022a) developed *MotionCLIP*, which exposes human motion generation to the CLIP space. *MotionCLIP* aligns motion sequences with textual descriptions in a unified embedding space, enabling text-driven motion synthesis and retrieval. By mapping motion data into CLIP space using a motion encoder, the model allows for zero-shot generalization, generating motions for unseen textual inputs. However, ensuring fine-grained alignment between motion nuances and textual semantics remains challenging.

Kim et al. (2022)) proposed *FLAME*, a framework for free-form language-based motion synthesis and editing. *FLAME* uses a transformer-based architecture to directly map natural language descriptions to motion sequences, facilitating intuitive motion generation and manipulation. The model handles diverse language inputs beyond predefined action labels, but accurately capturing complex linguistic nuances and requiring large paired datasets are notable challenges.

Baltrušaitis et al. (2019) provided a comprehensive survey of multimodal machine learning methods, highlighting key challenges such as representation learning, translation between modalities, and alignment. These challenges are pertinent to human motion

synthesis, where models must align and translate between modalities like language and motion.

For audio modalities, Li et al. (2021) introduced *AI Choreographer*, focusing on generating dance motions conditioned on music inputs. The model captures temporal and stylistic relationships between music and motion, generating synchronized dance sequences. Challenges include ensuring temporal alignment and handling the subjectivity in musical interpretation.

Tevet et al. (2022) applied contrastive learning to text-to-motion retrieval in *TMR* **?**. By learning a shared embedding space for text and motion, the model retrieves motion sequences corresponding to textual descriptions. While improving alignment between modalities, fine-grained semantic matching and handling ambiguous language inputs remain difficult.

Insights and Future Directions

Key insights in multimodal learning for motion synthesis highlight the importance of shared embedding spaces for cross-modal alignment, contrastive learning for improving embedding and model quality, and Transformer architectures for handling sequential data with long-range dependencies. However, effective multimodal models often require large datasets, underscoring the need for data-efficient approaches. Future research should focus on leveraging unpaired or weakly paired data to address data scarcity, enhancing semantic understanding of complex inputs, integrating domain knowledge from fields like linguistics and biomechanics, and addressing bias to ensure ethical applications. Overall, multimodal learning advances human motion synthesis by enabling realistic, diverse motion generation and holds promise for further enhancing model capabilities.

Other Models

Several alternative models have advanced human motion synthesis by tackling specific challenges and introducing unique frameworks. Li et al. (2022) utilized skeletal graph neural networks (GNNs) to capture spatial dependencies in 3D pose estimation, enhancing accuracy by modeling joints and bones as a relational graph, thus improving over traditional methods that ignore skeletal structure. In contrast, Holden et al. (2017) introduced Phase-Functioned Neural Networks (PFNNs), dynamically adjusting weights based on motion phase to create responsive and adaptable real-time character animations, making it particularly effective for complex motion control.

While GNNs and PFNNs focus on pose accuracy and motion control, Cai et al. (2021) addressed the gap between motion estimation and synthesis with UNIK, a unified framework that integrates both for more realistic virtual interactions. By combining kinematic constraints and deep learning, UNIK bridges estimation and synthesis, improving consistency in synthesized motion, especially for applications like VR. Extending motion synthesis further, Wang et al. (2021a) developed a style transfer model for motion and appearance, enabling diverse, expressive animations that mimic individual styles using

adversarial training.

Each model addresses unique needs in motion synthesis: GNNs emphasize structural pose accuracy, PFNNs excel in adaptable control, UNIK provides cohesive estimation-synthesis integration, and style transfer enriches personalization and creativity in animation.

2.3.2 Kinematic Constraints and Physical Corrections

Ensuring physical plausibility and realism in synthesized human motions is crucial, especially for applications in animation, robotics, and virtual reality. Domain-specific techniques such as enforcing kinematic constraints and applying physical corrections are employed to address issues like unnatural joint movements, foot-sliding artifacts, and violations of physical laws.

Foot-Sliding Prevention

Foot-sliding is a common artifact in motion synthesis where a character's feet appear to slide on the ground during contact phases, leading to unrealistic animations. Preventing foot-sliding involves enforcing constraints that keep the feet stationary relative to the ground when they are in contact.

Lee et al. (2010) proposed a data-driven method for foot-skate cleanup in motion capture editing. Their approach detects foot contact events and applies inverse kinematics (IK) to adjust joint angles, ensuring that the feet remain fixed during ground contact. By optimizing the motion trajectories while maintaining the overall motion dynamics, the method effectively reduces foot-sliding artifacts without introducing unnatural joint movements. This technique enhances the realism of synthesized motions, particularly in sequences involving complex interactions with the environment.

Joint Constraints and Inverse Kinematics

Enforcing joint limits and utilizing inverse kinematics are essential for generating realistic human motions. Joint constraints prevent the occurrence of anatomically impossible poses by restricting joint angles to physiologically feasible ranges. Inverse kinematics solves for joint configurations that achieve desired end-effector positions, such as placing a hand on an object or maintaining balance.

Yamane and Nakamura (2003a) introduced a dynamics filter for real-time generation of human body motions that integrates physical constraints and joint limits within a dynamics simulation framework. Their method ensures that the generated motions are physically plausible by considering the equations of motion and applying constraints to joint angles and velocities. By incorporating inverse kinematics and dynamics, the approach produces motions that are not only kinematically correct but also dynamically consistent, allowing for responsive and realistic motion generation suitable for real-time applications.

Applying kinematic constraints and physical corrections is vital for enhancing the quality of synthesized motions. These techniques address common artifacts and ensure compliance with physical laws, which is particularly important in applications where motion realism directly impacts user experience or system performance, such as virtual reality simulations, ergonomic studies, and human-robot interaction.

2.3.3 Performance Evaluation

Comparative Analysis

The methodologies discussed in the previous sections represent a diverse range of approaches to human motion synthesis, each leveraging different strengths to address the complex challenges inherent in modeling human movement. This comparative analysis examines these methodologies—RNN-based models, Variational Autoencoders (VAEs), Generative Adversarial Networks (GANs), Transformer-based models, Diffusion models, and Multimodal learning—highlighting their advantages, limitations, and suitability for various tasks within human motion synthesis.

RNN-Based Models excel in capturing temporal dependencies due to their recurrent architecture, making them suitable for sequential data modeling (Chung et al., 2015; Harvey et al., 2020). They effectively handle short to medium-length sequences and can model the dynamics of motion transitions. However, RNNs suffer from limitations such as vanishing gradients and difficulty in capturing long-range dependencies, which can affect their performance on extended motion sequences. Additionally, their sequential nature poses challenges for parallelization and computational efficiency.

Variational Autoencoders (VAEs) provide a probabilistic framework for learning latent representations of motion data (Ling et al., 2020). VAEs enable the generation of diverse motions by sampling from the learned latent space, capturing variability in human movements. They are particularly effective for tasks requiring stochastic generation and interpolation between motions. Nonetheless, VAEs may produce blurred or less sharp outputs compared to other generative models, and they can struggle with preserving fine-grained details in complex motion data.

Generative Adversarial Networks (GANs) are powerful for generating high-fidelity motion sequences (Guo and Joo, 2020). GANs can produce realistic motions conditioned on specific inputs, such as action labels, and are adept at modeling complex data distributions. Their adversarial training framework encourages the generator to produce outputs indistinguishable from real data. However, GANs are notorious for training instability, mode collapse, and sensitivity to hyperparameters. Ensuring diversity in generated motions and balancing the generator-discriminator dynamic remain significant challenges.

Transformer-Based Models have revolutionized motion synthesis by effectively modeling long-range dependencies and handling variable-length sequences (Aksan et al., 2020; Petrovich et al., 2021). The attention mechanism allows Transformers to capture com-

plex spatial and temporal relationships in motion data. Their parallelizable architecture enables efficient training on large datasets, and they excel in tasks involving multimodal inputs, such as text-to-motion or music-to-dance generation. However, Transformers require substantial computational resources and large amounts of data for effective training. Overfitting can be a concern, and their performance may degrade with limited or noisy data.

Diffusion Models represent a significant advancement by directly modeling data distributions through iterative denoising processes (Zhang and Wang, 2022). They are capable of generating high-quality, diverse motions and can incorporate various conditioning information, such as textual descriptions or physics constraints. Diffusion models offer advantages in capturing the stochastic nature of human motion without relying on adversarial training. Their primary limitation lies in computational intensity; the iterative nature of the denoising process results in slower generation times, which is impractical for real-time applications. They also require large datasets and significant computational resources for training.

Multimodal Learning and Cross-Modal Representations focus on integrating information from multiple modalities to generate contextually appropriate motions (Tevet et al., 2022a; Kim et al., 2022). By leveraging shared embedding spaces and contrastive learning, these approaches enable models to understand and synthesize motions conditioned on text, audio, or visual cues. They enhance the versatility and applicability of motion synthesis models in tasks like text-driven animation or music-conditioned dance generation. Challenges in this domain include the need for large, diverse datasets and the complexity of aligning and translating between different modalities.

Other Models, such as those utilizing graph neural networks (Li et al., 2022) or phase-functioned neural networks (Holden et al., 2017), address specific aspects of motion synthesis like structural representations and real-time character control. These models contribute valuable insights but may have limited applicability compared to more general frameworks.

Comparison and Insights: RNNs and VAEs are more data-efficient and computationally less demanding than Transformers and diffusion models, which excel with large datasets but struggle to generalize with limited data. Transformers and diffusion models also require significant computational resources, especially GANs and diffusion models. In generation quality, GANs and diffusion models lead with high-quality and diverse outputs, while VAEs offer diversity but lack sharpness, and Transformers generate realistic, multimodal-compatible motions. For long-term dependencies, Transformers and diffusion models are superior, whereas RNNs may lose coherence. Multimodal tasks benefit from Transformers, as RNNs and VAEs are less versatile.

Gaps in the Literature: Key challenges persist, including data scarcity, particularly for diverse motion styles, and maintaining physical plausibility, though PhysDiff (Yuan et al., 2023) addresses this partially. High computational demands, especially with Transformers and diffusion models, hinder real-time applications, and models often

struggle to generalize to new motions or domains without substantial retraining.

Evaluation Metrics

Evaluating human motion synthesis models requires metrics that can accurately assess the quality, realism, and diversity of generated motions. The choice of evaluation metrics significantly influences model development and comparison. This section discusses common evaluation metrics used in the field, highlighting their advantages and limitations, as shown in Table 2.2.

Table 2.2: Common evaluation metrics for human motion synthesis models, highlighting their advantages and limitations.

Metric	Advantages	Limitations
Mean Per Joint Position Error (MPJPE) (Ionescu et al., 2014)	Simple to compute; provides a direct measure of accuracy in joint positions.	Sensitive to misalignments; does not account for perceptual realism or temporal coherence.
Frechet Inception Distance (FID) (Dowson and Landau, 1982)	Measures both quality and diversity of generated motions; sensitive to higher-order statistics.	Requires a pretrained feature extractor; may not correlate perfectly with human perception of motion quality.
Diversity Metrics (e.g., Average Pairwise Distance (APD)) (Guo and Joo, 2020)	Important for evaluating models producing diverse outputs; highlights mode collapse issues in generative models.	High diversity does not guarantee realism; may encourage unrealistic motions to maximize diversity.
Physical Plausibility Metrics (Holden et al., 2016)	Directly evaluate physical correctness; crucial for applications requiring high realism.	Difficult to define and compute for complex motions; do not assess perceptual quality.
User Studies and Perceptual Metrics (Wang et al., 2021c)	Provide insights into human perception; capture aspects not reflected in quantitative metrics.	Subjective and time-consuming; results may vary depending on participant demographics.
Task-Specific Metrics (e.g., Top-k Accuracy) (Petrovich et al., 2021)	Evaluate the relevance and correctness of generated motions with respect to the input.	Do not assess motion quality; may be misleading if the model generates generic motions that fit multiple classes.

Considerations in Metric Selection

- *Alignment with Objectives*: Metrics should align with the intended application and objectives of the model. For instance, if diversity is crucial, diversity metrics should be emphasized.

- *Combination of Metrics*: Relying on a single metric may not provide a comprehensive evaluation. Combining multiple metrics can offer a balanced assessment.

- *Correlation with Human Perception*: Metrics should ideally correlate with human judgments of motion quality. Metrics that fail to capture perceptual aspects may not reflect true performance.

- *Reproducibility*: Metrics should be well-defined and reproducible to allow for fair comparisons across models.

2.4 Challenges and Future Directions

Human motion synthesis faces several key challenges that continue to limit the applicability and performance of current models. One prominent issue is semantic ambiguity in text-based models. Text-conditioned motion generation relies heavily on the quality and specificity of textual input. However, natural language is inherently ambiguous, leading to difficulties in generating precise and contextually appropriate motions. Models like TEMOS and MotionCLIP attempt to map textual descriptions to motion, but they often struggle to disambiguate phrases that describe abstract or multiple possible motions, resulting in outputs that may not fully capture the intended action. Future research could focus on integrating more sophisticated natural language understanding mechanisms or multi-modal cues to mitigate this ambiguity.

Another challenge is temporal coherence in long sequences. Motion models often excel in generating short, coherent motion sequences but encounter difficulties when required to produce longer, temporally consistent motions. Models based on recurrent neural networks (RNNs), Transformers, or VAEs typically have trouble maintaining consistency over long time frames due to accumulated prediction errors or the loss of global structure. Ensuring that long sequences maintain natural transitions and adhere to physical constraints remains an open problem. Advanced sequence modeling techniques and the integration of memory-based architectures could help improve long-term temporal coherence.

Generalization to unseen motions is another significant hurdle. Most current models are trained on specific datasets, and their ability to generalize to new, unseen motion types or styles is often limited. While approaches like VAEs and GANs introduce some level of variability, these models still require large, diverse datasets to perform well on unfamiliar actions. One potential direction for future research is leveraging transfer learning, domain adaptation, or self-supervised learning techniques to enhance the generalization

capabilities of motion synthesis models, reducing their dependency on large annotated datasets.

Finally, computational efficiency remains a critical concern, especially for real-time applications like virtual reality or interactive gaming. High-quality motion generation models, such as diffusion models or Transformer-based architectures, are often computationally expensive, limiting their use in real-time systems. Optimizing these models for efficiency, possibly through model pruning, quantization, or the development of lightweight architectures, will be essential for broader adoption of human motion synthesis in latency-sensitive applications.

2.5 Summary

In summary, the field of human motion synthesis has made significant advancements, with key methodologies such as Transformers, VAEs, GANs, and diffusion models contributing to the generation of realistic and diverse human motions. From early probabilistic models to deep learning-based approaches, these techniques have addressed many challenges in temporal dynamics, diversity, and conditioning on external inputs such as text or actions. However, challenges remain, particularly in handling semantic ambiguity in text-based models, ensuring temporal coherence in long motion sequences, improving generalization to unseen motion styles, and enhancing computational efficiency for real-time applications. Addressing these challenges will be crucial for future advancements, enabling more robust, flexible, and efficient motion synthesis systems.

Chapter 3

Methodology

This chapter outlines the methodology employed to develop a transformer-based autoencoder for human motion synthesis. The central objective of this research is to create a model capable of generating realistic human motion sequences in response to intuitive control inputs, such as joystick movements or touch gestures, without relying on predefined action categories or textual descriptions. To achieve this, we first construct a convolutional autoencoder to learn a valid motion manifold from unlabelled motion capture data, capturing the underlying structure of human movements. Recognizing the limitations inherent in convolutional architectures, we subsequently extend our approach by incorporating a transformer-based autoencoder. This model leverages self-attention mechanisms to enhance the modeling of complex temporal dynamics and long-range dependencies in human motion data.

Section 3.1 introduces the challenges associated with existing human motion synthesis methods and defines the problem our research aims to address. And provides justification for the proposed approachSection 3.2 discusses the development of the motion manifold using an autoencoder, including the convolutional autoencoder's architecture, training, and performance in Subsection 3.2.1, followed by the transformer autoencoder's architecture, training, and evaluation in Subsection 3.2.2. Section 3.3 describes the mapping of interactive control inputs to human motion, covering control representation, footstep contact integration, mapping control inputs to motion, and implementation details. The chapter concludes in Section 3.4 with a summary of key findings and how the proposed methodology addresses the research objectives.

3.1 Problem Definition

Human motion synthesis faces persistent challenges, particularly in adapting to dynamic, interactive environments. Traditional methods often rely on predefined action

categories or textual descriptions, limiting their flexibility and responsiveness to real-time user inputs. These approaches require extensive manual annotation and struggle to generalize to novel or unforeseen movements, hindering their applicability in interactive systems such as gaming or virtual reality. Therefore, generating realistic human motion sequences from intuitive controls without predefined categories remains an open and critical problem.

Our research addresses this issue by developing a model that translates intuitive control inputs, such as joystick or gesture-based commands, into natural human motion sequences. To do this, we utilize the comprehensive AMASS dataset, which aggregates diverse motion capture data, allowing the model to learn complex motion patterns and generalize effectively. By leveraging the SMPL (Skinned Multi-Person Linear) model for consistent, anatomically accurate data representation, we ensure high fidelity in the generated motions.

Justification for the Proposed Approach

The proposed methodology is justified by the need to overcome several limitations inherent in current human motion synthesis approaches. First, convolutional autoencoders, while effective in learning motion manifolds, struggle to capture long-range temporal dependencies in motion data, leading to less coherent sequences over time. Recognizing this, we initially build a convolutional autoencoder to establish a motion manifold but then extend this with a transformer-based architecture. The transformer's self-attention mechanism allows it to effectively model both local and global dependencies in motion sequences, enabling smoother and more natural motion synthesis.

We choose the transformer architecture due to its strength in handling complex temporal dynamics, which is crucial for human motion data. Unlike convolutional architectures, transformers can attend to any frame in a sequence, capturing long-range dependencies essential for smooth transitions between movements. By using this architecture, our model can generate more accurate and fluid motion sequences, as evaluated through quantitative metrics like Mean Squared Displacement (MSD) and Mean Per Joint Position Error (MPJPE), which assess the coherence and realism of motion.

Moreover, this approach enables the model to translate intuitive control inputs into motion in real time, broadening its applicability in interactive environments like virtual reality, gaming, and simulation. The combination of the AMASS dataset and transformer-based modeling ensures the model's scalability and adaptability to various motion patterns and input types, providing a robust foundation for future extensions such as terrain-aware motion generation or diffusion-based models.

3.2 Learning Motion Manifold via Autoencoder

In the field go human motion synthesis, it is easy to find a configuration in this space which does not represent valid human motion - either by setting the joint angles to

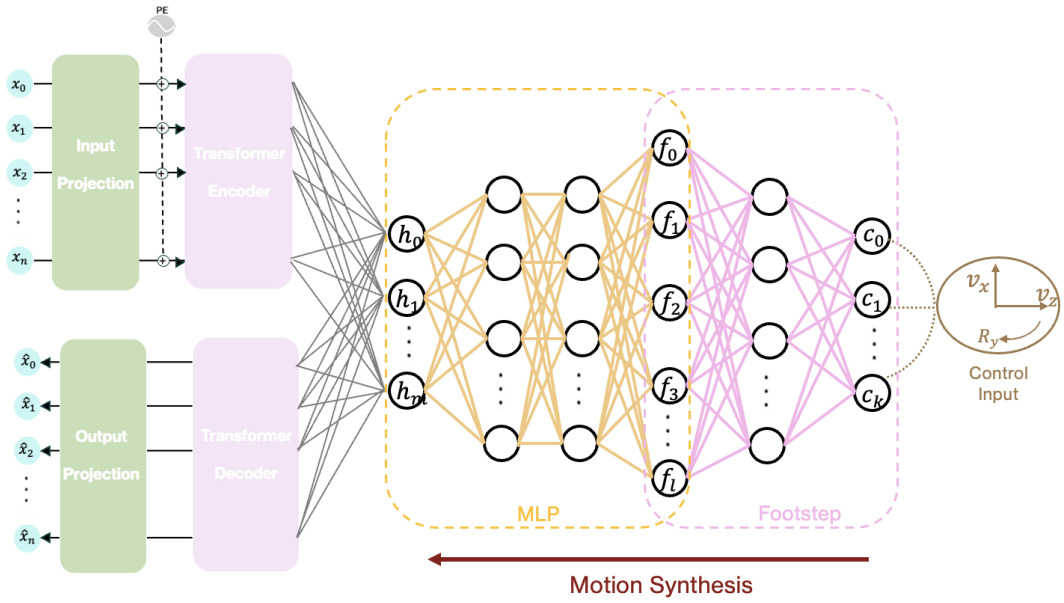


Figure 3.1: Overall model architecture for generating seamless human motion from interactive control inputs. **Left:** The Transformer Autoencoder, where the motion manifold is trained in the latent space. **Right:** The mapping of interactive control inputs to the latent space.

extreme configurations and making poses that are biologically impossible, corrupted motions, or by creating large differences between frames of the time-series such that the motion is too fast for humans to achieve. Therefore, it is of great interest to researchers to find good techniques for finding this subspace of valid motion, called the motion manifold. This is because it allows for data processing tasks to be defined with respect to it - such as interpolating motion only through the space of valid motions - or taking distances which do not pass through the space of impossible motion.

3.2.1 Convolutional Autoencoder

We first present a technique for learning a manifold of human motion data using Convolutional Autoencoders, inspired by Holden et al. (2016). Motion data is typically represented as a time-series where each frame represents some pose of a character. Poses of a character are usually parametrised by the character joint angles, or joint positions. This representation is excellent for data processing, but valid human motion only exists in a small subspace of this representation.

Data Acquisition

Preprocessed Dataset

The dataset used for training the autoencoder consists of retargeted motion capture data, sourced from the CMU Motion Capture Database (Carnegie Mellon University, 2003), HDM05 (Müller and Röder, 2007), MHAD (Ofli et al., 2013), and additional motion data captured locally by the University of Edinburgh (Holden et al., 2016). These datasets were retargeted to a consistent skeletal structure, ensuring uniformity in scale and bone lengths across all data. The retargeting process involved first copying the joint angles from the source skeleton to the target skeleton, followed by scaling the source skeleton to match the dimensions of the target. Finally, a full-body inverse kinematics scheme (Yamane and Nakamura, 2003b) was applied to adjust the target skeleton's joint positions to correspond accurately with those of the source skeleton. As a result, the final dataset is approximately twice the size of the CMU Motion Capture Database, containing around six million frames of high-quality motion data, sampled at 120 frames per second.

Data Format for Training

For training purposes, the motion data are subsampled to 60 frames per second and converted from the original joint angle representation to a 3D joint position format. These joint positions are defined in the local coordinate system of the body, with the origin projected onto the ground plane at the root position. The forward direction of the body (Z-axis) is calculated by averaging the vectors across the left and right shoulders and hips and computing the cross product with the vertical axis (Y-axis).

In addition to joint positions, the global velocity in the XZ-plane and the rotational velocity around the vertical axis (Y-axis) are appended to each frame. These velocities can be integrated over time to reconstruct the global translation and rotation of the character. Foot contact labels, indicating whether the toe or heel of the character is in contact with the ground, are also appended to the input representation (Lee et al., 2002). To normalize the data, the mean pose is subtracted from each frame, and the joint positions are divided by their respective standard deviations. Velocities and foot contact labels are similarly normalized.

This model is designed to handle motion sequences of varying lengths; however, to improve computational efficiency during training, the motion data are divided into overlapping windows of n frames, with a 50% overlap. his overlapped sampling setup ensures temporal coherence, enabling the model to effectively learn robust temporal dynamics and inter-joint correspondences. During training, we set $n = 240$, resulting in input vectors of the form $X \in \mathbb{R}^{n \times d}$, where n is the window size and $d = 73$ represents the number of degrees of freedom. The first three elements of each vector correspond to the root position, projected as $(0, 0, 0)$. The next 63 elements capture the 3D coordinates (x, y, z) of 21 body joints. Following this, three elements are used to encode the character's global trajectory, comprising the global velocity in the XZ-plane and the rotational ve-

locity around the Y-axis. The final four elements represent foot contact labels, indicating the contact state of the left heel, left toe, right heel, and right toe.

Network Structure

The autoencoder designed to construct a valid motion manifold comprises a one-layer convolutional neural network (CNN) encoder and a one-layer CNN decoder. It performs one-dimensional convolutions over the temporal domain for each filter independently, hence referred to as a convolutional 1D network. The network facilitates a forward operation Φ (encoding) and a backward operation $\Phi^\dagger$ (decoding). The forward operation intakes the input vector $\mathbf{X}$ in the visible unit space and outputs encoded values $\mathbf{H}$ in the hidden unit space. The output from the convolution process, termed a feature map, signifies the presence of a specific filter at distinct time points.

The forward operation:

$$\Phi(\mathbf{X}) = \mathrm{ReLU}\left(\Psi(\mathbf{X} * \mathbf{W}_0 + \mathbf{b}_0)\right) \tag{3.1}$$

consists of a convolution (denoted $*$) using weights matrix $\mathbf{W}_0 \in \mathbb{R}^{d \times v \times u_0}$, addition of a bias $\mathbf{b}_0 \in \mathbb{R}^m$, a max pooling operation Ψ, and the nonlinear operation $\mathrm{ReLU}(x) = \max(x, 0)$, where u_0 is the temporal filter width and m is the number of hidden units in the autoencoding layer, each set to 25 and 256 in this work; a temporal filter width of 25 ensures each filter roughly corresponds to half a second of motion, while 256 hidden units is experimentally found to produce good reconstructive resolution.

The max pooling operation Ψ returns the maximum value of each pair of consecutive hidden units on the temporal axis. This reduces the temporal resolution, ensures that the learned bases focus on representative features, and also allows the bases to express a degree of temporal invariance. Using of ReLu because ....

The backward operation:

$$\Phi^\dagger(\mathbf{H}) = (\Psi^\dagger(\mathbf{H}) - \mathbf{b}_0) * \widetilde{\mathbf{W}}_0 \tag{3.2}$$

takes hidden units $\mathbf{H} \in \mathbb{R}^{2 \times m}$ as input, and consists of an inverse-pooling operation $\Psi^\dagger$, a subtraction of a bias $\mathbf{b}_0$, and convolution using the weights matrix $\widetilde{\mathbf{W}}_0$. $\widetilde{\mathbf{W}}_0 \in \mathbb{R}^{d \times m \times u_0}$ is simply the weights matrix $\mathbf{W}_0$, reflected on the temporal axis, and transposed on the first two axes used to invert the convolution operation.

Justification of design choice:

Typically, the max-pooling operation in convolutional neural networks is configured to return the indices of the maximum values, facilitating precise reconstruction during the inverse pooling process. However, in the context of our framework, employing these indices for inverse pooling is strategically avoided. This decision supports direct manipulation of motion data within the hidden space, enhancing the functionality of the motion manifold for subsequent computational tasks.

In conventional settings, retaining the exact indices is necessary for accurately mapping back to the original inputs during the decoding phase. If these indices were not stored, each forward pass would necessitate re-encoding to retrieve them, which would be computationally expensive and impractical for scalable systems. By eliminating the dependency on exact indices, our model allows for more flexible and efficient handling of motion data.

For the purposes of motion editing and future integration—such as incorporating trajectory inputs as conditions or enabling cross-model data integration—we design the system to allow direct interactions with the hidden representations. Consequently, the decoder is capable of outputting motion data without requiring a backtrack to retrieve specific indices, which streamlines the process and preserves the integrity of the decoded motions.

During the inverse pooling operation, rather than relying on stored indices to place the value precisely, our model randomly assigns the pooled value to one of the potential positions within the output tensor, setting the alternative position to zero. This approach not only simplifies the architecture but also aligns with our use of a max pooling kernel of size 2 with a stride of 2, ensuring efficient data processing while maintaining the quality of the motion representation in the hidden space.

Training

The training of our network is structured to enable the autoencoder to accurately reproduce a given input X, adhering to both the forward and backward operational dynamics described earlier. The objective of the training is formalized through the minimization of a cost function with respect to the network parameters $\theta = \{W_0, b_0\}$:

$$\text{Cost}(X, \theta) = \|X - \Phi^{\dagger}(\Phi(X))\|^2 + \alpha\|\theta\|_1 \tag{3.3}$$

In this equation, $\|X - \Phi^{\dagger}(\Phi(X))\|^2$ quantifies the squared reproduction error, which measures the fidelity of the input reconstruction. The term $\alpha\|\theta\|_1$ introduces a sparsity constraint, ensuring that only the essential network parameters are active, thereby promoting a compact and efficient model representation. The sparsity coefficient α is empirically set to 0.0006, optimized to balance between reconstruction accuracy and model simplicity. The comprehensive overview of hyperparameter settings are summarized in Table 4.3.

Training is implemented using the PyTorch framework, utilizing the Adam optimizer with a learning rate of 1×10^{-5}. The model undergoes 15 training epochs, with each batch comprising 16 samples, and incorporates a dropout rate of 0.2 to mitigate overfitting. The entire training process is completed in approximately 1 minute and 30 seconds on a NVIDIA A100 GPU, demonstrating the efficiency of our computational setup.

Upon completion of the training phase, the network's filters reveal strong temporal and

inter-joint correspondences. A visualization of the learned weights is presented in Figure 3, where it is demonstrated that each filter captures the movement of several joints over a period of time.

Hyperparameter Optimization

The training of our autoencoder required meticulous fine-tuning of hyperparameters to achieve optimal performance. Initially, hyperparameter optimization was conducted with a fixed sparsity coefficient, $\alpha = 0.1$. This initial phase helped establish a baseline for other parameters such as batch size, learning rate, and dropout rate (see Table 4.1).

Table 3.1: Initial Hyperparameter Tuning with $\alpha = 0.1$

Trial	**Dropout**	**Batch Size**	**Learning Rate (Lr)**	**Training Loss**	**Test Loss**
1	None	32	1e-2	100.000	0.902
2	None	32	1e-3	11.000	0.873
3	None	32	1e-4	1.600	0.897
4	None	32	1e-5	0.750	1.110
5	0.2	32	1e-5	1.056	0.968
6	0.2	32	1e-4	2.053	0.944
7	0.2	32	1e-3	11.246	0.947
8	0.2	32	1e-2	99.083	0.960
9	0.2	32	1e-6	389.160	0.932
10	0.2	16	1e-4	1.998	1.104
11	0.2	64	1e-4	2.038	1.041
12	0.2	16	1e-5	1.068	0.925
13	0.2	64	1e-5	1.030	1.072

After the initial experiments, it was observed that while the network could reproduce motions, the output was excessively static. To address this issue, we conducted further experiments by reducing the sparsity coefficient, α, to zero. This modification led to more dynamic reconstructions that more closely mirrored the ground truth, though it also introduced significant shakiness in the outputs. Further refinement led to the exploration of various α settings to balance motion fidelity and stability (see Table 4.2). The optimal value of α was identified as 0.0006, which effectively balanced the sparsity constraint with the model's ability to capture and reproduce complex movements.

These experiments underscore the critical impact of α on the model's performance, highlighting its role in controlling the trade-off between accuracy and the generalization capability of the autoencoder. Detailed visual assessments and quantitative metrics guided the final selection of hyperparameters, ensuring robust model training and effective motion reproduction.

Table 3.2: Refined Hyperparameter Tuning for Optimal α

Trial	Alpha	Training Loss	Test Loss	Visual
1	0	meaningless	meaningless	Very shaky movements
2	0.1	1.068	0.925	Shaky movements
3	0.01	0.817	0.701	Shaky movements
4	0.001	0.608	0.530	Shaky movements
5	0.0001	0.539	0.389	Shaky movements
6	0.0004	0.507	0.310	Slightly Shaky movements
7	0.0005	0.481	0.230	Smooth and aligned
8	0.0006	0.467	0.251	Smoothest
9	0.0007	0.408	0.241	Smooth and aligned

Summary

This section detailed the construction and validation of a convolutional autoencoder specifically designed for learning a valid motion manifold. Through rigorous hyperparameter optimization, the model demonstrated robustness in handling complex dynamic sequences, effectively capturing and reproducing the motion manifold with high fidelity. The trained model achieves a delicate balance between accuracy and stability in motion reconstruction. Learning a motion manifold has many practical applications in human motion synthesis. A key advantage of this approach is its explicit consideration of the temporal domain of the manifold, linking not only the position of individual joints within the same pose but also across different frames in the motion sequence.

3.2.2 Transformer Autoencoder

Transformer models, initially developed for natural language processing (Vaswani et al., 2017b), have demonstrated remarkable versatility in sequence-based tasks. This has led to their adoption in various fields, including computer vision and human motion synthesis, where temporal dependencies and long-range interactions between features are crucial. In this section, we extend the traditional autoencoder framework to incorporate transformers, leveraging their attention mechanisms to enhance the modeling of complex temporal dynamics in human motion data.

Rationale for Using Transformer Autoencoders

Traditional convolutional or recurrent autoencoders rely on local context or sequential memory to capture relationships in the data. However, these models may struggle with capturing long-term dependencies or interactions between distant elements in a sequence. Human motion data, consisting of joint positions over time, often contains such long-range dependencies that are critical for capturing the fluidity and naturalness of motion.

The transformer architecture addresses this limitation by using a self-attention mechanism, which allows the model to focus on different parts of the sequence independently of

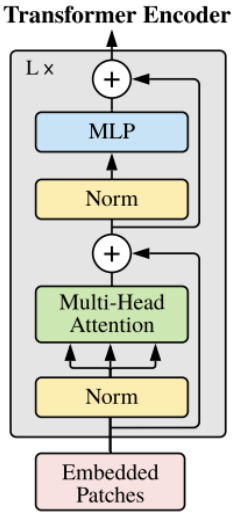


Figure 3.2: Transformer Encoder. Figure from (Dosovitskiy et al., 2021b)

their position. This characteristic makes transformers particularly effective at capturing both local and global dependencies in motion data, potentially leading to more accurate and realistic motion reconstruction compared to convolutional-based autoencoders.

Architecture of the Transformer Autoencoder

The proposed transformer autoencoder consists of two main components: a transformer encoder and a transformer decoder. The encoder processes the input sequence (i.e., a motion clip) by applying self-attention across all time steps, allowing the model to learn interdependencies between joints at different points in time. A fixed number of layers and multi-head attention mechanisms are employed to balance computational efficiency and model expressiveness. A visualization of Transformer Encoder is in Figure 3.2.

Similarly, the decoder reconstructs the original sequence by applying attention mechanisms to the latent representation produced by the encoder. This architecture enables the model to reconstruct complex motion sequences with high fidelity while preserving the temporal structure of the data.

The overall architecture can be summarized as follows:

- **Input projection layer**: The input, a sequence of joint positions, is first projected into a higher-dimensional embedding space.
- **Positional encoding**: Since transformers do not inherently encode the order of the input sequence, a learnable positional encoding is added to the input embedding to provide the model with information about the temporal order of frames.

- **Transformer Encoder**: The encoded input is passed through a stack of transformer encoder layers, each of which applies multi-head self-attention and feedforward layers. These layers capture both local and long-range dependencies in the motion sequence.
- **Transformer Decoder**: The decoder reconstructs the motion sequence by applying self-attention mechanisms to the latent representation. The output of the decoder is then projected back to the original input dimensions.
- **Output projection layer**: Finally, a linear projection maps the decoder output back to the original motion representation (e.g., 3D joint coordinates).

Positional Encoding in Transformer Autoencoders

Transformer-based models, unlike convolutional and recurrent neural networks, lack inherent knowledge of the sequential ordering of input data. To address this, positional encodings are added to the input embeddings, providing information about the relative or absolute position of tokens in a sequence. Two primary methods of positional encoding are widely used: sinusoidal encoding and learnable encoding.

Sinusoidal positional encodings are based on fixed sine and cosine functions, where the positions are mapped to periodic functions of different frequencies. This method provides a non-learnable yet smooth encoding scheme that inherently captures sequential relationships. As shown in Figure 3.3, the positional vectors exhibit a clear periodic pattern that spreads information across various dimensions. While this method is computationally efficient and theoretically grounded, it may limit the model's ability to fully adapt to the task-specific patterns inherent in more complex datasets like human motion sequences.

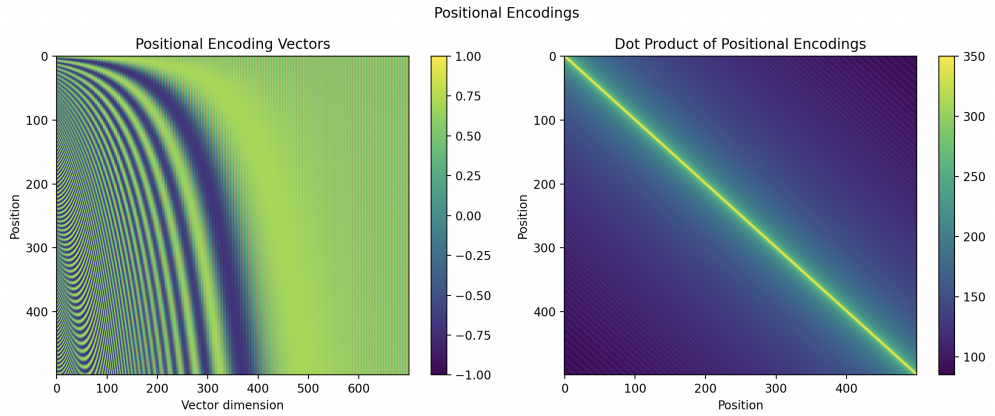


Figure 3.3: Visualization of Sinusoidal Positional Encoding Vectors and their Dot Product.

Learnable positional encodings, on the other hand, allow the model to optimize the en-

coding scheme during training. This flexibility enables the model to discover encoding patterns that better capture the intricacies of the data. The enhanced learned positional encoding vectors, depicted in Figure 3.4, show higher variability across dimensions, suggesting a more dynamic adaptation to the input data. The learned encoding also exhibits less structure, indicating that the model is learning specific positional features for the task at hand.

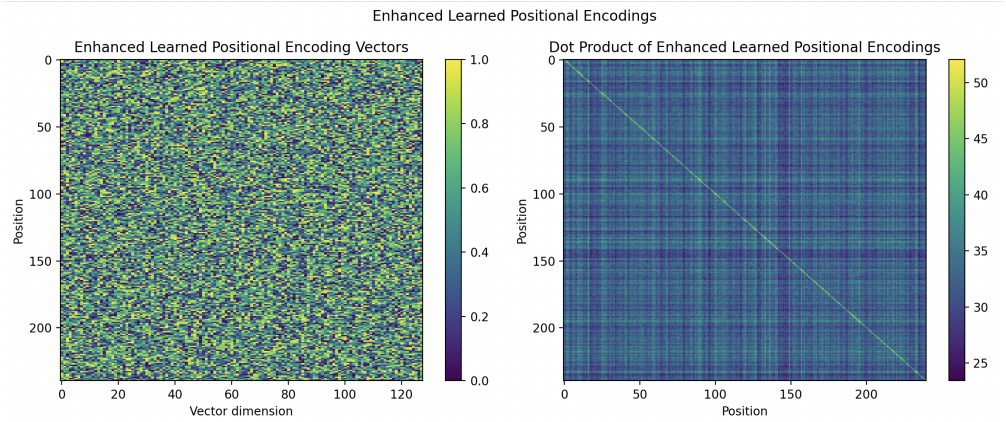


Figure 3.4: Visualization of Enhanced Learned Positional Encoding Vectors and their Dot Product.

The performance comparison between sinusoidal and learnable encodings, as highlighted in Table 3.3, demonstrates a significant difference in the average test error, with learnable encodings outperforming sinusoidal encodings. Despite slightly higher training loss (0.7975 for learnable encoding vs. 0.7579 for sinusoidal encoding), the learnable encoding achieves a lower test error (0.1202 compared to 0.3131 for sinusoidal encoding). This suggests that learnable encodings generalize better to unseen data, capturing motion patterns in a more expressive manner.

Table 3.3: Comparison of Sinusoidal and Learnable Positional Encoding Methods

Positional Encoding	**Training Loss (Avg)**	**Test Error (Avg)**
Sinusoidal	0.757940	0.313129
Learnable	0.797529	0.120244

Implications for Human Motion Synthesis

Given the lower test error and the adaptability of the learnable encoding, it appears to be a more suitable choice for tasks involving complex temporal dynamics, such as human motion synthesis. By enabling the model to learn the most effective positional representations, the learnable encoding strikes a balance between theoretical simplicity and

practical accuracy. This choice ultimately allows the transformer autoencoder to generate more realistic and expressive motion patterns, crucial for applications in character animation and virtual environments.

Summary

The transformer autoencoder represents a significant advancement in the modeling of human motion data. By leveraging the self-attention mechanism, the transformer is capable of capturing both local and global dependencies in motion sequences, leading to more accurate reconstructions. This architecture holds potential not only for motion synthesis tasks but also for broader applications where sequence modeling is required. Further work will explore the integration of transformer autoencoders with other generative models to enhance motion diversity and expressiveness.

3.3 Mapping Interactive Control Inputs to Seamless Human Motions

This chapter elaborates on the methodologies employed to map hinteractive control inputs into seamless human motion sequences. Building upon the learned motion manifold from the autoencoder, we aim to generate realistic human motion sequences in response to control inputs, such as trajectories or target positions. To achieve this, we first process the control inputs through a network that incorporates footstep contact information, ensuring realistic locomotion. Subsequently, we map the augmented control inputs into the latent space of the learned motion manifold using a multilayer perceptron (MLP), drawing inspiration from the work of Holden et al. (2016). This approach enhances the precision and applicability of motion synthesis, ensuring that generated motions are both natural and responsive to the control inputs.

3.3.1 Control Input Representation

The control input C encapsulates high-level parameters that guide motion synthesis. In the context of locomotion, C may include trajectory points, orientation angles, and speed profiles, which represent the path, facing direction, and target speed at various points. In interactive control, such as in mobile gaming (e.g., League of Legends), the control inputs come from a joystick or a touch control circle, where the player manipulates the speed, direction, and rotation. The velocity along the x-axis, z-axis, and the rotation around the y-axis correspond to how far the joystick is pushed or how far the finger moves from the center of the touch circle. This intuitive and interactive approach allows the player to control the character's movement on the x-z plane. During the training process, these inputs were appended to the end of the motion data to enable the network to learn the mapping between control and realistic motion synthesis effectively.

3.3.2 Incorporating Footstep Contact for Realistic Locomotion

Without explicit foot contact constraints, motion generated by simply following a control trajectory often lacks realistic foot contact behavior, leading to issues such as floating or sliding instead of natural walking, running, or jumping. To address this, adding foot contact information to the control input is essential. The purpose of incorporating foot contact parameters is to ensure physically plausible movement, where foot placement adheres to human-like stepping patterns and proper ground interaction.

To link the control input c with foot contact parameters T, we employ a two-layer convolutional neural network that maps the control trajectory to learned foot contact parameters. The network operates as follows:

$$T(c) = \text{ReLU}(c * W_4 + b_4) * W_5 + b_5 \tag{3.4}$$

Here, W_4, W_5, b_4, and b_5 represent the learned weights and biases of the network. This architecture generates foot contact information based on the interactive control inputs, such as forward velocity (V_x), lateral velocity (V_z), and yaw rotation (R_y). Once trained, the network computes the frequency and step duration of the square wave function used to mimic human footstepping, ensuring realistic foot-ground interactions during motion.

By incorporating foot contact information, we prevent unrealistic motion artifacts, like floating or sliding, and ensure that the generated motion adheres to the physical constraints of stepping and foot placement, thereby significantly enhancing its realism. For further details on how foot contact timings are computed and integrated, refer to Section 3.3.4.

3.3.3 Mapping to Control Inputs to Motion

After augmenting the control inputs with footstep contact information, we map the combined vector into the latent space of the learned motion manifold using a multilayer perceptron (MLP). This network transforms the control and footstep information into a representation suitable for generating motion sequences. A illustration of this end-to-end mapping is in Figure 3.5.

The MLP comprises several fully connected layers with nonlinear activation functions:

- **Input Layer**: Receives the augmented control input vector $[\mathbf{C}, \mathbf{T}]$.
- **Hidden Layers**: Multiple layers with specified numbers of neurons, each followed by an activation function (e.g., ReLU).
- **Output Layer**: Produces the latent representation $\mathbf{H}$ corresponding to the hidden space of the autoencoder.

Mathematically, the mapping is expressed as:

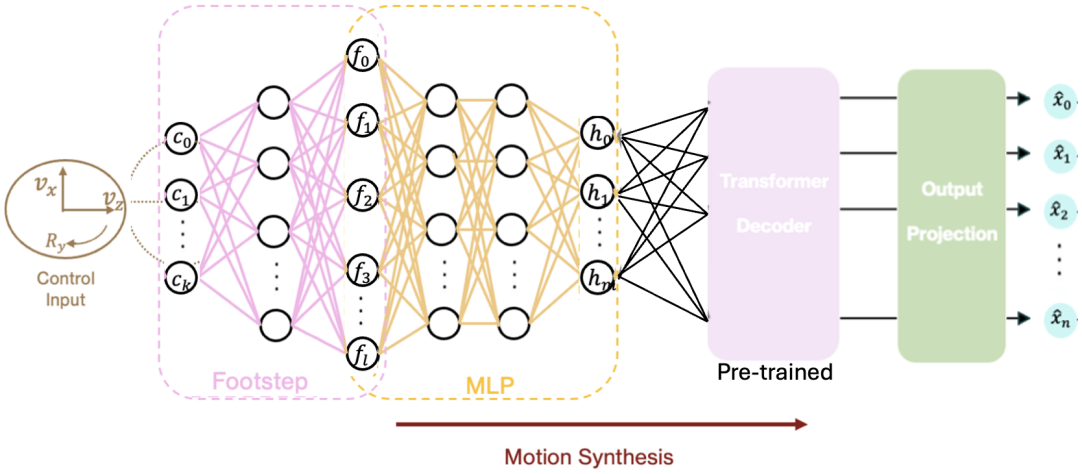


Figure 3.5: Revised network architecture for illustrating the end-to-end motion synthesis with pre-trained Tranformer Decoder.

$$\mathbf{H} = f_{\text{MLP}}([\mathbf{C}, \mathbf{T}]; \Theta_{\text{MLP}}) \tag{3.5}$$

where f_{MLP} represents the function of the MLP parameterized by Θ_{MLP}, the set of weights and biases across all layers. This network learns to translate the high-level commands and foot contact information into a latent representation that the decoder can use to generate precise human motions.

Training Dataset Preparation

For training the mapping network, we prepared a dataset consisting of pairs of control inputs and corresponding motion sequences. Each sample includes:

- The control input $\mathbf{C}$, representing the high-level commands.
- The foot contact information $\mathbf{T}$, derived from the motion data.
- The corresponding motion sequence $\mathbf{X}$, represented in the same format used for training the autoencoder.

The motion sequences provide ground truth data for learning the mapping from control inputs to motion. The foot contact information is essential for capturing the timing and placement of steps, which are critical for realistic locomotion.

Training Procedure

The training aims to learn the parameters of both the footstep incorporation network and the MLP mapping network jointly. However, the autoencoder, which was previously trained to learn the motion manifold, is kept frozen during this process. Only its decoder is used to reconstruct motion sequences from the latent representations generated by the MLP.

The steps are as follows:

1. **Compute Footstep Contact Information**: For each control input $\mathbf{C}$, use the footstep incorporation network to predict $\mathbf{T}$.
2. **Map to Latent Space**: Use the MLP to map the augmented control input $[\mathbf{C}, \mathbf{T}]$ to the latent representation $\mathbf{H}$:

$$\mathbf{H} = f_{\text{MLP}}([\mathbf{C}, \mathbf{T}]) \tag{3.6}$$

3. **Reconstruct Motion**: Pass $\mathbf{H}$ through the **decoder** of the autoencoder to generate the reconstructed motion sequence $\hat{\mathbf{X}}$:

$$\hat{\mathbf{X}} = \Phi^{\dagger}(\mathbf{H}) \tag{3.7}$$

4. **Compute Loss**: Calculate the loss between the reconstructed motion $\hat{\mathbf{X}}$ and the ground truth motion $\mathbf{X}$, including reconstruction error and any regularization terms:

$$L = \|\mathbf{X} - \hat{\mathbf{X}}\|^2 + \lambda L_{\text{reg}} \tag{3.8}$$

   where L_{reg} represents regularization terms, and λ is a weighting coefficient.

5. **Optimize Parameters**: Use backpropagation and an optimizer (e.g., Adam) to update the weights and biases of the footstep incorporation network and the MLP, minimizing the loss L. The parameters of the autoencoder's decoder are kept fixed during this process.

By training the networks together while keeping the autoencoder's decoder frozen, we ensure that the mapping from control inputs to motion sequences accurately reflects both the high-level commands and the necessary foot contact dynamics, all within the established motion manifold.

Integration with the Autoencoder

In our overall network architecture, only the **decoder** of the trained autoencoder is utilized to reconstruct human motion from the latent representations generated by the

MLP. The autoencoder was trained separately to learn the motion manifold, and its parameters are kept fixed during the training of the mapping network. This approach offers several advantages:

- **Modularity**: The autoencoder serves as a foundational component that captures the essential characteristics of human motion. By keeping it fixed, we can train different mapping networks for various control schemes without the need to retrain the autoencoder.
- **Stability**: Freezing the autoencoder's parameters ensures that the learned motion manifold remains consistent, providing a reliable latent space for the MLP to map into.
- **Efficiency**: Training only the mapping network and footstep incorporation network reduces computational overhead and training time.

The overall pipeline consists of four stages. First, control inputs $\mathbf{C}$ (e.g., velocities V_x, V_z, and rotation R_y) are processed. Second, the footstep incorporation network computes foot contact information $\mathbf{T}$. Third, an MLP maps $[\mathbf{C}, \mathbf{T}]$ into the latent space $\mathbf{H}$. Finally, the decoder $\Phi^\dagger$ reconstructs the human motion $\hat{\mathbf{X}}$. This pipeline operates end-to-end from the control inputs to the generated human motion, even though the autoencoder and footstep incorporation network were trained separately. The end-to-end nature of the pipeline refers to the seamless flow of data through the network during inference, resulting in responsive and natural motion generation.

3.3.4 Implementation Details

Foot Contact Computation for Realistic Locomotion

Accurate modeling of foot-ground interactions is crucial for generating realistic synthesized locomotion. This section details the mathematical computations used to determine the frequency, step duration, and square wave representations for modeling foot contact states.

Modeling Foot Contact Using Square Waves

Square wave functions are employed to represent the contact states of the character's feet—the left heel, left toe, right heel, and right toe—indicating ground contact at each time step. The foot contact matrix $\mathbf{F}$ is defined as:

$$\mathbf{F}(\omega, \tau) = \begin{bmatrix} \text{sign}\left(\sin(c\omega + a_h) - b_h - \tau l_h\right) \\ \text{sign}\left(\sin(c\omega + a_t) - b_t - \tau l_t\right) \\ \text{sign}\left(\sin(c\omega + a_h + \pi) - b_h - \tau r_h\right) \\ \text{sign}\left(\sin(c\omega + a_t + \pi) - b_t - \tau r_t\right) \end{bmatrix}^\top, \tag{3.9}$$

where ω is the angular frequency controlling the gait cycle, τ adjusts the step duration,

and c is a scaling factor. Constants a_h, a_t, b_h, and b_t are phase and amplitude modifiers for the heel and toe contacts, facilitating locomotion style adjustments. Variables l_h, l_t, r_h, and r_t represent the left heel, left toe, right heel, and right toe, respectively. The function sign$(\cdot)$ returns the sign of its argument, generating square waves.

This formulation simulates the periodic nature of foot contacts during locomotion, capturing the alternating patterns of foot placement in a gait cycle.

Deriving Wave Parameters from Motion Data

To reflect realistic gait dynamics, the parameters ω and τ are derived from motion capture data. The angular frequency ω_i at frame i is accumulated using incremental changes $\Delta\omega_i$:

$$\omega_i = \omega_{i-1} + \Delta\omega_i = \sum_{k=0}^{i} \Delta\omega_k, \tag{3.10}$$

where $\Delta\omega_i = \dfrac{\pi}{L_i}$, and L_i is the step wavelength at frame i, calculated by averaging the intervals between successive foot-off to foot-on transitions for all contact points.

The step duration parameter τ_i is computed by analyzing the proportion of the gait cycle during which each foot is grounded (down) versus lifted (up):

$$\tau_i = \cos\left(\frac{\pi d_i}{u_i + d_i}\right), \tag{3.11}$$

where d_i is the duration (in frames) the foot is in contact with the ground, and u_i is the duration it is lifted off. The cosine function maps the ratio of down-phase duration to the total gait cycle into a value that adjusts the step duration in the square wave representation.

These parameters are consolidated into the matrix:

$$\Gamma = \{\tau_{\mathrm{lh}}, \tau_{\mathrm{lt}}, \tau_{\mathrm{rh}}, \tau_{\mathrm{rt}}, \Delta\omega\}, \tag{3.12}$$

where τ_{lh}, τ_{lt}, τ_{rh}, and τ_{rt} correspond to the τ values for the left heel, left toe, right heel, and right toe, respectively, and $\Delta\omega$ contains the angular frequency increments.

Utilizing these parameters allows the square wave functions to accurately replicate the timing and duration of foot contacts observed in real motion capture data, leading to more realistic synthesized locomotion. synthesized locomotion.

Network and Training Hyperparameters

The network comprises two components: the footstep incorporation network and the MLP mapping network. The footstep incorporation network uses two convolutional layers with filter widths of 5 and 3, activated by ReLU functions, to capture the temporal patterns of foot contact. The MLP mapping network has three hidden layers consisting of 512, 256, and 128 neurons, each activated by ReLU. The autoencoder decoder, which remains fixed during training, reconstructs motion from the latent representations produced by the MLP.

For training, the Adam optimizer is used with a learning rate of 1×10^{-4} and a batch size of 32. The training process spans 50 epochs, and L2 regularization is applied with a weight decay coefficient $\lambda = 0.0001$ to prevent overfitting. These parameters were chosen based on experimentation to balance the training time and model accuracy.

3.4 Summary

In this chapter, we presented a methodology for mapping interactive control inputs to seamless human motions by integrating footstep contact information and utilizing a multilayer perceptron (MLP) to map control inputs into the latent space of a transformer-based autoencoder. This approach enables the generation of realistic and natural human motions that are responsive to high-level commands, addressing limitations of previous methods.

By incorporating transformer architectures with self-attention mechanisms, the autoencoder captures long-range dependencies more effectively than convolutional counterparts, resulting in smoother and more coherent motion sequences. Additionally, the modularity of the system allows the pretrained autoencoder's decoder to remain fixed during training, offering flexibility in adapting to various control schemes without retraining the entire system.

Overall, we have developed a flexible and effective framework for motion synthesis in interactive applications. The integration of footstep contact information and the transformer-based autoencoder enhances the realism and responsiveness of the generated human motions. This work contributes to the advancement of human motion synthesis techniques and lays a foundation for further research in this field.

3.5 Summary

In this chapter, we presented a methodology for mapping interactive control inputs to seamless human motions by integrating footstep contact information and utilizing a multilayer perceptron (MLP) to map control inputs into the latent space of a transformer-based autoencoder. This approach enables the generation of realistic and natural human motions that are responsive to interactive control inputs, addressing the limitations of previous methods.

- **Transformer-Based Autoencoder**: By incorporating transformer architectures with self-attention mechanisms, our autoencoder captures long-range dependencies more effectively than convolutional counterparts. This leads to smoother and more coherent motion sequences, as the model can better understand temporal relationships within the motion data.
- **Modularity and Efficiency**: By keeping the pretrained autoencoder's decoder fixed during training, we maintain a stable motion manifold while allowing the mapping network to be trained end-to-end. This modularity offers flexibility in adapting to various control schemes without retraining the entire system.

We have developed a flexible and effective framework for motion synthesis in interactive applications. The integration of footstep contact information and the use of a transformer-based autoencoder enable the generation of realistic and seamless human motions from interactive control inputs. This work contributes to the advancement of text-conditioned and action-conditioned techniques and pushing the boundaries of responsive human motion synthesis.

Insights

The incorporation of transformer architectures demonstrates the potential for models that better capture the complex temporal dynamics of human motion. Our findings suggest that self-attention mechanisms are particularly effective in understanding long-range dependencies, which are crucial for realistic motion synthesis.

Integrating biomechanical constraints, such as footstep contact information, highlights the importance of embedding physical plausibility into motion synthesis models. This approach not only improves realism but also opens avenues for incorporating additional constraints, such as balance and momentum conservation, in future work.

Chapter 4

Evaluation

In this chapter, we present a comprehensive evaluation of our proposed transformer-based autoencoder for human motion synthesis. The goal is to assess the performance of our model in mapping interactive control inputs to smooth and realistic human motions. We compare our approach with baseline models using the CMU Motion Capture (CMU MoCap) dataset and provide both quantitative and qualitative analyses. Additionally, we discuss the evaluation metrics relevant to human motion synthesis and provide insights into the strengths and limitations of our method.

4.1 Experimental Setup

All experiments were conducted using Google Colab with access to NVIDIA A100 GPUs. Due to the cloud-based environment, detailed hardware specifications beyond the GPU are not available. The deep learning framework used is PyTorch 1.10, with additional libraries such as NumPy, SciPy, and Matplotlib for data processing and visualization. The use of Google Colab provides sufficient computational resources for training deep neural networks while offering ease of access and reproducibility.

4.2 Benchmark Datasets

4.2.1 CMU Motion Capture Dataset

The primary dataset used for evaluation is the CMU Motion Capture dataset (Carnegie Mellon University, 2003), which contains a wide variety of human motion recordings, including walking, running, jumping, and other complex actions. The dataset comprises over 2,600 motion sequences captured from 144 subjects performing various activities.

Justification: The CMU MoCap dataset is widely used in the motion synthesis community due to its diversity and availability. It provides a rich set of motions that are essential

for training and evaluating models aiming to generalize across different movement patterns. The CMU Motion Capture (CMU MoCap) dataset is a widely used benchmark in human motion synthesis research. It contains over 2,600 motion sequences captured from 144 subjects performing various activities.

4.2.2 AMASS Dataset

The Archive of Motion Capture as Surface Shapes (AMASS) dataset (Mahmood et al., 2019) aggregates motion capture data from multiple sources into a unified format compatible with the Skinned Multi-Person Linear (SMPL) model. AMASS offers a more extensive and diverse collection of human motions compared to CMU MoCap. By leveraging AMASS data and rendering it using SMPL (Loper et al., 2015b), we enhance the realism and variability of the motion data, which contributes to the improved performance of our transformer-based autoencoder.

Data Preprocessing

For the CMU MoCap dataset, standard preprocessing steps are applied. Joint positions and rotations are normalized to ensure consistent scaling across the data. Motion sequences are resampled to a uniform frame rate of 30 frames per second, and long motion sequences are segmented into shorter, fixed-length clips to facilitate batch training.

Data Splits

The dataset is divided into training, validation, and test sets. The training set comprises 70% of the data for model training, while 15% is allocated to the validation set for hyperparameter tuning and early stopping. The remaining 15% forms the test set, used for final model evaluation. Subjects are partitioned such that motions from specific subjects appear only in one of the splits to ensure that the models generalize to unseen individuals.

4.3 Results and Analysis

4.3.1 Evaluation Metrics

In evaluating human motion synthesis models, it is crucial to select metrics that align with our objectives and effectively measure the aspects of motion most relevant to our work. Given our goal to assess the performance of our transformer-based autoencoder in mapping interactive control inputs to smooth and realistic human motions, we have selected the following metrics.

Mean Squared Displacement (MSD)

Mean Squared Displacement (MSD) measures the average squared distance that a joint moves between consecutive frames. It assesses the smoothness of the reconstructed motion sequences, with lower MSD values indicating smoother motions. MSD is particularly

important for evaluating the temporal consistency of motion, ensuring that there are no abrupt changes that could lead to unnatural movements.

The MSD is computed using the following equation:

$$\text{MSD} = \frac{1}{N} \sum_{i=1}^{N} \|\mathbf{x}_{i+1} - \mathbf{x}_i\|_2^2 \tag{4.1}$$

where:

- $\mathbf{x}_i$ represents the position vector of all joints at frame i.
- N is the total number of frames minus one (since we are looking at frame differences).
- $\|\cdot\|_2$ denotes the Euclidean norm.

A comparison of MSD values for both the convolutional and transformer autoencoders is shown in Table 4.1. This comparison allows us to quantitatively demonstrate the smoothness of the reconstructed motions.

Table 4.1: Mean Squared Displacement (MSD) Comparison for Convolutional and Transformer Autoencoders

Model	**Mean Squared Displacement (MSD)**
Convolutional Autoencoder	7.6817
Transformer Autoencoder	**1.4562**

As shown in the table, the transformer autoencoder yields a lower MSD compared to the convolutional autoencoder, indicating that it generates smoother motion sequences.

Mean Per Joint Position Error (MPJPE)

The Mean Per Joint Position Error (MPJPE) measures the average Euclidean distance between the predicted joint positions and the ground truth positions across all joints and frames. This metric directly assesses the spatial accuracy of the reconstructed motions, which is essential for realistic human motion synthesis.

MPJPE is calculated using the following equation:

$$\text{MPJPE} = \frac{1}{T \times J} \sum_{t=1}^{T} \sum_{j=1}^{J} \left\| \mathbf{p}_{j,t}^{\text{pred}} - \mathbf{p}_{j,t}^{\text{gt}} \right\|_2 \tag{4.2}$$

where:

- T is the total number of frames.

- J is the number of joints.
- $\mathbf{p}_{j,t}^{\text{pred}}$ is the predicted position of joint j at frame t.
- $\mathbf{p}_{j,t}^{\text{gt}}$ is the ground truth position of joint j at frame t.
- $\|\cdot\|_2$ denotes the Euclidean norm.

Lower MPJPE values indicate higher accuracy in joint position reconstruction.

Mean Absolute Error (MAE)

Mean Absolute Error (MAE) measures the average absolute difference between the predicted motion vectors and the ground truth motion vectors. It provides a general measure of reconstruction quality, capturing the average magnitude of errors without emphasizing larger errors disproportionately.

MAE is calculated as:

$$\text{MAE} = \frac{1}{T} \sum_{t=1}^{T} \left\| \mathbf{x}_t^{\text{pred}} - \mathbf{x}_t^{\text{gt}} \right\|_1 \tag{4.3}$$

where:

- $\mathbf{x}_t^{\text{pred}}$ and $\mathbf{x}_t^{\text{gt}}$ are the predicted and ground truth motion vectors at frame t, respectively.
- $\|\cdot\|_1$ denotes the L_1 norm (sum of absolute differences).

Root Mean Squared Error (RMSE)

Root Mean Squared Error (RMSE) measures the square root of the average squared differences between the predicted and ground truth motion vectors. RMSE is sensitive to larger errors due to the squaring of differences before averaging, thus providing insight into the presence of significant deviations in reconstruction.

RMSE is calculated as:

$$\text{RMSE} = \sqrt{\frac{1}{T} \sum_{t=1}^{T} \left\| \mathbf{x}_t^{\text{pred}} - \mathbf{x}_t^{\text{gt}} \right\|_2^2} \tag{4.4}$$

Justification of Metrics

While some of these metrics may appear similar or related, each provides unique insights into different aspects of the model's performance. Including all of them ensures a comprehensive evaluation:

- **MSD vs. MPJPE**: MSD focuses on the smoothness of motion by measuring the displacement of joints between consecutive frames, capturing temporal consistency. In contrast, MPJPE assesses the spatial accuracy of joint positions with respect to the ground truth across all frames. They address different dimensions of motion quality—temporal smoothness and spatial accuracy.
- **MAE vs. RMSE**: Both MAE and RMSE evaluate the overall reconstruction error, but they differ in sensitivity to outliers. MAE computes the average magnitude of errors, treating all deviations equally, whereas RMSE squares the errors before averaging, thus giving more weight to larger errors. Including both metrics allows us to understand not only the average performance but also the impact of significant deviations.
- **Importance of Multiple Metrics**: Human motion synthesis is a complex task that involves reproducing accurate joint positions (spatial accuracy), ensuring smooth transitions (temporal consistency), and maintaining natural motion dynamics. By utilizing a combination of these metrics, we can thoroughly assess how well the models perform across these critical aspects.
- **Complementary Perspectives**: Each metric complements the others by highlighting different strengths and weaknesses of the models. For instance, a model may achieve low MPJPE (accurate joint positions) but have high MSD (less smooth motion), indicating that while the positions are accurate, the motion lacks fluidity. Analyzing multiple metrics enables a more nuanced understanding of model performance.

Conclusion Including MSD, MPJPE, MAE, and RMSE in our evaluation provides a well-rounded assessment of the models' capabilities in generating realistic human motions. These metrics collectively measure spatial accuracy, temporal smoothness, and overall reconstruction quality, which are all essential for effective human motion synthesis. By analyzing the results across these metrics, we can identify specific areas where models excel or need improvement, guiding future enhancements and ensuring that the synthesized motions meet the desired standards of realism and naturalness.

4.3.2 Quantitative Results and Analysis

In this section, we present a comprehensive quantitative evaluation of our proposed transformer-based autoencoder compared to the convolutional autoencoder baseline. The analysis focuses on the models' abilities to reconstruct human motion sequences accurately and smoothly, leveraging the evaluation metrics defined earlier: Mean Squared Displacement (MSD), Mean Per Joint Position Error (MPJPE), Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), and Mean Acceleration Error (MAC).

Baseline Model

To assess the effectiveness of our transformer-based framework, we compare it to a convolutional baseline model adapted from Holden et al. (2016). This model employs

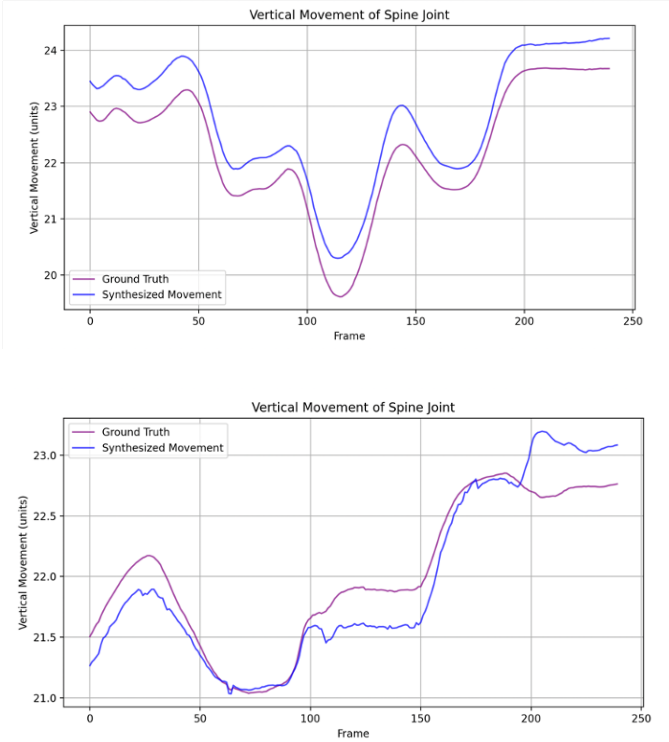


Figure 4.1: This figure compares the vertical movement of the spine joint between the ground truth and the synthesized motion, illustrates the movement in terms of vertical displacement (in meters) over frames for the ground truth (blue line) and synthesized movement (purple line), showing how closely the synthesized motion follows the true motion pattern across the entire motion sequence.

convolutional layers for both encoding and decoding human motion sequences, learning to reconstruct motion manifolds by capturing both temporal and spatial correlations in complex motion capture data. Given its proven success in prior research, it serves as a robust baseline. We replicate its results using the CMU MoCap dataset to ensure a fair comparison with our proposed method.

Overall Performance Comparison

Table 4.2 summarizes the performance of both models across the selected metrics. And their performance over these evaluation metrics are shown in Figure 4.2.

The transformer autoencoder outperforms the convolutional autoencoder in most metrics. Notably, it achieves significantly lower MSD and MAC values, indicating smoother motion sequences and better temporal dynamics. The lower MPJPE demonstrates higher accuracy in reconstructing joint positions, essential for realistic human motion

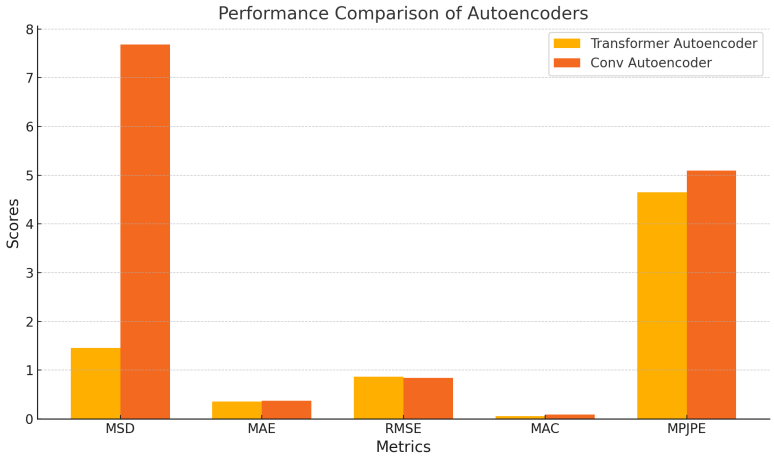


Figure 4.2: Performance Comparison on Evaluation Metrics

Table 4.2: Performance Comparison of Autoencoder Models

Model	MSD	MAE	RMSE	MAC	MPJPE
Transformer AE	**1.4562**	**0.3560**	0.8656	**0.0559**	**4.6513**
Convolutional AE	7.6817	0.3762	**0.8431**	0.0932	5.0940

synthesis.

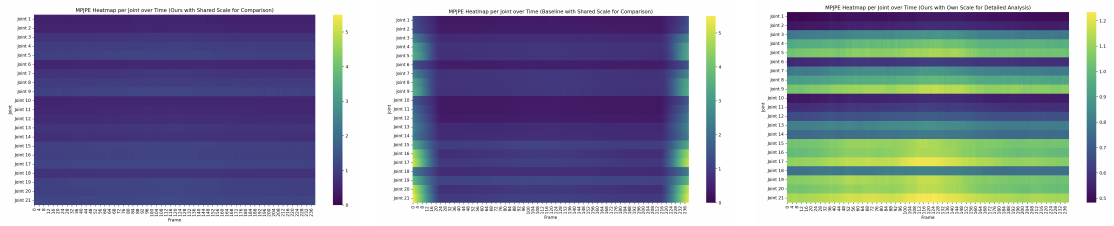
Analysis of Motion Smoothness (MSD and MAC)

The Mean Squared Displacement (MSD) and Mean Acceleration Error (MAC) are critical for assessing the smoothness and temporal consistency of the reconstructed motions. The transformer autoencoder's MSD of 1.4562 is substantially lower than the convolutional autoencoder's 7.6817, suggesting that the transformer model produces smoother transitions between consecutive frames.

Similarly, the transformer autoencoder achieves a lower MAC of 0.0559 compared to 0.0932 for the convolutional autoencoder. A lower MAC indicates better capture of acceleration patterns, leading to more natural and fluid movements. These results highlight the transformer model's ability to effectively model temporal dependencies through self-attention mechanisms.

Analysis of Reconstruction Accuracy (MPJPE, MAE, RMSE)

In terms of spatial accuracy, the transformer autoencoder attains a lower MPJPE of 4.6513 versus 5.0940 for the convolutional autoencoder, demonstrating more precise joint position reconstruction. The MAE values further support this finding, with the transformer model achieving a lower MAE of 0.3560 compared to 0.3762.



(a) Transformer AE (Shared Scale)

(b) Convolutional AE (Shared Scale)

(c) Transformer AE (Own Scale)

Figure 4.3: Comparison of MPJPE Heatmaps per Joint over Time. (a) and (b) use a shared scale for comparison, while (c) uses the transformer's own scale for detailed analysis.

However, the convolutional autoencoder records a slightly better RMSE of 0.8431 against the transformer's 0.8656. Since RMSE is more sensitive to larger errors, this suggests that while the transformer model generally produces accurate reconstructions, it may exhibit larger errors in certain instances. This observation points to potential areas for improvement, such as refining the model to handle outliers more effectively.

Heatmap Analysis

To provide a visual comparison, we present the MPJPE heatmaps as per joint over time for both models in Figure 4.3.

Heatmaps (a) and (b) allow direct comparison using a shared color scale. The transformer autoencoder's heatmap (a) shows predominantly cooler colors (blue regions), indicating lower error values across joints and frames. In contrast, the convolutional autoencoder's heatmap (b) displays warmer colors (yellow and green regions), especially in later frames and specific joints, revealing higher errors.

Heatmap (c) uses the transformer autoencoder's own scale to provide a detailed view of its performance. While overall errors are low, certain joints and time frames exhibit relatively higher MPJPE values. This fine-grained analysis suggests that although the transformer model consistently outperforms the convolutional model, there are specific areas where further improvements are possible.

Impact of Model Architectures and Hyperparameters

The superior performance of the transformer autoencoder can be attributed to its architectural advantages. The self-attention mechanism enables the model to capture long-range dependencies and complex temporal dynamics more effectively than convolutional architectures, which primarily focus on local patterns.

Both models were trained with similar hyperparameters: a learning rate of 1×10^{-5} and a batch size of 16. The convolutional autoencoder employed a dropout rate of

0.2 to mitigate overfitting, whereas the transformer autoencoder did not use dropout, relying instead on inherent regularization provided by layer normalization and attention mechanisms.

Table 4.3 compares the training and test losses for both models.

Table 4.3: Training and Test Loss Comparison for Autoencoders with $\alpha = 0.0006$

Model	Dropout	Learning Rate	Training Loss	Test Loss
Convolutional AE	0.2	1×10^{-5}	1.068	0.925
Transformer AE	N/A	1×10^{-5}	0.798	**0.120**

The transformer autoencoder achieves a significantly lower test loss, indicating better generalization to unseen data. This result underscores the effectiveness of the transformer architecture in modeling human motion sequences.

Insights and Implications

The analysis reveals that the transformer autoencoder outperforms the convolutional autoencoder in capturing both spatial and temporal aspects of human motion. The self-attention mechanisms allow the transformer model to understand complex dependencies across time, leading to enhanced motion smoothness and accuracy. Despite the overall superior performance, the transformer autoencoder exhibits slightly higher RMSE values. This suggests that while the model excels in general reconstruction, it may be susceptible to larger errors in specific instances. Addressing this could involve refining the model's capacity to handle outliers or incorporating additional regularization techniques. The heatmap analysis identifies specific joints and time frames where errors are relatively higher. Future work could focus on targeted improvements in these areas, potentially through enhanced attention mechanisms or specialized training strategies.

Conclusion

The quantitative evaluation demonstrates that the transformer-based autoencoder offers significant advantages over the convolutional autoencoder in human motion synthesis tasks. By effectively capturing long-range dependencies and complex temporal dynamics, the transformer model achieves smoother and more accurate motion reconstructions. These findings highlight the transformative potential of self-attention mechanisms in modeling sequential data, particularly in applications requiring nuanced understanding of temporal relationships. The insights gained from this analysis provide a foundation for further advancements in human motion synthesis and related fields.

4.3.3 Qualitative Results and Analysis

The qualitative assessment demonstrates that our transformer-based autoencoder generates highly realistic motion sequences. As shown in Figure 4.1, the vertical movements of the spine joints closely align with real human motions, ensuring smooth and natural dynamics. In visual comparisons, Figure 4.4 illustrates that the baseline convolutional

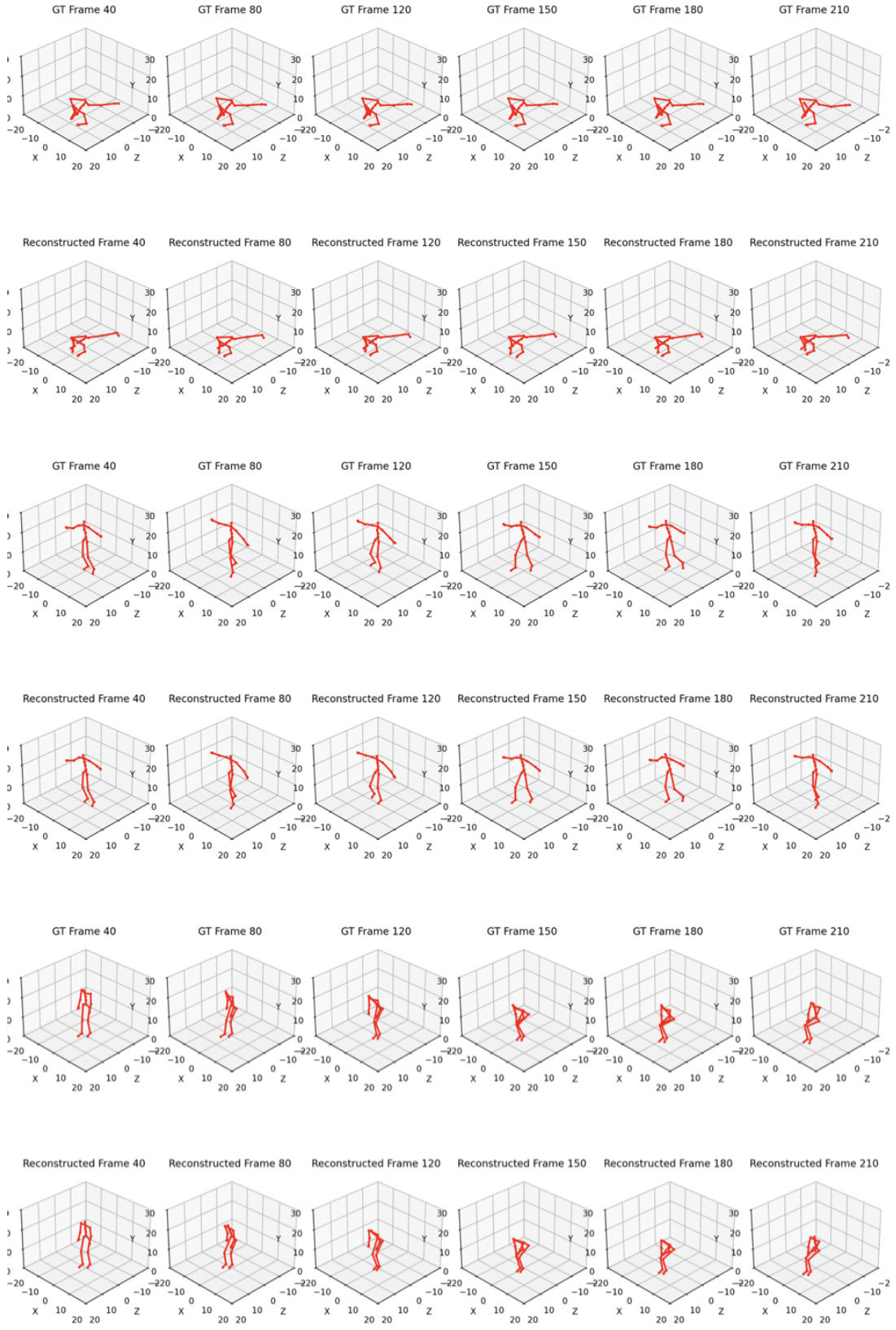


Figure 4.4: Comparison between ground truth and reconstructed motion frames at selected intervals for the baseline model.

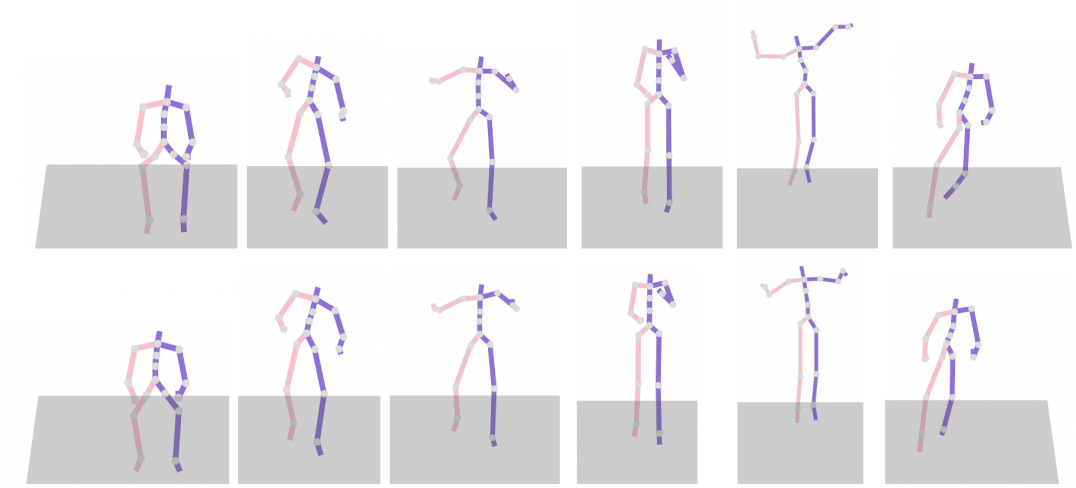


Figure 4.5: Comparison of Motion Sequences: Ground Truth (Up) vs.Reconstructed (Down).

Figure 4.6: Motion Sequence Rendered in SMPL

autoencoder effectively mirrors the ground truth but exhibits minor discrepancies in joint details and temporal transitions. In contrast, Figure 4.5, 4.7 and 4.6 highlight our model's superior performance, accurately capturing dynamic transitions and joint relationships, which results in noticeably smoother and more authentic motion sequences.

Observations

Our method produces smoother and more natural motions with accurate foot placement, avoiding artifacts such as foot sliding. The incorporation of footstep contact information significantly enhances the realism of locomotion. The visual comparisons demonstrate that the our work is better at capturing complex motion patterns and maintaining temporal coherence. In contrast, the baseline exhibits less precise joint positions and may produce less smooth transitions, as evident from the reconstructed sequences.

Impact of Self-Attention Mechanism To assess the impact of the self-attention

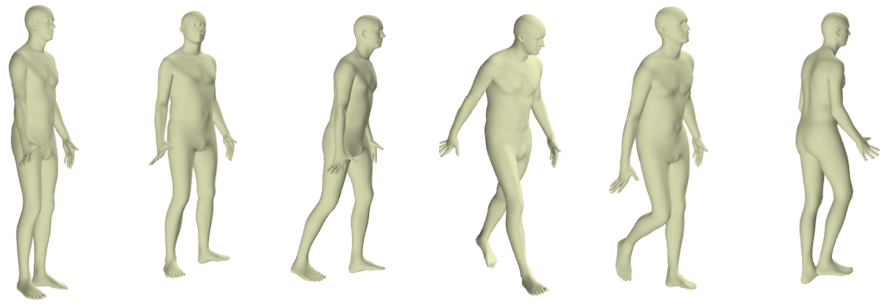


Figure 4.7: Motion Sequence Rendered in SMPL

mechanism, we replaced the transformer encoder with a standard feedforward network and observed the performance. The model without self-attention resulted in higher MPJPE and less coherent motions, indicating that the self-attention mechanism is effective in capturing long-range dependencies and improving motion reconstruction.

Analysis and Insights

The qualitative comparisons indicate that our transformer autoencoder outperforms the convolutional baseline in reproducing realistic human motions. The enhanced performance can be attributed to the transformer's ability to model long-range dependencies and capture complex temporal dynamics through self-attention mechanisms. This results in more accurate joint movements and smoother transitions, which are critical for the perception of natural motion.

The visual alignment of the spine joints' vertical movements demonstrates the model's proficiency in preserving biomechanical nuances, further contributing to the realism of the synthesized motions. By effectively capturing both spatial and temporal aspects of human motion, our model addresses limitations observed in the convolutional autoencoder, such as inadequate handling of subtle joint movements and temporal inconsistencies.

These qualitative observations corroborate the quantitative results discussed earlier, reinforcing the effectiveness of our proposed approach in human motion synthesis. The ability to generate high-fidelity motion sequences has significant implications for applications requiring realistic human animation, such as virtual reality, gaming, and simulation environments.

4.4 Summary

In this chapter, we conducted a comprehensive evaluation of our proposed transformer-based autoencoder for human motion synthesis. Through quantitative metrics and qualitative analyses, we demonstrated that our model outperforms the convolutional autoen-

coder baseline in reconstructing accurate and realistic human motions. The self-attention mechanisms in the transformer architecture enable the model to capture long-range dependencies and complex temporal dynamics, resulting in smoother and more coherent motions.

We provided detailed analyses of the models' performances using evaluation metrics such as MPJPE, MSD, MAE, RMSE, and MAC, and visualized the error distributions using heatmaps. Our qualitative assessments confirmed that the transformer autoencoder produces more natural motions with accurate foot placements.

4.4.1 Limitation

Despite its improved performance, the transformer-based model has some limitations. While the model produces smooth and accurate motions, occasional larger errors (as indicated by higher RMSE) suggest challenges with certain complex motions. Additionally, reliance on specific datasets like CMU MoCap and AMASS may limit generalization to other types of human movement. Finally, the model has not yet been tested in highly dynamic, real-time environments like virtual reality or gaming. Addressing these issues will enhance its scalability and adaptability.

Chapter 5

Conclusion

This thesis advances the task of human motion synthesis by developing a transformer-based autoencoder that maps interactive control inputs to seamless and realistic human movements. The proposed framework addresses the limitations of existing methods, which often rely on text descriptions or predefined action labels and struggle to produce natural transitions in response to intuitive controls like joysticks or touch inputs. By leveraging the capabilities of transformer architectures and integrating them with the Skinned Multi-Person Linear (SMPL) model, we have developed a system that enhances both the naturalness and responsiveness of motion synthesis. This work offers a streamlined solution with broad applications in gaming, virtual reality, and other interactive environments. We believe that the insights and methodologies presented in this thesis will drive future research and innovation in human motion synthesis and related fields, ultimately leading to more immersive and interactive digital experiences.

5.1 Contributions

The contributions of this thesis are threefold. First, we introduce a transformer-based autoencoder specifically designed for human motion synthesis, which captures complex temporal and spatial dependencies through self-attention mechanisms. This results in smoother and more natural synthesized motions compared to conventional models, such as convolutional or recurrent neural networks. Second, we develop a user-friendly framework that directly translates interactive control inputs into human motion. Unlike text-based or action label-based inputs, this method allows for more intuitive control through devices like joysticks or touchscreens, enabling real-time motion generation that accurately reflects user intent. Third, we integrate the transformer-based autoencoder with the SMPL model, ensuring anatomically accurate and physically plausible motion synthesis. This integration broadens the applicability of the model for use cases requiring high-fidelity human motion representation, such as virtual reality and animation.

5.2 Future Work

The research presented in this thesis introduces a transformer-based autoencoder framework that effectively maps intuitive control inputs to realistic human motion synthesis. While this work has achieved significant advancements in improving the naturalness, responsiveness, and scalability of motion generation, there remain several areas for further exploration and enhancement. In particular, two promising directions for future work include integrating terrain adaptation into the motion synthesis process and further advancing the interactive control mechanisms. These areas hold the potential to address existing limitations by expanding the applicability of human motion synthesis models in dynamic and interactive environments. This section outlines two primary approaches for incorporating terrain-aware motion generation and enhancing real-time user interaction, building upon the foundational contributions of this research.

5.2.1 Terrain-Aware Motion Generation

While the Human Motion Diffusion Model (MDM) represents the state of the art in generating diverse motion sequences, it does not address the challenge of handling varying terrain. In contrast, the work presented in this thesis demonstrates the transformer encoder's exceptional capacity for capturing complex motion data across different modalities. This insight opens the door to using the transformer encoder to incorporate terrain height information into the existing MDM architecture.

Adapting the MDM to account for terrain height variations would enable it to generate more realistic and natural motion for characters walking on uneven surfaces. The challenge lies in how to effectively integrate height information into the diffusion model's workflow. The following are two potential approaches for achieving this goal:

Approach: Positional Encoding and Conditioning with Terrain Height

Incorporating terrain height into the transformer encoder via positional encoding constitutes a logical extension of the current framework. Transformers already employ positional encodings to capture the temporal structure of motion sequences; thus, an additional dimension representing terrain height could be introduced to enable the model to distinguish between movements on flat versus uneven surfaces.

This modification would involve expanding the positional encoding to include terrain height as an additional dimension. Terrain height can be represented as a time-varying signal that is fed into the model alongside the motion data. Such an enhancement enables the transformer to encode both temporal information and height variations in the terrain. This approach leverages the transformer's inherent sequence modeling capabilities with minimal architectural adjustments. By embedding height information within the positional encoding, the transformer can learn motion dynamics responsive to terrain changes, allowing it to adapt naturally to variations in surface height.

This approach parallels the way text conditioning is handled in the MDM model, yet ac-

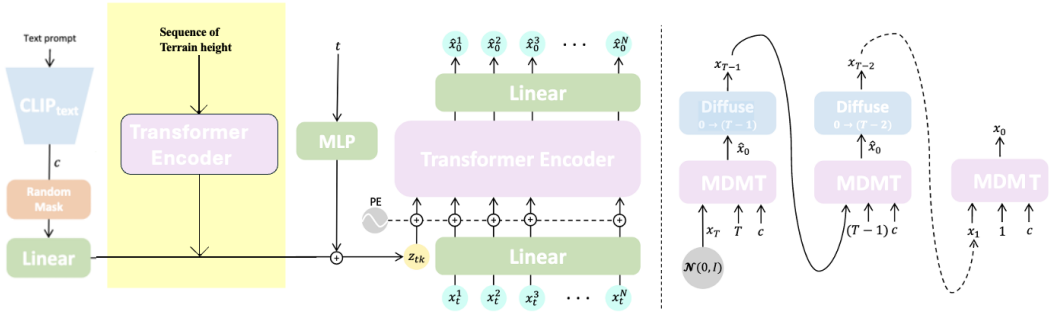


Figure 5.1: Architecture of the MDMT network, adapted from the original MDM model. The diagram illustrates the incorporation of terrain height encoding alongside motion data, enabling the model to account for terrain variations in human motion synthesis. Figure is revised from (Tevet et al., 2022b)

commodates the dynamic nature of terrain height changes over time. By treating terrain height as a dynamic conditioning variable, the model can adjust to terrain fluctuations on a frame-by-frame basis, enabling more realistic and terrain-aware motion generation. The network architecture of the proposed approach, termed "MDMT" and adapted from MDM, is presented in Figure 5.1.

5.2.2 Advancing Interactive Control Inputs

Building on the MDM as a backbone, this thesis focuses on integrating interactive control inputs, such as velocity components (V_x, V_z) and rotation (R_y), to guide the generated motion. During training, these control inputs can be provided for each frame, allowing the model to learn how to generate motions that respond to user inputs in real-time. Both of the terrain-aware approaches discussed above could be applied in conjunction with these control inputs, thereby enhancing the model's ability to generate responsive and interactive motions.

As highlighted in Chapter 1, many current models, including MDM, are either text-conditioned or action-conditioned. However, integrating real-time control inputs, as demonstrated in this thesis, represents a significant advancement, enabling a more interactive and responsive system for human motion synthesis.

Justification for the Proposed Methodology

Terrain-Aware Motion Generation This integrated approach takes full advantage of the transformer's ability to model complex sequences and handle multi-modal inputs.

By embedding terrain height both in the positional encoding and as a conditioning signal, the model can efficiently capture the relationship between terrain changes and motion dynamics. Moreover, this design minimizes architectural changes while enhancing the model's ability to generate realistic, terrain-adaptive motion. The combined use of positional encoding and dynamic conditioning allows for flexible and context-sensitive motion generation without sacrificing the efficiency of the transformer framework.

Advancing Interactive Control Inputs The decision to incorporate joystick-based control inputs (section 3.3.1) and terrain adaptation into future work is driven by the need for more interactive and adaptable motion synthesis systems. This thesis along with recent advance, such as MotionCLIP and MDM, have demonstrated the strength of transformer-based models in aligning motion with text and action labels. However, these methods do not account for real-time control inputs or environmental changes, such as varying terrain heights. By introducing interactive controls, such as velocity components (V_x, V_z) and rotation (R_y), into the transformer encoder, the proposed methodology can provide a more natural and responsive motion synthesis experience. Additionally, incorporating terrain height as part of the input sequence, rather than treating it as a static condition, allows the model to generalize to different environmental contexts. By building on these advancements, the future work aims to enhance the versatility and realism of motion synthesis for a broader range of applications.

Relation to Thesis Contributions

The future work proposed here builds directly on the foundation established by the transformer-based autoencoder explored in this thesis. The transformer architecture has already proven effective in human motion synthesis, particularly in mapping intuitive control inputs to realistic human motions. Extending this methodology by moving from static input forms, such as text, to more interactive and terrain-aware control mechanisms will increase the flexibility and practicality of transformer-based models.

This future work highlights the transformer's self-attention mechanism, which is well-suited to capturing the dynamic nature of human motion and adapting to environmental challenges. Incorporating terrain-aware motion generation and interactive control inputs not only expands the capabilities of the transformer autoencoder but also strengthens its application in interactive systems, such as gaming and virtual reality. Ultimately, this work provides a clear direction for the continued evolution of human motion synthesis models and contributes to the broader field by enabling more adaptable, real-time, and context-aware motion generation.

Bibliography

Aksan, E.; Kaufmann, M.; and Hilliges, O., 2020. Attention, please! a spatio-temporal transformer for 3d human motion prediction. *arXiv preprint arXiv:2004.08692*, (2020). [Cited on pages 34 and 42.]

Andriluka, M.; Iqbal, U.; Ensafutdinov, E.; Pishchulin, L.; Milan, A.; Gall, J.; and Schiele, B., 2018. Posetrack: A benchmark for human pose estimation and tracking. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 5167–5176. [Cited on page 18.]

Arjovsky, M.; Chintala, S.; and Bottou, L., 2017. Wasserstein gan. https://arxiv.org/abs/1701.07875. [Cited on page 13.]

Athanasiou, N.; Petrovich, M.; Black, M. J.; and Varol, G., 2022. Teach: Temporal action composition for 3d humans. https://arxiv.org/abs/2209.04066. [Cited on page 23.]

Ba, J. L.; Kiros, J. R.; and Hinton, G. E., 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*, (2016). [Cited on page 14.]

Baltrušaitis, T.; Ahuja, C.; and Morency, L.-P., 2019. Multimodal machine learning: A survey and taxonomy. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41, 2 (2019), 423–443. [Cited on page 39.]

Bengio, Y.; Simard, P.; and Frasconi, P., 1994. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5, 2 (1994), 157–166. [Cited on page 7.]

Cai, Y.; Liu, J.; Wang, Y.; and Wang, C., 2021. UNIK: A unified framework for real-world human kinematics estimation and motion synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 12054–12063. [Cited on page 40.]

Carnegie Mellon University, 2003. Cmu graphics lab motion capture database. http://mocap.cs.cmu.edu/. Accessed: 2024-10-22. [Cited on pages 17, 50, and 66.]

Chen, X.; Jiang, B.; Liu, W.; Huang, Z.; Fu, B.; Chen, T.; Yu, J.; and Yu, G., 2023. Executing your commands via motion diffusion in latent space. https://arxiv.org/abs/2212.04048. [Cited on page 37.]

Chung, J.; Kastner, K.; Dinh, L.; Goel, K.; Courville, A. C.; and Bengio, Y., 2015. A recurrent latent variable model for sequential data. In *Advances in Neural Information Processing Systems*, vol. 28, 2980–2988. [Cited on pages 31 and 42.]

Dabral, R.; Mughal, M. H.; Golyanik, V.; and Theobalt, C., 2023. Mofusion: A framework for denoising-diffusion-based motion synthesis. https://arxiv.org/abs/2212.04495. [Cited on page 37.]

Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T.; Dehghani, M.; Minderer, M.; Heigold, G.; Gelly, S.; Uszkoreit, J.; and Houlsby, N., 2021a. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*. [Cited on page 10.]

Dosovitskiy, A. et al., 2021b. An image is worth 16x16 words: Transformers for image recognition at scale. *International Conference on Learning Representations (ICLR)*, (2021). [Cited on page 55.]

Dowson, D. C. and Landau, B. V., 1982. The fréchet distance between multivariate normal distributions. *Journal of Multivariate Analysis*, 12, 3 (1982), 450–455. [Cited on page 44.]

Elman, J. L., 1990. Finding structure in time. *Cognitive Science*, 14, 2 (1990), 179–211. [Cited on page 7.]

Godoy, D. V., 2021. Seq2seq rnn encoder-decoder with attention mechanism. https://commons.wikimedia.org/wiki/File:Seq2seq_RNN_encoder-decoder_with_attention_mechanism. CC BY 4.0 License. Accessed: 23 October 2024. [Cited on page 7.]

Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; and Bengio, Y., 2014. Generative adversarial nets. In *Advances in Neural Information Processing Systems*, 2672–2680. [Cited on pages 11 and 12.]

Guo, C. and Joo, H., 2020. Action2motion: Conditioned generation of 3d human motions. In *Proceedings of the ACM International Conference on Multimedia*, 2021–2029. [Cited on pages 23, 33, 42, and 44.]

Harvey, F. R.; Pal, C.; and Courville, A., 2020. Recurrent transition networks for character locomotion. In *Proceedings of the ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, 1–12. [Cited on pages 32 and 42.]

He, K.; Chen, X.; Xie, S.; Li, Y.; Dollár, P.; and Girshick, R., 2021. Masked autoencoders are scalable vision learners. *arXiv preprint arXiv:2111.06377*, (2021). [Cited on page 11.]

Hinton, G. E. and Salakhutdinov, R. R., 2006. Reducing the dimensionality of data with neural networks. *Science*, 313, 5786 (2006), 504–507. [Cited on page 8.]

Hochreiter, S. and Schmidhuber, J., 1997. Long short-term memory. *Neural Computation*, 9, 8 (1997), 1735–1780. [Cited on page 7.]

Holden, D.; Komura, T.; and Saito, J., 2017. Phase-functioned neural networks for character control. *ACM Transactions on Graphics*, 36, 4 (2017), 42:1–42:13. [Cited on pages 40 and 43.]

Holden, D.; Saito, J.; and Komura, T., 2016. A deep learning framework for character motion synthesis and editing. *ACM Transactions on Graphics (TOG)*, 35, 4 (2016), 1–11. [Cited on pages 19, 21, 44, 49, 50, 58, and 70.]

Holden, D.; Saito, J.; Komura, T.; and Joyce, T., 2015. Learning motion manifolds with convolutional autoencoders. In *SIGGRAPH Asia 2015 Technical Briefs*, SA '15 (Kobe, Japan, 2015). Association for Computing Machinery, New York, NY, USA. doi:10.1145/2820903.2820918. https://doi.org/10.1145/2820903.2820918. [Cited on page 8.]

Huang, Z.; Li, J.; Huang, Y.; Xiao, J.; and Zhu, H., 2020. Learning to generate diverse dance motions with transformer. *arXiv preprint arXiv:2008.08171*, (2020). [Cited on page 28.]

Ioffe, S. and Szegedy, C., 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, 448–456. [Cited on page 13.]

Ionescu, C.; Papava, D.; Olaru, V.; and Sminchisescu, C., 2014. Human3.6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36, 7 (2014), 1325–1339. [Cited on pages 17 and 44.]

Jiang, B.; Chen, X.; Liu, W.; Yu, J.; Yu, G.; and Chen, T., 2023. Motiongpt: Human motion as a foreign language. https://arxiv.org/abs/2306.14795. [Cited on page 36.]

Kim, K.; Park, S.; and Yu, S., 2022. Flame: Free-form language-based motion synthesis & editing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 14456–14465. [Cited on pages 39 and 43.]

Kingma, D. P. and Welling, M., 2014. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, (2014). [Cited on page 8.]

LeCun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P., 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86, 11 (1998), 2278–2324. [Cited on page 5.]

Lee, J.; Chai, J.; Reitsma, P. S.; Hodgins, J. K.; and Pollard, N. S., 2010. Data-driven footskate cleanup for motion capture editing. *ACM Transactions on Graphics*, 29, 4 (2010), 54:1–54:10. [Cited on page 41.]

Lee, J.; Shin, S.-H.; and Gleicher, M., 2002. Footplant detection and enforcement for effective character animation. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*, 536–544. [Cited on page 50.]

Li, J.; Huang, Z.; Li, W.; Wang, T.; Zhao, J.; and Komura, T., 2021. Ai choreographer: Music conditioned 3d dance generation with aist++. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 13401–13412. [Cited on pages 27, 28, and 40.]

Li, Z.; Li, Z.; and Li, Q., 2022. Learning skeletal graph neural networks for hard 3d pose estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44, 5 (2022), 2544–2556. [Cited on pages 40 and 43.]

Li, Z. and Zhang, Y., 2019. Learning to generate diverse dance motions with transformer. *arXiv preprint arXiv:1912.10229*, (2019). [Cited on page 35.]

Ling, H.; Habermann, M.; Xu, W.; Sunkavalli, K.; Pons-Moll, G.; and Theobalt, C., 2020. Character controllers using motion vaes. *ACM Transactions on Graphics*, 39, 4 (2020), 40:1–40:14. [Cited on pages 32 and 42.]

Loper, M.; Mahmood, N.; Romero, J.; Pons-Moll, G.; and Black, M. J., 2015a. Smpl: A skinned multi-person linear model. *ACM Transactions on Graphics*, 34, 6 (2015), 248:1–248:16. [Cited on page 16.]

Loper, M.; Mahmood, N.; Romero, J.; Pons-Moll, G.; and Black, M. J., 2015b. Smpl: A skinned multi-person linear model. In *ACM Transactions on Graphics (TOG)*, vol. 34, 1–16. ACM. [Cited on page 67.]

Mahmood, N.; Ghorbani, N.; Troje, N. F.; Pons-Moll, G.; and Black, M. J., 2019. Amass: Archive of motion capture as surface shapes. *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, (2019), 5442–5451. [Cited on pages 17 and 67.]

Mehta, D.; Rhodin, H.; Casas, D.; Fua, P.; Sotnychenko, O.; Xu, W.; and Theobalt, C., 2017. Monocular 3d human pose estimation in the wild using improved cnn supervision. In *2017 International Conference on 3D Vision (3DV)*, 506–516. [Cited on page 18.]

Miyato, T.; Kataoka, T.; Koyama, M.; and Yoshida, Y., 2018. Spectral normalization for generative adversarial networks. https://arxiv.org/abs/1802.05957. [Cited on page 13.]

Müller, M. and Röder, T., 2007. Documentation: Mocap database hdm05. In *Technical Report CG-2007-2, University of Bonn*. [Cited on page 50.]

Ofli, F.; Chaudhry, R.; Kurillo, G.; Bajcsy, R.; and Vitaladevuni, S. N., 2013. Berkeley mhad: A comprehensive multimodal human action database. *Proceedings of the IEEE Workshops on Applications of Computer Vision (WACV)*, (2013), 53–60. [Cited on page 50.]

Petrovich, M.; Black, M. J.; and Varol, G., 2021. Action-conditioned 3d human motion synthesis with transformer vaes. *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, (2021), 10985–10995. [Cited on pages 22, 23, 34, 42, and 44.]

Petrovich, M.; Black, M. J.; and Varol, G., 2022. Temos: Generating diverse human motions from textual descriptions. In *Proceedings of the European Conference on Computer Vision (ECCV)*. [Cited on page 24.]

Plappert, M.; Mandery, C.; and Asfour, T., 2016. The kit motion-language dataset. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, 286–292. [Cited on page 18.]

Pons-Moll, G.; Taylor, J.; Shotton, J.; Hertzmann, A.; and Fitzgibbon, A., 2015. Metric regression forests for human pose estimation. In *British Machine Vision Conference*, 1–12. [Cited on page 15.]

Punnakkal, A. V.; Varol, G.; Hilliges, O.; and Black, M. J., 2021. Babel: Bodies, action and behavior with english labels. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 722–731. [Cited on page 18.]

Radford, A.; Kim, J. W.; Hallacy, C.; et al., 2021. Learning transferable visual models from natural language supervision. In *Proceedings of the International Conference on Machine Learning*, 8748–8763. [Cited on pages 25 and 39.]

Ren, P.; Hu, W.; Li, J.; Wang, Z.; and Zhu, Q., 2020. Self-supervised neural motion prediction for real-time inference. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 6423–6432. [Cited on pages 20 and 21.]

Ren, Y.; Lee, C.; Zhang, X.; Li, Z.; Zhang, S.-C.; Yang, X.; and Cui, S. S., 2022. Bailando: 3d dance generation by actor-critic gpt with choreographic memory. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 11036–11045. [Cited on pages 28 and 35.]

Salimans, T.; Goodfellow, I.; Zaremba, W.; Cheung, V.; Radford, A.; and Chen, X., 2016. Improved techniques for training gans. https://arxiv.org/abs/1606.03498. [Cited on page 13.]

Siddiqui, Y.; Zhang, C.-H. P.; Tkach, A.; Tagliasacchi, A.; and Pauly, M., 2020. Total capture: A 3d deformation model for tracking faces, hands, and bodies. *arXiv preprint arXiv:2001.10952*, (2020). [Cited on page 18.]

Sigal, L. and Black, M. J., 2010. Humaneva: Synchronized video and motion capture dataset and baseline algorithm for evaluation of articulated human motion. *International Journal of Computer Vision*, 87, 1-2 (2010), 4–27. [Cited on page 15.]

Srivastava, N.; Hinton, G.; Krizhevsky, A.; Sutskever, I.; and Salakhutdinov, R., 2014. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15, 1 (2014), 1929–1958. [Cited on pages 13 and 14.]

Sutskever, I.; Vinyals, O.; and Le, Q. V., 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, 3104–3112. [Cited on page 8.]

Taylor, G. W.; Hinton, G. E.; and Roweis, S. T., 2007. Modeling human motion using binary latent variables. In *Advances in Neural Information Processing Systems*, 1345–1352. [Cited on page 19.]

Tevet, G.; Gordon, B.; Hertz, A.; Bermano, A. H.; and Cohen-Or, D., 2022a. Motionclip: Exposing human motion generation to clip space. https://arxiv.org/abs/2203.08063. [Cited on pages 26, 39, and 43.]

Tevet, G.; Raab, S.; Gordon, B.; Shafir, Y.; Cohen-Or, D.; and Bermano, A. H., 2022b. Human motion diffusion model. https://arxiv.org/abs/2209.14916. [Cited on pages 25 and 81.]

Varol, G.; Laptev, I.; and Schmid, C., 2017a. Long-term temporal convolutions for action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40, 6 (2017), 1510–1517. [Cited on page 6.]

Varol, G.; Romero, J.; Martin, X.; Mahmood, N.; Black, M. J.; Laptev, I.; and Schmid, C., 2017b. Learning from synthetic humans. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, (2017), 109–117. [Cited on page 18.]

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I., 2017a. Attention is all you need. In *Advances in Neural Information Processing Systems*, 5998–6008. [Cited on page 9.]

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I., 2017b. Attention is all you need. In *Advances in neural information processing systems*, 5998–6008. [Cited on pages 34 and 54.]

von Marcard, T.; Pons-Moll, G.; Rosenhahn, B.; and Black, M. J., 2018. Recovering accurate 3d human pose in the wild using imus and a moving camera. In *Proceedings of the European Conference on Computer Vision (ECCV)*, 601–617. [Cited on page 18.]

Wang, H.; Chen, W.; and Tulyakov, S., 2021a. Imitating arbitrary human style in motion and appearance. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 11222–11231. [Cited on page 40.]

Wang, H.; Chen, W.; and Tulyakov, S., 2021b. Motion style transfer between humans and robots. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, 1194–1200. [Cited on page 29.]

Wang, J.; Rong, Y.; Liu, J.; Yan, S.; Lin, D.; and Dai, B., 2022. Towards diverse and natural scene-aware 3d human motion synthesis. https://arxiv.org/abs/2205.13001. [Cited on page 30.]

Wang, W.; Lin, Z.; Zhang, S.; He, Z.; Liu, Y.; and Zhou, Y., 2021c. Scene-aware generative network for human motion synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8208–8217. [Cited on pages 29 and 44.]

Yamane, K. and Nakamura, Y., 2003a. Dynamics filter—concept and implementation of online motion generator for human figures. *IEEE Transactions on Robotics and Automation*, 19, 3 (2003), 421–432. [Cited on page 41.]

Yamane, K. and Nakamura, Y., 2003b. Simulating and generating motions of human figures. *The International Journal of Robotics Research*, 23, 3 (2003), 293–308. [Cited on page 50.]

Yuan, Y.; Song, J.; Iqbal, U.; Vahdat, A.; and Kautz, J., 2023. Physdiff: Physics-guided human motion diffusion model. https://arxiv.org/abs/2212.02500. [Cited on pages 37 and 43.]

Zhang, J.; Zhang, Y.; Cun, X.; Huang, S.; Zhang, Y.; Zhao, H.; Lu, H.; and Shen, X., 2023a. T2m-gpt: Generating human motion from textual descriptions with discrete representations. https://arxiv.org/abs/2301.06052. [Cited on pages 25 and 35.]

Zhang, M.; Guo, X.; Pan, L.; Cai, Z.; Hong, F.; Li, H.; Yang, L.; and Liu, Z., 2023b. Remodiffuse: Retrieval-augmented motion diffusion model. https://arxiv.org/abs/2304.01116. [Cited on page 37.]

Zhang, Y. and Wang, J., 2022. Motiondiffuse: Text-driven human motion generation with diffusion model. *arXiv preprint arXiv:2208.15001*, (2022). [Cited on pages 25, 37, and 43.]

Zhou, X.; Huang, Q.; Sun, X.; Xue, X.; and Wei, Y., 2017. Towards 3d human pose estimation in the wild: a weakly-supervised approach. In *Proceedings of the IEEE International Conference on Computer Vision*, 398–407. [Cited on pages 20 and 21.]