Modeling Variation of Human Motion

A dissertation submitted towards the degree
Doctor of Engineering
of the Faculty of Mathematics and Computer Science
of Saarland University

by

Han Du

Saarbrücken, 2022

Day of Colloquium	04 / 05 / 2023
Dean of the Faculty	Univ.-Prof. Dr. Jürgen Steimle
Chair of the Committee	Prof. Dr.-Ing. Thorsten Herfet
Reporters	
	Prof. Dr.-Ing. Philipp Slusallek
	Prof. Dr.-Ing. Martin Manns
	Prof. Dr. Dietrich Klakow
Academic Assistant	Dr. Klaus Fischer

Abstract

The synthesis of realistic human motion with large variations and different styles has a growing interest in simulation applications such as the game industry, psychological experiments, and ergonomic analysis. The statistical generative models are used by motion controllers in our motion synthesis framework to create new animations for different scenarios. Data-driven motion synthesis approaches are powerful tools for producing high-fidelity character animations. With the development of motion capture technologies, more and more motion data are publicly available now. However, how to efficiently reuse a large amount of motion data to create new motions for arbitrary scenarios poses challenges, especially for unsupervised motion synthesis.

This thesis presents a series of works that analyze and model the variations of human motion data. The goal is to learn statistical generative models to create any number of new human animations with rich variations and styles. The work of the thesis will be presented in three main chapters. We first explore how variation is represented in motion data. Learning a compact latent space that can expressively contain motion variation is essential for modeling motion data. We propose a novel motion latent space learning approach that can intrinsically tackle the spatial-temporal properties of motion data. Secondly, we present our Morphable Graph framework for human motion modeling and synthesis for assembly workshop scenarios. A series of studies have been conducted to apply statistical motion modeling and synthesis approaches for complex assembly workshop use cases. Learning the distribution of motion data can provide a compact representation of motion variations and convert motion synthesis tasks to optimization problems. Finally, we show how the style variations of human activities can be modeled with a limited number of examples. Natural human movements display a rich repertoire of styles and personalities. However, it is difficult to get enough examples for data-driven approaches. We propose a conditional variational autoencoder (CVAE) to combine large variations in the neutral motion database and style information from a limited number of examples.

Zusammenfassung

Die Synthese realistischer menschlicher Bewegungen mit großen Variationen und unterschiedlichen Stilen ist für Simulationsanwendungen wie die Spieleindustrie, psychologische Experimente und ergonomische Analysen von wachsendem Interesse. Datengetriebene Bewegungssyntheseansätze sind leistungsstarke Werkzeuge für die Erstellung realitätsgetreuer Charakteranimationen. Mit der Entwicklung von Motion-Capture-Technologien sind nun immer mehr Motion-Daten öffentlich verfügbar. Die effiziente Wiederverwendung einer großen Menge von Motion-Daten zur Erstellung neuer Bewegungen für beliebige Szenarien stellt jedoch eine Herausforderung dar, insbesondere für die unüberwachte Bewegungssynthesemethoden. Das Lernen der Verteilung von Motion-Daten kann eine kompakte Repräsentation von Bewegungsvariationen liefern und Bewegungssyntheseaufgaben in Optimierungsprobleme umwandeln.

In dieser Dissertation werden eine Reihe von Arbeiten vorgestellt, die die Variationen menschlicher Bewegungsdaten analysieren und modellieren. Das Ziel ist es, statistische generative Modelle zu erlernen, um eine beliebige Anzahl neuer menschlicher Animationen mit reichen Variationen und Stilen zu erstellen. In unserem Bewegungssynthese-Framework werden die statistischen generativen Modelle von Bewegungscontrollern verwendet, um neue Animationen für verschiedene Szenarien zu erstellen. Die Arbeit in dieser Dissertation wird in drei Hauptkapiteln vorgestellt. Wir untersuchen zunächst, wie Variation in Bewegungsdaten dargestellt wird. Das Erlernen eines kompakten latenten Raums, der Bewegungsvariationen ausdrucksvoll enthalten kann, ist für die Modellierung von Bewegungsdaten unerlässlich. Wir schlagen einen neuartigen Ansatz zum Lernen des latenten Bewegungsraums vor, der die räumlich-zeitlichen Eigenschaften von Bewegungsdaten intrinsisch angehen kann. Zweitens stellen wir unser Morphable Graph Framework für die menschliche Bewegungsmodellierung und -synthese für Montage-Workshop-Szenarien vor. Es wurde eine Reihe von Studien durchgeführt, um statistische Bewegungsmodellierungs und syntheseansätze für komplexe Anwendungsfälle in Montagewerkstätten anzuwenden. Schließlich zeigen wir anhand einer begrenzten Anzahl von Beispielen, wie die Stilvariationen menschlicher Aktivitäten modelliert werden können. Natürliche menschliche Bewegungen weisen ein reiches Repertoire an Stilen und Persönlichkeiten auf. Es ist jedoch schwierig, genügend Beispiele für datengetriebene Ansätze zu erhalten. Wir schlagen einen Conditional Variational Autoencoder (CVAE) vor, um große Variationen in der neutralen Bewegungsdatenbank und Stilinformationen aus einer begrenzten Anzahl von Beispielen zu kombinieren. Wir zeigen, dass unser Ansatz eine beliebige Anzahl von natürlich aussehenden Variationen menschlicher Bewegungen mit einem ähnlichen Stil wie das Ziel erzeugen kann.

Contents

List of Figures

Chapter 1

Introduction

1.1 Overview

Digital simulation is widely used in production planning and verification. It can efficiently reduce the cost and time compared to using a physical prototype. One of the key components in many digital simulation applications is a Digital Human Model (DHM). For the tasks of visualization and simulation, it is important to generate convincing and natural animations for virtual characters. Today, the growing demand for individualized products leads to a continuously increasing number of product variants, which requires digital simulation tools to be agile and adaptive. It also poses challenges to the DHM to be able to produce large variations and different styles in order to satisfy different scenarios. Additionally, in the early stage of character animation, the motions are usually robotic. This can not satisfy the demands for realistic, human-like animations nowadays. High-fidelity and natural-looking motions can give the designers and audience more immersive feelings and provide more accurate results for tasks like ergonomics analysis.

Although lots of significant progress have been made in the last decade, it is still a challenging task to create high-quality and natural-looking virtual human motions. In general, human motion is very complex because of the high degrees of freedom of the body. A natural movement could have any number of variations and randomness. However, the motion data also has a strong intrinsic correlation in both spatial and temporal domains. Each body joint is conditionally correlated to the movement. In addition, our movement is the most familiar thing we observe every day. Therefore, any incorrect or unnatural motions will be easily detected by human observers.

In the early stage, procedural motion synthesis approaches have been successfully used in films and games. It requires manually designing all the keyframes before generating a continuous motion by interpolation. Although procedural motion synthesis approaches can produce controllable motions with good quality, it requires considerable effort from animators to create natural motions with rich variations compared to real motions. This process does not scale well to large and rapidly changing tasks.

An alternative to animating virtual characters manually is the data-driven method. Data-driven motion synthesis approaches have demonstrated their power to create high-fidelity and natural motions and numerous successes have been achieved. With the fast progress of motion capture techniques, a large amount of high-quality motion capture data becomes easily accessible and affordable. It has been a very active research field to apply machine learning technologies to motion data.

One of the main challenges in data-driven motion synthesis approaches is how to efficiently reuse recorded motion data for different scenarios [MC12]. Usually, we are not just interested in creating one animation, but creating as many as we want. For example, in a digital simulation of assembly workshops, the animations of virtual workers include not only locomotions from one place to the other but also hand operations with objects, tools, and so on. Any changes in the sizes of objects, and types of tools will require an update of the animation. It is not feasible to capture all the possible instances.

Previous data-driven approaches, e.g.: Motion Graph [KGP02], decompose the record motion data into small pieces and search for the best transition points to reassemble motions. They use graph models to represent motion capture data and convert the motion synthesis problems to a graph search problem. For periodic motions like walking, they can produce new motions with arbitrary lengths. But they cannot generate any new variations that are not contained in the record database. Motion editing based on example poses can address this issue, but similar to procedural approaches, it could require considerable effort to produce a long motion [WCW14].

In order to learn a continuous motion space that can efficiently generate any number of new variations given the example motions, the captured motion sequences can be modeled by generative statistical models. Generative statistical models provide a compact, continuous representation of motion capture data. It can also provide a general framework for motion synthesis tasks, converting the search problems into optimization problems.

Another important property of human motion data is that people's personality, age, gender, and mood will influence how they behave. It is commonplace that people can distinguish each other simply by observing his or her body language, which is usually called personal style [Wes05; KSR15]. If we assume that human motion in general follows some kinds of distribution, these are conditional distributions which can be modeled as well. Integrating styles into virtual character animation can give observers a more realistic feeling. And it can also help to represent and detect the characters' mood and intention with style variations. Moreover, the current trends in virtual reality indicate that the demand for stylized animation will grow in the future.

For crowd simulation, typically a lower fidelity motion could be acceptable. However, for 3D cartoon and video games, virtual characters acting with their own styles and emotions could make the audience feel more immersive and engaging. Realistic motions with a rich repertoire of styles can help to detect the characters' intention and mood. For instance, in pedestrian simulation for autonomous driving testing, it will improve safety if the car can predict the pedtrians' intention correctly.

In general, it is not feasible to manually create keyframes for each style or capture large amounts of motions for every variation of every style. Motion style transfer provides the possibility of creating a synthetic stylized motion database from existing capturing without additional motion capturing efforts. Furthermore, some special styles are hard to capture. For instance, a drunk walking model would be interesting for pedestrian simulation of autonomous driving. However, capturing a lot of drunk people walking is much harder than normal walking. How to separate content and style for both image and motion data is still an active research area. It is appealing to reuse the motion content while transferring its style to another domain.

In order to address aforementioned challenges, this thesis presents a systematic study of generative statistical motion modeling. Different motion data representations for statistical modeling are investigated. A novel dimension reduction method which addresses the spatial and temporal properties of human motion data is proposed. Generative statistical models are trained and integrated into our graph-based motion synthesis framework. The framework is evaluated in the simulation of the automotive assembly workshop. Moreover, we combine style transfer with generative models and propose style-constrained generative statistical models to include style variations into our motion synthesis framework.

1.2 Research Objectives

The goal of this thesis is to provide a comprehensive study of human motion variation analysis and modeling. A recurring problem of character animation is how to create realistic, natural-looking human motion. Most of the state-of-the-art work [HKS17; Zha+18; Sta+19; Sta+20] focuses on producing high-quality and controllable animations. However, the generated motions are usually deterministic for the given input. Besides, natural human motion normally consists of large variations, which means that motions can be diverse given the same control parameters. Learning a controllable, generative, and probabilistic model for human motion is a nontrivial task and less explored.

The thesis will focus on two main topics: (1) How to efficiently represent the variation of motion data in the latent space for statistical modeling; (2) learning controllable, generative and probabilistic models for human motion variation and style modeling.

1.3 Contributions

The main contributions of this work are two parts: (1) Providing a novel motion data representation for human motion modeling and synthesis. A motion synthesis framework named Morphable Graph is constructed on top of the learned representation space. Our model can support different types of users' constraints in a general optimization framework. In addition, we accelerate the motion synthesis by introducing a space partitioning data structure for efficient searching. (2) Modeling variation and style for locomotion. In order to generate convincing pedestrian simulation, we study the problem of including variation and style in locomotion synthesis. A general latent motion space is learned using an autoencoder. We introduce a conditional statistical modeling approach for style motion modeling, which can enrich the motion database with stylistic motions given a few style examples and a neutral motion database.

1.3.1 Motion Data Representation Learning

Challenges. For data-driven motion synthesis, the choice of motion data representation impacts the effectiveness of modeling the variability presented in the data. As a common step of data-driven motion modeling and synthesis, an expressive latent space is first constructed for motion data. For statistical distribution learning, the dimensionality of latent space is usually much smaller than the original motion space due to the "curse of dimensionality" [Bel66]. So the representation learning can also be treated as dimension reduction of the motion data.

For the human observers, the variation of human motion refers to changing joint positions in Euclidean space (observation space). However, there are different ways to represent motion data as feature vectors (see Section 3). Different features could have different influences on joint positions in observation space. For example, in rotational representation, each feature is the relative rotation angle to its parent joint in the human body kinematic chain. A small rotation in the root joint could lead to a big movement of hands in Euclidean space.

Most dimension reduction approaches focus on reducing the reconstruction loss between input motion features and reconstructed features. All the features are usually treated equally and the ones with low variance could be discarded. The learned latent spaces do not take the spatial and temporal properties of motion data into consideration. And the statistical model built on the latent space might not efficiently capture the real variation of motion data corresponding to human observation.

Contributions. We propose a novel framework to learn a compact, expressive latent space for motion modeling and synthesis by leveraging the spatial and temporal properties of motion data. We apply functional data analysis to address the temporal consistency of motion data and automatically rescale motion features by optimizing the weights of each dimension in observation space. Our approach does not require any specifications of motion parameterization and can automatically adapt feature weights to different parameterizations. We evaluate the idea by constructing a new variant of widely used dimension reduction method Principal Component Analysis (PCA) for motion data, namely Scaled Functional Principal Component Analysis (SFPCA), and demonstrate that our method can retain the desirable variance of the motion data in observation space better compared to standard PCA, which is important for the visualization quality of the reconstructed motion. Additionally, our idea is not limited to PCA, it can be easily extended to other dimension reduction approaches. The details of the work can be found in the previous publications [Du+16a; Du+16b].

Furthermore, a statistical motion synthesis framework named Morphable Graph is constructed on top of learned motion representation space. Based on previous successful work Motion Graphs++ [MC12], we use a directed graph to model the high-level variations of motion. Each node in the graph represents the distribution of a group of structurally and semantically similar motions named Motion Primitive. The motions in each motion primitive are projected into a low dimensional space using SFPCA and the distribution is modeled using a Gaussian Mixture Model (GMM). Our framework provides a compact representation for the large, heterogeneous motion database. The motion synthesis tasks are converted to optimization problems. In order to accelerate the speed of optimization, we propose a new space partitioning data structure for statistical models. Our system has been evaluated in digital workshop simulation and demonstrates that it can generate different kinds of actions for different types of control signals. The content of this work corresponds to the joint publications [Her+17; Man+18]. My main contribution to this joint work is learning generative model models for fast motion synthesis and optimization.

1.3.2 Modeling Variation and Style for Locomotion

Challenges. Variation and style are critical components for a convincing natural human locomotion, especially for pedestrian simulation. Most of all when simulating a group of characters, the resulting motions will appear robotic and not natural anymore if all avatars are simulated with the same walk cycle. While many previous research work focuses on high-quality, interactive motion synthesis, the results are usually deterministic and do not include rich variations and styles in the generated motions.

It is a challenging task for character animation to achieve precise control while preserving natural variation. The large variation of human motion could cause big ambiguity for controlled motion synthesis. The control signals from users' input is usually high-level, for instance, the trajectories to follow for locomotion and the target positions for end-effectors to reach for punching. There could be many possible motions with different speeds, step sizes, and styles to meet the same constraints. To improve the motion quality, many state-of-the-art motion synthesis approaches [HSK16; HKS17; Sta+19] include additional information to reduce the ambiguity. While these approaches produce high-quality motions, the results are deterministic. It is still an active research topic to find a good balance between responsive controllability and diversity.

Another important aspect of human motion variation is style. For many applications like crowd simulation, it is desired that each group of characters should have distinctive styles. Data-driven motion synthesis methods usually require a large amount of motion capture data for training. It is not feasible to capture large amounts of data for all possible styles. Hence, how to reuse an existing motion database to create stylistic generative models with a few style examples pose challenges.

Contributions. In order to address the aforementioned challenges, we apply Variational Autoencoder (VAE) to create generative models for motion synthesis and controlling. We extend our graph-based generative motion synthesis model using VAE. A general low-dimensional motion space is learned for all actions of interest using autoencoder and the distribution is modeled using VAE for each motion primitive. Our framework can also be used for stylistic motion modeling. We propose the first work to implicitly combine style transfer and statistical motion modeling by using VAE to formulate the style-conditioned distribution of human motion. The stylistic generative model is formulated as a conditional distribution. We take a single style motion as the condition to transfer the distribution of neutral motions to match the target style. Any number of new stylistic motions with rich variations can be generated given a single or a few style examples. The content of the work corresponds to the previous publications [Du+19b; Du+19a].

1.3.3 Related Publications

[1] **Han Du**, Martin Manns, Erik Herrmann, and Klaus Fischer. Joint angle data representation for data driven human motion synthesis. Procedia CIRP, 41:746–751, 2016.

[2] **Han Du**, Somayeh Hosseini, Martin Manns, Erik Herrmann, and Klaus Fischer. Scaled functional principal component analysis for human motion synthesis. In Proceedings of the 9th International Conference on Motion in Games, pages 139–144. ACM, 2016.

[3] Erik Herrmann, Martin Manns, **Han Du**, Somayeh Hosseini, and Klaus Fischer. Accelerating statistical human motion synthesis using space partitioning data structures. Computer Animation and Virtual Worlds, 28(3-4):e1780, 2017.

[4] Martin Manns, Klaus Fischer, **Han Du**, Philip Slusallek, and Kosmas Alexopoulos. A new approach to plan manual assembly. International Journal of Computer Integrated Manufacturing, 31(9):907–920, 2018.

[5] **Han Du**, Erik Herrmann, Janis Sprenger, Klaus Fischer, and Philipp Slusallek. Stylistic locomotion modeling and synthesis using variational generative models. In Motion, Interaction and Games, 2019.

[6] **Han Du**, Erik Herrmann, Janis Sprenger, Noshaba Cheema, Somayeh Hosseini, Klaus Fischer, and Philipp Slusallek. Stylistic locomotion modeling with conditional variational autoencoder. In Eurographics (Short Papers), pages 9–12, 2019.

1.4 Outline of Thesis

The thesis is structured as follows: Chapter 2 gives a literature review on the related works to the research presented in this thesis. Chapter 3 presents our data-driven motion modeling and synthesis framework named Morphable Graph. We apply functional representation to address temporal smoothness and scale the influence of joints based on skeleton hierarchical structure. A novel scaled function principal component analysis (SFPCA) is proposed to learn a low-dimensional space for motion representation. Charter 4 is about modeling the variations of motion capture data using generative statistical models. We construct generative statistical models for multiple actions based on low-dimensional hidden space learned by SFPCA. The models are integrated into our graph-based motion synthesis framework. In addition, a novel space partition clustering approach is proposed to accelerate the sampling-based motion synthesis. Our system is deployed in the simulation of an automotive assembly workshop. The variations and performance of generated motions have been evaluated. Chapter 5 explores the usage of the deep generative models in motion synthesis. We present a solution to combine the large variation in the neutral motion database and style information from a limited number of examples. Style transfer is implicitly applied during the model learning process. Conditional variational autoencoder (CVAE) is applied to learn the distribution, stylistic examples are used as constraints. We demonstrate that our approach can generate any number of natural-looking human motions with a similar style to the target.

Chapter 2

Related Work

2.1 Overview

In this section, we present a comprehensive review in the field of modeling skeleton-based human motion variation and style. The goal is to give the reader an overview of the progress of modeling motion variations in motion synthesis methods. We will also present some concepts relevant to the rest of the thesis.

2.2 Procedural Character Animation

Procedural methods control a simplified model of the body based on inverse kinematics or inverse dynamics using different form of controllers (see [GP12]): Proportional differential controllers for individual joints [YLP07; CBP10]), optimization-based approaches that solve for forces for all joints at once (e.g. [LMH10; ADH13]), or recent global sampling-based methods (e.g. [Liu+10; Häm+14; LYG15]). However, procedural approaches usually require careful configurations of the controllers, and do not automatically generate natural motion.

2.2.1 Adding Noise as Variations

In the early stage of character animation, the keyframe based animations look usually repetitive and robotic. Creating high-quality keyframes with large variations requires a significant amount of efforts and skills. Therefore, a central problem of generating a natural-looking character animation is how to automatically and systematically include variations of real human motions in a visual and physical correct way.

Perlin [Per95; PG96] applied stochastic noise functions to simulate dynamics of natural motion. This is based on his previous successful work of creating natural appealing textures using Perlin noise. In order to make the motion lifelike, an animator must tune the parameters of noise functions. Bodenheimer et al. [BSH99] added noise to cyclic running motion. The noise is solely introduced to the upper body and is synchronized with the arm swings in the running cycle. Nonetheless, there is no guarantee that added noise will align seamlessly with the existing motion. Incorporating random noise may result in unwanted distortions in the motion, and the use of noise with tailored distributions entails a manual parameter tuning process that involves trial and error.

Recently, in the state-of-the-art deep learning based motion synthesis approach, Starke et al. [Sta+20] added Gaussian noise to latent space to increase the variation of the generated motion. The variations are ensured in the valid range by the powerful decoder. However, it does not represent natural variations since it is not learned from the motion data. There is no guarantee that the generated and added noises will approach well with the true motion variations. Usually, the noise functions are hand-craft and require prior knowledge of the target motion. It is not robust for general motion synthesis frameworks.

In addition, biomechanical researches [HW98; Gol+02; RT02] have recognized that intrinsic variability is not just noise or error, but is a functional component of motion. From this point of view, adding random noise to existing motion is not a principled approach.

2.2.2 Model Variations using Parametric Functions

Another popular way to create new variations is to modify the existing animations such as hand-crafted animations or motion capture data. Motion blending, also known as motion interpolation, has been extensively used in the automatic character animation for design tools and video games since it remains relatively simple and produces good quality results in many cases. These procedures are usually described by some mathematical functions. Parametric functions, either linear or nonlinear, are commonly used to perform blending or interpolating example motions to generate new variations. In Interpolation synthesis for articulated figure motion, Wiley et al. [WH97] utilized linear interpolation of sample motions to generate novel motions that exhibit specific characteristics. For instance, they employed this technique to interpolate reaching motions and create a motion where the character reaches a different target location or writes on a whiteboard. The technique for motion generation was restricted to linear interpolation of the exemplar motions, thereby rendering it unsuitable for generating motions that precisely met the requirements in situations where non-linear interpolation was necessary.

Interpolation-based methods [Kov04b; HG07b; FXS12] create new variations by interpolating between temporally aligned structurally similar example motions. Motions are placed into an arbitrary user-defined parameter space by finding a mapping function from parameter to interpolation weights. Furthermore, there are approaches combining concatenation and interpolation, such as Motion Fields [Lee+10].

2.3 Classical Statistical Modeling Methods

In this section, we cover the major probabilistic models used for modeling motion data before "deep learning revolution". In general, any probability density estimation approaches can be used for modeling the variation in motion data. Since motion data displays a rich repertoire of spatial and temporal variations, the focus of statistical modeling can be categorized as motion dynamic modeling and motion density modeling. Dynamic modeling regards motion data as time series data and models the conditional probability between poses. Motion density modeling assume the motion data represented in latent space is generated independently and tries to learn the probability density function. Additionally, how to generate new variations using different statistical models are discussed.

2.3.1 Dynamic Modeling

Linear Dynamical System

Human motion data has intrinsic consistence in time domain. Hence, it can be treated as continuous temporal signal. Linear Dynamical System (LDS) is parametric model that can model sequential data linearly. Since it is simple and can be solved exactly, LDS is commonly used to model time-series data, including human motion data [Bre97; CH07]. Chai and Hodgins [CH07] presented a statistical dynamic model that can generate motions to meet a variety of user-defined constraints. They created a reduced subspace using PCA and learn the dynamic variations using an m-order LDS. Although LDS is effective and easy to learn, it has been proven that it is not suitable for modeling motion data due to the nonlinearity and complexity of human motion [Bis05].

Instead of using LDS on the whole sequence of motion, Chai and Hodgins [CH05] proposed an approach to automatically learn a series of local models from motion capture examples. Their model can handle nonlinearity since they constructed local models dynamically. The dynamics of each local model are captured by LDS. The local model was dynamically learned by searching the closest examples to the control signals. Local LDS was applied on the low-dimensional manifold space learned by PCA.

Nonlinear Dynamical Systems

In order to overcome the limitation of LDS, an intuitive way is to use nonlinear functions to model the data. A nonlinear extension of LDS named Switching Linear Dynamical System (SLDS) has been proposed and studied. SLDS improves LDS by defining local state and switching states to introduce nonlinearity, Pavlovic and his colleague [PRM00] explored using SLDS to model human dynamic by casting it to a Dynamic Bayesian Network (DBN). Similar to DBN, the full motion is decomposed as a set of finite discrete states. Each state is modeled by LDS. So the nonlinearity of motion data can be achieved by switching among the linear dynamic models over time. Three different inference schemes were derived for switching dynamics. They demonstrated that SLDS models are a promising tool for motion analysis, such as walking and jogging.

A similar work is conduced by Li et al. [LWS02]. They introduced a two-level statistical model named motion texture for synthesizing dance motion. The long dance motions are decomposed into small clips named "motion texton". Local LDS is used to captured the local dynamics in each motion texton. For the low-level variation in each texton, it is modeled by Gaussian noise in local LDS. For the high-level variation between texton, they modeled the distribution as a first-order Markov

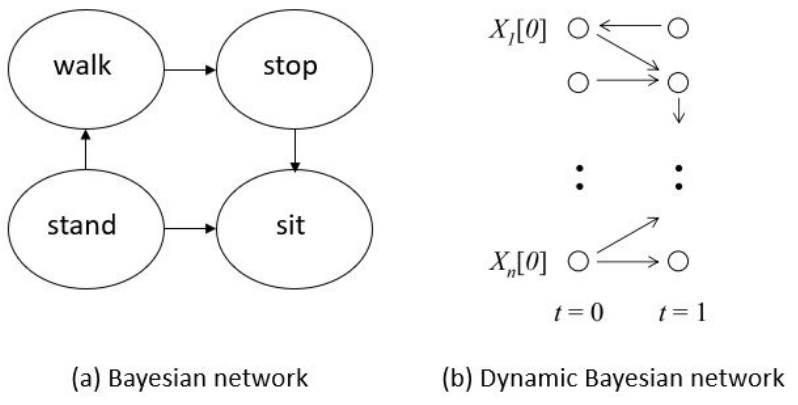


FIGURE 2.1: An illustration of Bayesian network and Dynamic Bayesian network. Image (b) from [LBK09]
.

chain. The synthesis of new motions was also done in two-step manner. A lowest cost path was found in texton state space and each texton generated local dynamics by sampling Gaussian noise. They used 20 minutes of dancing motion as training data. If a large amount of data is available, it is possible to use random resequencing of motion clips without being able to detect repetitions in the motion.

Although SLDS approaches are more expressive than LDS models and can model more complex motions, it is usually required to decide the appropriate number of the discrete switching states manually. This could highly depend on the variation in the input motion and need to be analyzed case by case. The transition probabilities between states are normally modeled by calculating the frequency of the transition happened in the data. Therefore, it requires a large amount of data to get an accurate estimation.

Dynamic Bayesian Network

A Bayesian Network is a probabilistic graphical model that can model the joint distribution over a set of random variables $\mathbf{X} = X_1, ..., X_n$. It uses a directed acyclic graph to explicitly model the conditional dependencies between variables or states. For constructing a Bayesian network, the discrete variables need to be defined as the nodes in the graph. Bayesian network is commonly used to define the high-level behaviors [YT07]. For example, it can effectively define the transitions between actions

Figure 2.1 (a), which represents the high-level variations of motion. However, for time-series data like motion, the state of each variable is also changed over time. It is better to use dynamic Bayesian network DBN to represent the temporal conditions. Figure 2.1 (b) gives an example of DBN for two poses at $t = 0$ and $t = 1$. Each degree of freedom of the pose is represented as a variable. The spatial and temporal dependencies can be modeled.

Lau et al. [LBK09] modeled the spatial and temporal variations together using a Dynamic Bayesian Network (DBN) from a small set of recorded motions. Each node in the graph represents the variation of a Degree of Freedom (DOF) of the motion at time t. A prior network is trained based on the distribution of values for the DOFs for the first two time steps. Then a transition network is trained to model the distribution of change from $t+1$ to $t+2$ dependent on t and $t+1$. This way motions of arbitrary lengths can be synthesized.

Hidden Markov Model

Hidden Markov Model (HMM) is a type of Dynamic Bayesian Network that is widely applied for modeling a distribution over a series of observations. They have been applied by Bowden [Bow00] to model human motion distributions. They use HMM to learn motion patterns from a large varied stylistic motion dataset. The learned distribution can synthesize novel motion data in any interpolation or extrapolation of styles. Lee et al. [Lee+02] applied clusters of Gaussian mixture models in an unsupervised way on top of the Markov decision process modeling the transitions in a structure. The approaches based on HMMs fail to model some complex nonlinear dynamics of human movements. The state size grows exponentially with the number of components that becomes intractable for complex motions. However for specific applications it is still useful. Zhao et al. [ZSJ20] applied a HMM and formulate the synthesis as an adversarial Bayesian inference problem.

Gaussian Process Dynamical Model

Motion data is temporally continuous and each DOF can be treated as a function over time. It has variations in both spatial and temporal domains. Hence, the variability of motion can be described as a random process. Gaussian Process (GP) and its variants have been applied to model the dynamics of motion data. GP is defined as a probability distribution over functions $y(t)$ such that the set of values of $y(t)$ evaluated at an arbitrary set of points $(t_1, ..., t_n)$ jointly have a Gaussian distribution. Compared to normal regression models, GP regression models can predict a distribution at each point of time instead of a number. Wang et al. [WFH08] addressed the dynamics of human motion by introducing a Gaussian Process Dynamical Model (GPDM) to model the temporal sequence of human motion. A smooth low-dimensional space is found for motion data by taking dynamics as the constraint. GPDM provides a general framework to model the motion dynamics and it can work well with a small database. Conversely, GPDM doesn't scale well for a large amount of data. Therefore, it cannot take advantage of including more natural variations from new examples.

2.3.2 Density Estimation

Another task for modeling motion variation is to learn the distribution for a collection of poses or a set of motion clips. The samples are assumed to be generated independently. Unlike learning the motion dynamics, which are usually formulated as a conditional distribution $P(X_t|X_{t-1})$, we are interested in learning the distribution of the observed samples $P(\mathbf{X})$ is assumed to be independent and identically distributed. Non-parametric density estimation (e.g. histogram, kernel density estimation, etc.) and parametric models (e.g. Gaussian, Mixture of Gaussian, etc.) can be used for modeling the distribution of motion samples. The learned statistical models can provide a compact representation of the motion data. There is no need to store original motion data and any number of new samples following the same distribution can be easily generated.

Kernel Density Estimation

Pullen and Bregler [PB00; PB02] used kernel-based density estimation to learn the joint probability of features in motion data. The features are represented at different frequency bands. The learned distribution can be used to synthesize new motions based on the statistical properties of example motions. The correlations between the degree-of-freedoms are explicitly defined. For example, they specify manually that the hip angle affects the knee, and the knee angle affects the ankle. They demonstrated their method by animating a 2-dimensional 5-DOF wallaby figure. However, it is hard to predict how well the method would work on a more complex 3D character. Furthermore, non-parametric density estimation cannot provide a compact representation for motion capture data. All the original data need to be stored in order to calculate the density. Therefore, it is not feasible for a big dataset.

Mixture of Gaussians

The normal distribution is a very important statistical distribution pattern occurring in many natural phenomena, so normal distribution could also be a good statistical model for human motion data. However, is a uni-model enough to capture the distribution of motion data? This question is not easy to answer by directly observing the data. Because it is not a simple task to visualize the motion data, even in low dimensional space. Therefore, GMM are widely used to model the distributions in natural language processing, computer vision, and graphics.

Min et al. [MCC09] decomposed motion variations into spatial and temporal variations. For spatial variations, they applied PCA to reduce the high dimensionality of original geometric motion values. For temporal variations, they used dynamic time warping to register all motion clips to a canonical timeline. So the temporal variation of each motion clip is represented by the time warping function. They also project the time warping functions into low-dimensional space while keeping the monotonic increasing property. Each motion clip is finally represented as a weighted concatenated vector of spatial and temporal parameters in low-dimensional space. The distribution of the motions is modeled by Gaussian Mixture Model. The learned distribution can be used for interactive human motion synthesis by formulating the users' specified constraints in a maximum posterior framework. However, their approach requires the motions need to be structurally similar and carefully aligned by manually annotating keyframes. This limits the ability of their approach to deal with complex motions and it could generate poor results if the constraints are out of the range in the training data.

Min and Chai [MC12] extended their previous approach [MCC09] to handle complex motions by introducing a directed graph to represent the high-level structures of human motion. They assumed that even though natural human motions display a rich repertoire of variations, the high-level structures are always finite. Their model named Morphable Graph employs a directed graph to represent an entire action. Each node in the graph is one of the predefined high-level structures of the action, which is called Motion Primitive, and each edge represents the possible transition between nodes. A semi-automatic preprocessing pipeline is required to decouple the long training samples into predefined motion primitives. The motion clips in each motion primitive are semantically and structurally similar. Each motion primitive is modeled as a statistical model using GMM, similar to [MCC09]. The transitions between motion primitives are modeled by GP. The generation of new motion will start with a graph walk in the morphable graph, and optimize the sample in each motion primitive to meet the environmental and users' constraints. Their work can not only model the low-level variations contained in each motion primitive but the high-level variations such as walking 2 steps left or walking 3 steps right as well.

Most statistical motion models formulate the conditioned motion synthesis task as an optimization problem. This might be not efficient for real-time animation systems. Herrmann et al. [Her+17] proposed an acceleration framework for statistical motion synthesis. They applied GMMs to learn the distribution of structurally and semantically similar motion clips named "Motion Primitive". For the synthesis stage, they pre-sample a large number of random samples from the learned statistical models and construct a space partitioning tree to assist a fast search for generated constrained motions.

2.4 Deep Generative Models

The surge of computational power in the last decade caused a vast increase in research focusing on neural networks and deep learning. Generative models based on deep neural networks have demonstrated outstanding performance in different domains such as images and natural language but also in human motion. This section gives an overview of different approaches for generative models for human motion data excluding deterministic approaches such as [HKS17].

While these forward interpolating neural networks (FINN) are providing a natural motion synthesis that can be controlled responsively, they are deterministic by nature and unable to create variations in the generated motion. Although the animation of a single avatar appears convincing, the animation of multiple avatars appears highly unnatural and robotic. Therefore, including natural variations as a statistical model is attractive for crowd animation and realistic simulation.

Reactive, real-time motion synthesis can be considered as a concatenation of subsequent frames. This process can be regarded as regression and thus enables the utilization of neural networks. The regression model predicts the next frame based on the previous frame(s). In order to control the motion synthesis, user input can be incorporated between the regression steps via a separate motion controller (e.g. [Cla16; LLL18]). In order to model time-series data, recurrent neural networks (RNNs) are considered ideal, as they incorporate the history of generated frames in a latent representation. Many different approaches have been already published [Fra+15; Li+17; LLL18]. However, the previous work shows that training RNNs is a very complex, time-consuming, and uncontrolled generation of motion using RNNs suffers from dying out of motion and an accumulation of error [Fra+15; Li+17; HKS17]. Using simple regression models suffers from pose averaging. Similar intermediate poses in different sequences, e.g. when both legs are crossing each other in a walk cycle, are averaged and prevent a natural transition of frames. Due to missing information from the recent past, the network can become unable to distinguish the foot going forward and does not accurately continue the walk cycle. Recently, [HKS17] proposed a novel network architecture that interpolates the network weights depending on the phase of the walk cycle. This methodology was further extended by [Zha+18] to generate complex quadruplet motion of dogs. A proof of concept of the same idea for reach motions was recently published by [Gai+18]. All of these approaches have in common, that the network weights of the regression network are interpolated and use a similar network structure for motion synthesis.

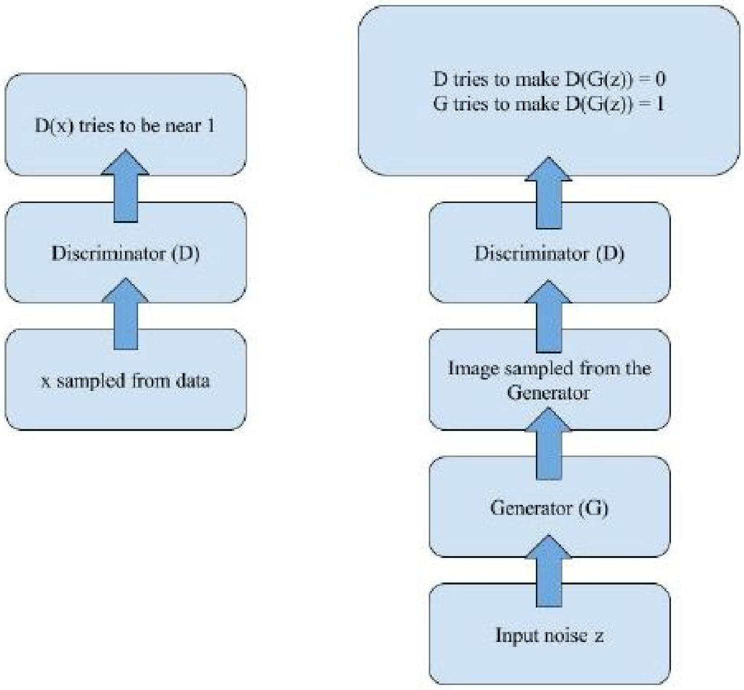


FIGURE 2.2: The standard working pipeline of vanilla GAN.

2.4.1 Restricted Boltzmann Machines

Restricted Boltzmann Machine (RBM) are shallow neural networks that consist of hidden and visible parameters and are usually used to model static data. RBMs can be stacked to form deep belief networks. Taylor et al. [THR11] trained a model based on RBMs for gait animations. The authors introduce an extension called conditional RBM to model time series data by treating the visible parameters of the previous steps as additional inputs. To learn different styles of motions, they stack CRBMs to form conditional deep belief networks.

2.4.2 Generative Adversarial Networks

GANs [Goo+14; Goo+20] have shown their powerful generation capability in computer vision areas. In general, GAN consists of two components, one is called the generator G that takes random vector z from a simple random distribution, such as the normal distribution or uniform distribution, as input and generates fake data. The other component is called discriminator D that is trained as a classifier to discriminate between fake and real data. G and D can freely choose as convolutional neural network (CNN), recurrent neural network (RNN) or multilayer perceptron (MLP). It works like a 2-player game. For the generator, its objective is to generate real-like data, which should be like the real data such that the discriminator cannot distinguish. In contrast, the training objective of the discriminator is to learn how to successfully classify fake generated data from real training data. During training, these two components should be well-matched in strength so that when one becomes stronger the other should become stronger as well. In the end, for an ideal case, the generator can generate samples real enough such that the discriminator cannot tell real or fake. Figure 2.2 shows this adversarial process of generator and discriminator. The ideal convergence is that the distribution of samples from the generator gets closer to real data distribution during training. In the end, the generated data's distribution overlapped with the real distribution. The data generated by G is so real that the discriminator cannot discriminate between generated data from real data.

GAN and its variants have been widely used in the task of motion sequence modeling [BKL18; KGB19; WCX19; Har+20; Wan+20] Barsoum et al. [BKL18] proposed a GAN-based 3D human motion prediction framework named HP-GAN. To our best knowledge, it is the first work that applies GAN to the task of human dynamics modeling. For the purpose of motion sequence modeling, the generator and discriminator of HP-GAN are both Recurrent Neural Network (RNN). HP-GAN uses critic loss as an additional constraint to help generate more real-like data when considering the fact that mean square error only may lead to blur results. In addition, it can model the sequential distribution and produce a probability as output. Two discriminators are used in HP-GAN, one is used for the training generator, and the other is used for measuring output quality. They evaluated their model on H36M database. Even though the predicted poses show good temporal variations, there is a noticeable discontinuity between input pose sequences and predicted pose sequences. In addition, their results seem to be converged to mean pose in long-term prediction.

Kundu et al. [KGB19] improved the previous work [BKL18] by introducing a bidirectional framework. They improve the quality of predicted motions and avoid model collapse by introducing a direct content loss in the training. The bidirectional framework can provide a cyclic reconstruction loss. They evaluated their approach on both the CMU [Dat18] and H36M [Ion+14] databases. They demonstrate that their model can achieve a better visual quality than HP-GAN [BKL18].

2.4.3 Variational Autoencoder

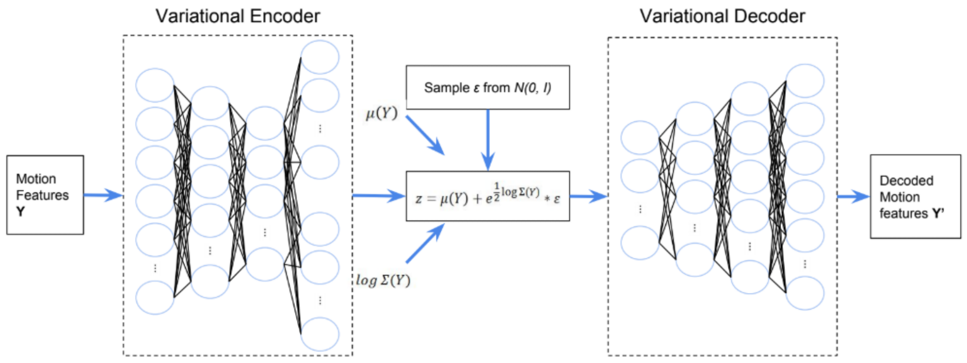


FIGURE 2.3: The architecture of a vanilla variational autoencoder.

Variational autoencoders (VAEs) [KW13; RMW14] have been proposed as unsupervised, non-parametric approaches, which are capable of modeling complex distributions. Since a VAE is a non-parametric model, there is no prior assumption about the distribution of the data. Some classical generative models, such as GMMs, usually assume that the motion data follows some specific distribution, for instance, normal distribution or a mixture of Gaussian. This assumption cannot be easily verified and may only work for a limited set of motions. It uses the computational power of neural networks to learn a transformation from a normal distribution to any kind of complex distribution. The network structure of VAE is very similar to a standard autoencoder. But they are fundamentally different. The goal of an autoencoder is to learn a manifold representation of the original data with an encoder and a decoder. It can be used as a powerful tool for dimension reduction or representation learning. VAE approximates the true distribution of a variable **Y** by optimizing the variational lower bound of the true distribution. A detailed derivation can be found in [Doe16].

Motegi et al. [MHM18] applied a variational autoencoder to model this distribution of the CMU database. The motion data is decomposed into fixed-length motion clips by using a sliding window. They used a single-layer convolutional autoencoder to create a 32 dimensional latent space. A three-layer VAE is trained on top of the latent space. They demonstrated that the low-dimensional latent space learned from the large motion dataset can well represent the motion data and generate natural-looking samples with new variations.

As a common request for motion synthesis, it is desirable to be able to generate a variable length of motions. A large amount of work [Toy+17; Yan+18; Du+19b; Hab+17; Spr+20; Yan+19a; Ali+20] have been proposed to extend VAE to model motion sequences. One straightforward approach is to combine VAE with neural network-based sequential models, such as RNNs. Habibie et al. [Hab+17] built a recurrent neural network approach by combining a VAE with Long Short-term Memory (LSTM) nodes in the decoder. The process of motion generation is guided by random samples from the latent space and control variables that confine the character's trajectory and velocity. These control variables are encoded and used to condition the RNN. During inference, the RNN's cell state is initialized with the latent code value, and the concatenation of cell state outputs at each time step is sent to the decoder to generate the synthesized motion sequence. Their approach can produce up to 210 frames with only a very small error if provided a target path constraint. Variants of recurrent variational autoencoder have been used to model the dynamic of motion data.

Yan et al. [Yan+18] proposed a variant of VAE named MT-VAE that focused on modeling and generating multimodal Human dynamics. Similar to [Hab+17], MT-VAE combines LSTM with VAE to address motion dynamics. They assumed that a long-term motion consists of a series of motion modes. They applied LSTM encoder and decoder to encode motion modes as feature vectors. VAE is applied to model the sequential distribution $\mathbf{P}(future|past)$. Future and past are two successive motion modes that are encoded by LSTM encoder. A pair of future and past states are fed into VAE to learn the conditional transition distribution. For motion synthesis, VAE decoder generates the future mode given the past mode as input, and the LSTM decoder brings the future mode to the motion sequence.

Ling et al. [Lin+20] built character controllers using deep reinforcement learning (DRL) on top of motion VAEs. Based on previous work [Zha+18], they combined the mixture-of-experts (MoE) prediction network from [Zha+18] with a VAE. The original MoE prediction network has a gating network to control the weights of the prediction network. Ling et al. constructed their model as an autoregressive framework. The past pose and current pose are concatenated as input to the encoder network. The random sample generated from the encoder is fed into both the gating network and prediction network. The prediction network serves as a decoder. The generated pose is feedback as a past pose for the encoder. Their model can generate an unlimited length of motion sequence with sufficient variations. For the controller, the model learns control policies using DRL based on motion VAE. They demonstrate that their approach can control motions with large variation while still fulfilling the user constraints. However, the flexibility of controllers still highly relies on the variation in the training data. Additional, although the generated motions can be diverse from each other, it is not as responsive as recent deterministic motion synthesis methods [Sta+19; Zha+18; Sta+20].

Guo et al.[Guo+20] proposed an architecture using a conditional temporal VAE for a label-conditioned motion generation. In their architecture an acRNN models predicts the latent parameters of the current pose at t based on the latent parameters of the previous pose $t - 1$. The latent parameters are then projected into a pose representation using an VAE architecture. The authors chose the exponential map to represent rotation. Petrovich et al. [PBV21] proposed an improvement to this method by combining a VAE with a Transformer. In addition to an action label the distributions are conditioned to the body shape based on the SMPL model [Lop+15] and they employed a 6D continuous rotation representation introduced by [Zho+19].

Chapter 3

Motion Representation Learning

Motion representation learning is essential for modeling motion variations. Natural human motion displays a rich repertoire of variations. It is complex due to the high degree-of-freedom (DOF) of the human body. Human motion data is high-dimensional data in both spatial and temporal domains. However, each DOF is not completely independent. The body parts are intrinsically coordinated and well synchronized. The variation in motion data follows physical and kinematic constraints. Therefore, it is possible to represent motion data in a low-dimensional space with fewer, but more expressive features. This is also very important for learning the distribution of motion variations since successful modeling in high-dimensional space is much harder than in low-dimensional space.

The standard principle for learning motion representation is to keep as much variance as possible. However, the variance presented in the data depends on how motion data is parametrized. There are different ways to parameterize characters' movement, which are discussed in Section 3.2. Even though the motion variation is unique in observation space, they might be nonlinearly mapped to feature space. For instance, using relative joint angles based on the kinematic chain, a small variance of root joint in feature space could lead to a large variance of hands in observation space. The spatio-temporal properties of motion data are usually not considered in representation learning.

We evaluate different motion parametrization methods and propose a novel dimensional reduction approach named SFPCA for motion data. We address the spatio-temporal properties of motion data by leveraging functional data analysis to ensure temporal smoothness. In addition, we combine forward kinematics with functional PCA to derive an automatic scaling approach to rescale the features of motion data irrespective of which parameterization approach is used. The low-dimensional space constructed from our approach can better preserve the motion variations in observation space. Besides, our idea is simple and can be easily extended to other dimension reduction methods for motion data.

In general, the study of the variability of human movements is not limited to skeleton-based full-body motion. To this end, a significant amount of work has been done on studying human activities and body movements, including full-body motion [LWS02; LBK09; MC12; HKS17; Sta+20], facial expression [KHS03; Wei+11; Yan+19b], muscles [Alb+05], body meshes [Lop+15; OBB20] and so on. The scope of this thesis is focused on modeling and synthesis of skeleton-based full-body human motion. In this chapter, we first introduce the concept of human motion variation and the types of variations (in Section 3.1). Section 3.2 explores the different methods to parameterize motion data. For the task of statistical modeling, the choices of the data representation are critical for successful learning. We present our work for motion representation learning in Section 3.3.

3.1 Motion Variation

Variation in human motion has been observed as a natural phenomenon for a long time. Different people usually perform the same action differently. Even when performed by the same individual, movements tend to vary between repetitions. Studying the variability in movements has been a long-standing research interest for anthroposociologist and biomechanists [Woo99; Fit54; GBD08]. In the early stage of character animation, the variability of characters' movement is not considered or included due to the limitations of tools and motion data. Animators usually create keyframes to control the animation. Even though a skilled animator can create high-quality animations, the animation curves generated by interpolating keyframes

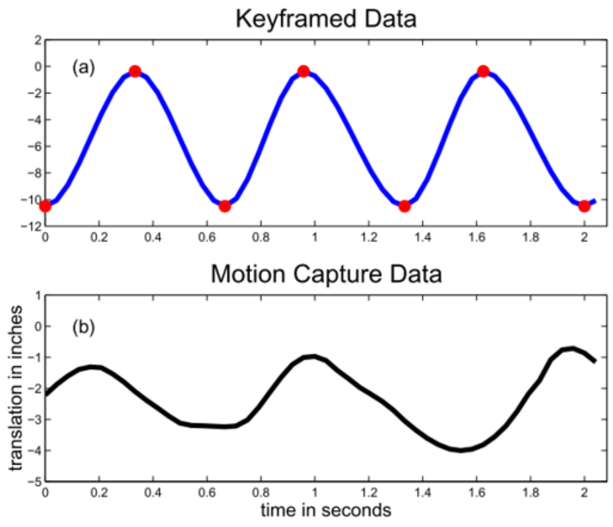


FIGURE 3.1: A comparison of keyframe simulated data and motion capture data for root joint y-axis translation for walking. Image from [PB02]
.

are still not the same as live motions. Figure 3.1 illustrates the difference between keyframed data and motion capture data. The keyframe simulation is very smooth and rhythmic. In contrast, the real motion shows irregularities and variations. The animator can introduce more variations by adding more keyframes, but the repetitiveness will still exist in the resulting motion and can appear unnatural. The variation is also observed in other natural phenomena such as textures, and these variations are often treated as noise. For realistic texture synthesis, it has been proven that adding pseudo-random noise [Per85] can significantly improve the realism of generated textures. Similar to texture synthesis, adding more subtle details and dynamics can improve visual realism. However, it is a nontrivial task to model and generates motion variation correctly. Human variation is more than just noise [HW98] and humans are more sensitive to unnatural variations of human motion than others such as image texture.

3.1.1 Spatial Variation

The spatial variation of human motion sometimes also refers to pose variation. For the same action or semantically similar pose, the real motion could display enormous variations as each person might do it differently. It is also not completely random since the motion still follows biomechanical and physical constraints. The

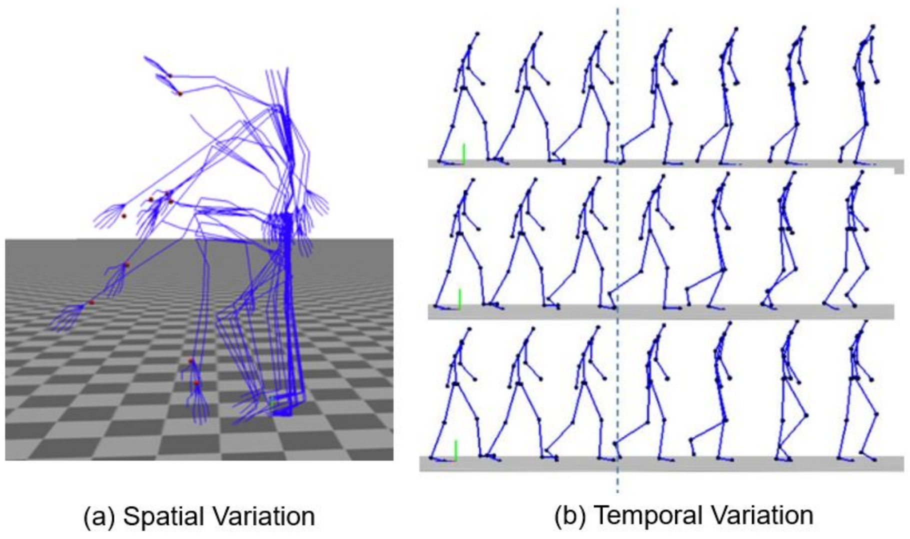


FIGURE 3.2: An illustration of spatial and temporal variation of human movement. (a): Picking at different target positions. (b): Given the same pose sequence as input, the future walking poses could be diverse along the timeline.

human body is highly correlated. Each body part is always well synchronized in order to keep a natural balance. The variation of each body part is different for different types of actions. For example, Figure 3.2 (a) shows multiple repetitions of picking an object at different positions. The upper body displays larger variations than the lower body., In contrast, for walking, even at the same step, the step size varies a lot for different steps. The variation of the upper body looks much smaller in this case. The correlation between each body part or each joint can be modeled as a conditional distribution [Zho+14]. In some early work [Osh08; KN12], the dependencies are manually specified based on the hierarchical structure of the human body. For instance, the ankle depends on the knee, and the knee movement is based on the hip. Another trend in modeling the pose variation of motion is to take the full-body pose as a single vector [MC12; MCC09; Du+16a]. They do not partition the human body into different groups and put any prior assumptions on the dependencies. They implicitly learn the spatial variation and the correlation between joints as a single statistical model.

Learning the spatial distribution of poses can generate new poses that are not in the captured data. Perlin and Bregler [PB00; PB02] combine the keyframe animation system with a statistical motion model from motion capture data. Their system allows the users to draw a sketch of the avatar. The detail is added by finding the best match in the learned distribution. Chai and Hodgins [CH05] propose an approach that can produce high-quality controllable animations from a low-dimensional control signal. The control signal is generated from a sparse subset of the full-body markers. They automatically fill the missing DOFs by using the prior knowledge from pre-recorded real motions. The pre-recorded motions are modeled as a statistical distribution.

3.1.2 Temporal Variation

Another important property of motion data is that it is a smooth, continuous signal in the time domain. It has variations over time. The temporal variation of motion is also referred to as a sequential variation or motion dynamics. Motion data can be represented as a sequence of poses in the time domain. Given the same input motion, the next pose is not necessarily deterministic. It also has variations in the timeline, which can be treated as a random process. The correlation between poses can be modeled as conditional distributions.

The dynamics of human motion sequences are mostly treated as a deterministic prediction problem. It is usually formulated as $X_{T+1:T'} = F(X_{1:T})$. Given the previous pose sequence $X_{1:T}$, the learned prediction model F produces the best future poses $X_{T+1:T'}$. However, natural human motion is a stochastic process. Figure 3.2 (b) shows that for the same input motion sequence, there could be multiple possible future motions. So human motion has time-dependent variation. Therefore, in contrast to a deterministic function F, the motion dynamics should be modeled as a random process $P(X_{T+1:T'}|X_{1:T})$.

3.1.3 Stylistic Variation

In this thesis, we consider motion style as a type of motion variation. The word *style* can have different meanings in different contexts. In the field of character animation, the concept of motion style seems to be straightforward, for example, male and female, old and young, happy and sad, and so on. However, there is no clear definition for motion style [THB06]. Some computer animation literature treat motion variation and style separately [LBK09; Ma+10]. Ma et al. [Ma+10] state that

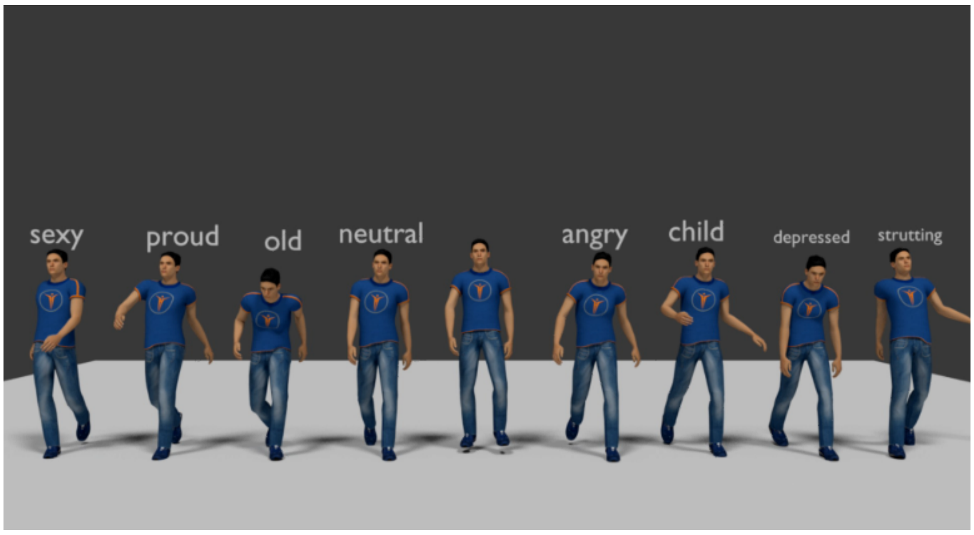


FIGURE 3.3: Different styles of walking. Style walking data is from [Xia+15].

"style differentiates between examples of the same behavior (slow walk vs. fast walk) while variation differentiates between examples of the same style (vigorous vs. lackadaisical arm swing)". However, fast and slow can also be considered speed variations of walking. There is no decision boundary for how fast is fast and how slow is slow. More generally, different actions can be treated as stylistic variations of human motion. For example, Brand and Hertzmann [BH00a] argued that "walking, running, strutting, etc., are all stylistic variations on bipedal locomotion". Generally speaking, the style usually refers to high-level variation in human motion. It could be defined by some adjective features shared by a group of people, such as gender, age, mood, physical fitness, etc. Each individual person can have their own style as well, which is sometimes considered a personality trait [RWV19].

3.2 Motion Parameterization

The choice of motion data representation impacts the effectiveness of modeling the variability presented in the data. In traditional character animation, the avatars are driven by a 3D-rigged skeleton with skinned meshes. The skeleton contains a list of articulated joints, which represent the hierarchical structure of the human body.

As shown in Figure 3.2, the visually observable variation of human motion is the position of the end-effectors of body joints. The variation is represented as joint positions in the observation space. However, we don't move each joint separately.

The human body has a complex and delicate hierarchical skeleton. The motion discussed in this thesis is skeleton-based movement. We assume the bone lengths are constant over time, the variations are the rotation of the bones, which are commonly represented relative to their parent joint. Therefore, the angular variation is the actual variation that causes the variability in joint positions. In addition, the angular variation is restricted by biomechanical and physical rules.

In traditional character animation, the avatars are driven by a 3D rigged skeleton with skinned meshes. The skeleton contains a list of articulated joints, which represent the hierarchical structure of the human body. The animators assign the values of each DOF manually or using some design tools. Motion capture technologies provide another important source of motion data. Most motion capture systems track body movement with markers. The results of recorded motions are point clouds in 3D observation space. There are also some optical-based tracking systems, which can provide the angular joint rotation with a built-in skeleton as the tracking result. Another way to obtain motion data is to reconstruct 3D motion from videos [Meh+20; Shi+20; Sun+21]. Similar to motion capturing, it can provide both joint position and joint orientation with a skeleton. Variation in motion is represented by the numerical values. The task of modeling human variation is to learn the variation in the data and create new samples that have a similar distribution to the original data. So choosing an effective way to represent human motion is important for modeling motion variation.

3.2.1 Rotational Representation

Due to the hierarchical structure of the human body, the posture can be understood as rotating the root joint and all of its descendants while keeping constant bone lengths. Therefore, It has been widely used in character animation to represent each pose as root joint global translation and orientation plus the relative joint orientation in the kinematic chain. As the bone length is kept constant for a while, angular representation can naturally decouple this invariant component from motion data. There are different methods to parameterize orientation or rotation. In general, any 3D rotation can be uniquely represented as a 3x3 matrix whose columns are orthonormal. Besides rotation matrices, there are more compact ways to represent rotation, such as Euler angles, Quaternions, exponential maps, and so on. However, these compact representations could lead to ambiguity and singularity issues [Gra98].

Euler Angles

Euler angles are probably the most straightforward way to represent 3D rotation. It is also widely used for skeleton-based motion data parameterization. Many motion capture data file formats [MM+01], for example, Biovision Hierarchy File Format (BVH), use Euler angles as motion data. Euler angles for 3D rotation are a vector of three DOFs. Each DOF represents a rotation about one of the coordinate axes. So the 3D rotation matrix $\mathbf{R}$ can be decomposed as three successive rotations $\mathbf{R}_x$, $\mathbf{R}_y$, and $\mathbf{R}_z$. Euler angles have good properties that it is easy to understand the meaning of each variable. And the derivatives are easy to compute, which is useful for inverse kinematics. However, there are some severe issues with Euler angles that limit its usage. A well-known issue with Euler angles is the gimbal lock [Sho85]. It means one DOF is "locked up" when the other two are aligned. The expected rotation cannot be applied to the locked axis. In addition, Euler angles is not ideal for the task of modeling variation of motion. The value of Euler angles is in Euclidean space but rotation is non-Euclidean [Gra98]. It could introduce unnecessary variations due to singularities [SMJ05], which means that two sets of rotation angles may correspond to the same rotation.

Quaternions

It has a long history for quaternions to be used as representing for rotations in computer graphics [Sho85]. Quaternions are hypercomplex values with a real part and three imaginary parts that represent a rotation in three degrees of freedom. A quaternion can be defined as a vector $q = [q_w, q_x, q_y, q_z]$. q_w is the real part of quaternion, and q_x, q_y, q_z are imaginary part. Compared to Euler angles, an advantage of unit quaternions is free from gimbal lock [Ala+13]. Therefore, unit quaternions are also widely used in motion modeling and synthesis [Sho85; Joh03; Du+16a; Pav+20; Xu+20] to overcome the issues in Euler angles.

Rotation Matrices

Rotation matrices are 3x3 orthonormal matrices, which means the columns are unit length and mutually orthogonal. Rotation matrices can uniquely represent 3D rotations in Euclidean space. It does not suffer singularities and ambiguities. However, rotation matrices require 3x3 elements. It is over-parameterized and ignores the intrinsic correlation between the parameters. Simply speaking, not all 3x3 matrices are valid rotation matrices. Normally, people do not directly use rotation matrices. A popular way to take advantage of rotation matrices is to use two orthogonal axes from the rotation matrix [Zha+18; Lin+20]. It is enough to clearly define the rotation. The third axis can be derived from the cross-product of the two axes.

3.2.2 Positional Representation

Since many motion capture systems track joint markers of human skeletons, an intuitive way to represent motion data is to directly use the global 3D joint positions in Euclidean space. The full-body pose is expressed as a vector of 3D positions of ordered joints. A significant amount of research work [BH00a; Hab+17; Du+19a; Lin+20; HAB20] use joint positions in body's local coordinate space for statistical motion modeling and synthesis. Compared to angular representation, positional representation does not suffer either singularities or ambiguities. In terms of effectively representing motion variation, positional representation enjoys some additional advantages.

One problem with the angular representation of motion data is that joint angles are usually relatively represented in the local joint coordinate system. It means that the different DOFs could have different impacts on the end-factor positions. For instance, the rotation of the root joint will rotate the whole body. The movement is much more noticeable than the same rotation applied to a finger. This could cause problems in modeling human variation. The variability in movement is observed in Euclidean space. Figure 3.4 shows the difference of variance for each joint using angular representation and positional representation for the same action. It clearly shows that the angular representation could scale the variations in each DOF which is not corresponding to visual observation. A large variation in Euclidean space

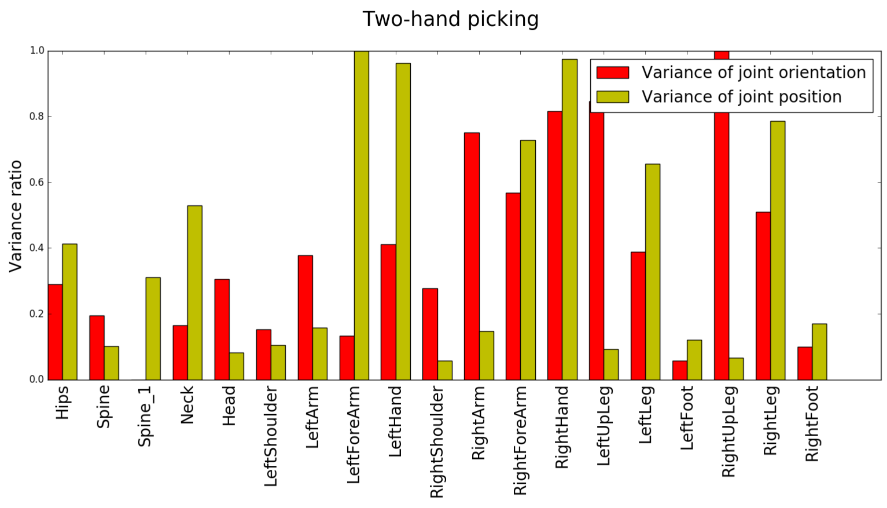


FIGURE 3.4: Comparison of joint variation of two-hand picking. (a): Variance of joints' orientation represented by quaternions. (b): Variance of joints' position in Euclidean space.

could be mapped to a small variation in angular representation. This is a problem for dimension reduction and motion modeling since the important variations might be overlooked. In positional representation, each DOF has the same unit. And the variation in the motion data is corresponding to our observation.

Another issue with the relative angular representation is error accumulation. As mentioned above, each DOF has a different influence on the joint positions in Euclidean space. Normally, the Gaussian noise from generated motions using statistical motion models is equally distributed to each DOF. This noise could have a much larger impact on angular representation than positional representation. The error in angular representation will accumulate along the kinematic chain. The end-effector joints such as hands could suffer a much larger error than the root joint.

However, positional representation also has some limitations. The most critical issue is that it cannot preserve the invariance of bone length, which is intrinsically preserved by angular representation. Additional bone length constraints are required while modeling the motion variation. Otherwise, new variances introduced by statistical methods could break the skeleton. In addition, the information about joint orientation is missing in positional representation. This could cause twisted bones for skinned character animation due to this is no constraints for joint orientations.

3.3 Learning Low-dimensional Representational for Motion Data

Effective learning usually requires a good representation of the training data. In Section 3.2, we discuss the main approaches to parameterizing motion data into numerical vectors. For the topic of learning the statistical distribution of the data, a common problem is the "curse of dimensionality". Motion data is usually high-dimensional data because of the high degrees of freedom (DOFs) of the human body. In order to learn a meaningful distribution in high-dimensional space, it normally requires a huge amount of data for training, which is usually not practical for motion capturing. Therefore, instead of directly working on the original motion space, most motion modeling approaches [TH00; CH05; CH07; THR07; WFH08; WMC11; MCC09; MC12; Du+16b; Du+19a; Lin+20; Spr+20; Gho+20] work on a latent space that the dimensionality usually is much lower than the original motion representation.

Our goal is to learn an expressive, low-dimensional space to better represent motion variation in observation space. In this section, we investigate functional data analysis on motion data and motion variation in different parametrizations. We propose a novel dimension reduction approach named Scaled Functional Principal Component Analysis (SFPCA) to address the Spatio-temporal properties of motion data. We develop a scaling strategy to automatically rescale motion features by combining forward kinematics with standard PCA. The work presented here is based on previous publications [Du+16a; Du+16b].

3.3.1 Introduction

Principal Component Analysis (PCA) [Pea01] is probably the most commonly used linear dimension reduction technique due to its simplicity and efficiency. It has been also widely used for compressing motion data. The high-frequency sampling rate of motion capture data and the intrinsic correlation between body parts indicate the redundancy in both temporal and spatial domains for motion data. Many variants of PCA have been proposed to address the challenges in human motion data. PCA is mainly used to identify the significant variations in the data. The principal components are in descending sort with the explained variance of each component. A compact low-dimensional space is created while keeping as much variance as possible, which is very important for effective statistical learning. Many existing work applied PCA on either poses [SHP04; TH00; CH05] or motion clips [MCC09; MLC10; MC12; Du+16a; Du+16b] as the necessary step before modeling.

PCA is proven to be optimal for linear data. However, human motion is vastly complicated and cannot be well described by linear models. This makes PCA unsuitable if the training data contains large variation. Chai and Hodgins [CH05] showed that performing PCA on the large and heterogeneous database could produce poor quality results. They proposed to construct local spaces by searching for nearest neighbors to the control signals. They applied PCA on the local spaces and constructed local statistical models for motion synthesis. They demonstrated that the local PCA can achieve much better reconstruction errors on a large heterogeneous database than global PCA.

Another issue for applying PCA on motion data is that PCA treats each dimension of the data equally in terms of variance. It would cause a problem for the rotational representation of motion data. As explained in Section 3.2, the hierarchical structure of rotational representation performs a nonlinear mapping of the variation in each joint in the observation space. So the reduced variance by PCA might be corresponding to a large variance in observation space. This is not a problem for positional representation, but it means that the performance of using PCA on motion data depends on the choice of motion parameterization.

Most nonlinear dimensionality reduction methods, e.g. Locally Linear Embedding [RS00] and ISOMAP [BS02], are not feasible for our work because they do not meet the requirement of an explicit inverse transform to the original motion space. This requirement enables us to map the generated random samples from the latent space back to the original motion space and visualize them. Amongst nonlinear methods, Gaussian Process Latent Variable Model (GPLVM) [Law04] is an exception that meets this requirement. GPLVM is a generalized form of PCA that can learn a nonlinear smooth mapping from the low-dimensional latent space to the observation space.

Growing research interests in GPLVM has led to many new variants of this method. A similar idea to our work is [Gro+04]. They proposed a scaled version of GPLVM to model the human poses in which the different dimensions of the observation space are of different scales. The idea behind scaled GPLVM is to introduce a scaling weight for each dimension of the observation space and learn them along with the parameters of the model and the latent space. GPLVM keeps the dissimilar points in the observation spaced apart in the latent space, however, it does not guarantee that the similar points in the observation space stay close to each other in the latent space. In [LJ06], a GPLVM with back constraints is proposed to preserve the local distances of the data from the observation space to the latent space. A discriminative prior over the latent variables is introduced by [UD07] for the class labels in GPLVM. While having the generalization ability of probabilistic methods, this method can also preserve the class labels in the latent space. However, GPLVM has been proven that it is computational expensive [Law07]. It requires computing a large covariance matrix, which can become computationally expensive as the number of data increases. The scalability limitation of GPLVM makes it difficult to work with very large databases.

3.3.2 Data Acquisition

Our motion capture data is recorded by a markerless motion capturing system Captury [Cap]. It contains 10 distinctive elementary actions performed in an assembly workshop scenario, including walking (125765 frames), carrying (413090 frames), two-hand picking (410190 frames), single-hand picking (86604 frames), two-hand placing (335310 frames), single-hand placing (50159 frames), sidestep (240734 frames), screwing (165920 frames), looking around (71034 frames), two-hand transferring (18854 frames). Figure 3.5 shows the workflow of motion capturing in our work. The output motion data is saved in BVH format. Each frame contains the global translation and rotation of the root joint and local rotation for other joints in the kinematic chain (Figure 3.6). Hence, we parameterize motion data as a vector of root translation and rotation, plus joint rotation. The orientations are represented by unit quaternions.

Similar to [MC12], the long input recordings are decomposed into small clips that have semantic meanings, for example, left steps, right steps, and so on. Structurally

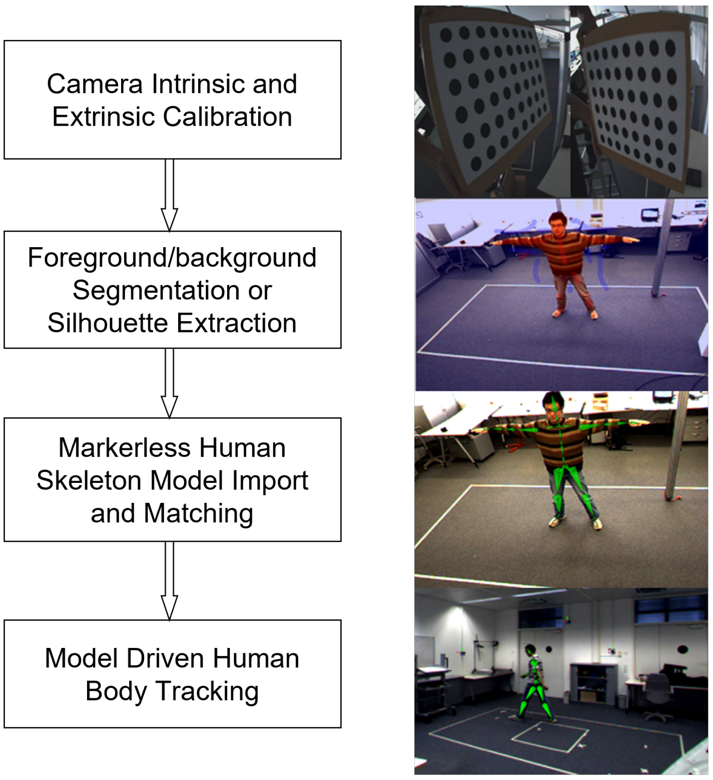

Camera Intrinsic and Extrinsic Calibration
Foreground/background Segmentation or Silhouette Extraction
Markerless Human Skeleton Model Import and Matching
Model Driven Human Body Tracking

FIGURE 3.5: The workflow of the markerless motion tracking system

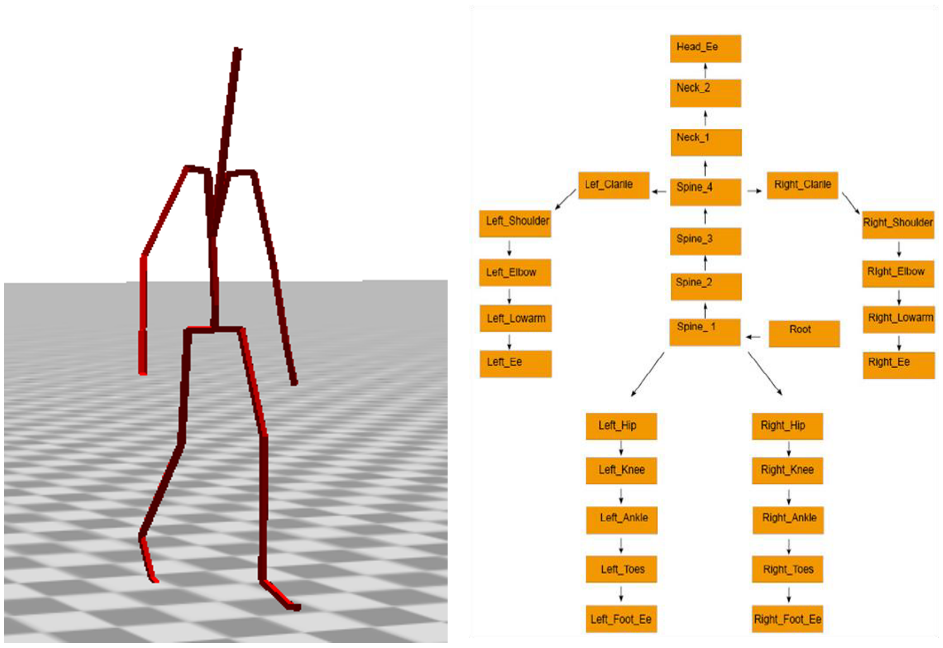


FIGURE 3.6: Left: An example of recorded walk. Right: The hierarchical skeleton structure of the character used in the capturing system.

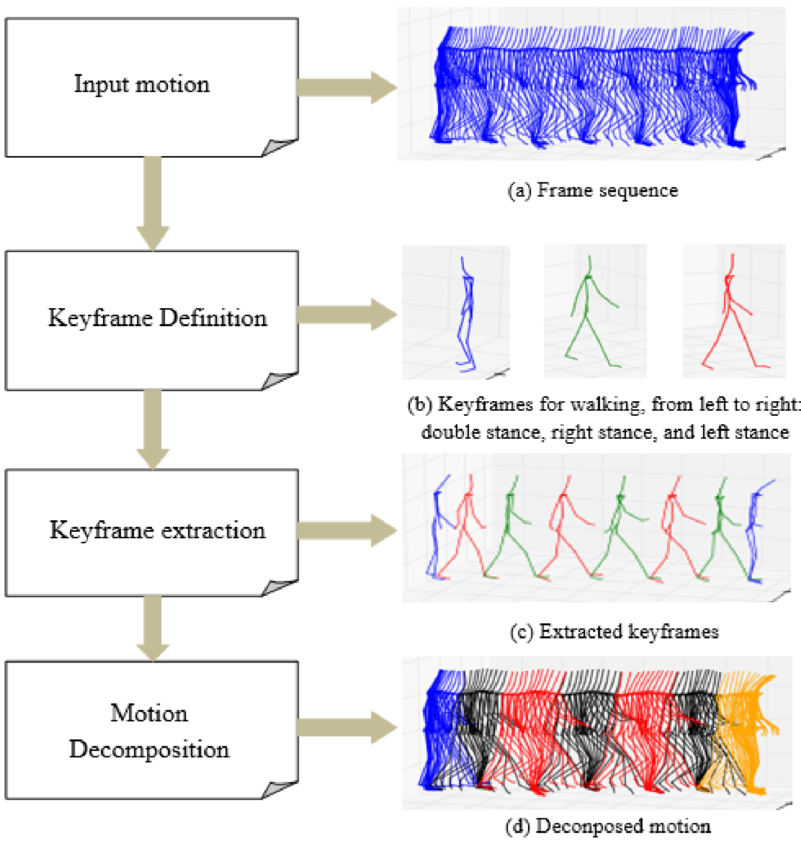


FIGURE 3.7: The pipeline of motion segmentation.

and semantically similar motion clips are grouped into classes called Motion Primitives. For example, a normal walk can be defined as an alternative to left stance and right stance. A set of keyframes are predefined to cut the motion sequences into small clips. A good decomposition should make the dissimilarity very small for clips within the same motion primitive, and very large for clips in different motion primitives. The number of keyframes for a good decomposition depends on the complexity of the motion. Figure 3.7 shows an example of segmenting walking into motion primitives. Motion clips that are within the same motion primitive could have different root positions, orientations, and the number of frames. We translate and rotate motion clips to have the same starting position and orientation.

3.3.3 Functional Data Analysis on Motion Data

Human motion changes smoothly and continuously over time. Therefore, the motion capture data which is smooth and continuous-time as well can be intrinsically represented and analyzed in the functional domain. There are two advantages associated with this representation of the motion: Firstly, motion capture data usually contains noise, which can be smoothed by functional data representation; Secondly, there is quite a lot of redundant information in the time domain due to the high frame rate, and functional representation offers a compact representation of the frame sequences to reduce this redundancy. When applying PCA on motion sequence, the temporal consistency between poses is not considered. The poses are commonly sequentially concatenated as a long vector. Functional PCA treats motion data as functional data [WCM16], which means each DOF is a continuous function, instead of a set of discrete values. PCA is then used on the coefficients of the functions. So temporal smoothness of motion data is intrinsically constrained by functional representation.

In this work, we consider frame sequences as multivariate functional data and the discrete values of each dimension over frames are interpolated as a smooth function represented by a linear combination of cubic B-spline functions:

$$\{q_{i1}, ..., q_{in}\} \rightarrow q_i(t) = \sum_{k=1}^{K} c_{ik}\phi_k(t), \;\; K \ll n \tag{3.1}$$

where, q_{ij} denotes quaternion value of i^{th} dimension of the motion in the j^{th} frame, $\phi_k(t)$ is the k^{th} cubic B-spline basis function and c_{ik} is the coefficient of k^{th} B-spline basis function for the i^{th} dimension of the motion.

By applying this functional representation, each motion $\mathbf{y_i}$ can be represented as a vector of continuous functions instead of a very long multivariate vector (Figure 3.8).

$$\mathbf{y}_i(t) = \{\mathbf{p}_i(t), \mathbf{q}_{i1}(t), ..., \mathbf{q}_{im}(t)\}^T \tag{3.2}$$

where $\mathbf{p}_i(t) \in \mathbf{R}^3$ is the root translation of i^{th} motion clip $\mathbf{y_i(t)}$, and $\mathbf{q}_{ij}(t) \in \mathbf{R}^4$ is the j^{th} joint orientation represented by quaternion.

Since the motion data is sampled at a constant frame rate, we select equally-spaced knots for our B-spline representation. The choice of the number of B-spline basis functions K depends on the complexity of each motion type. In our work, for each type of action, we use the cut-off point that increasing the number of basis functions will not significantly reduce the root mean squared error in joint space.

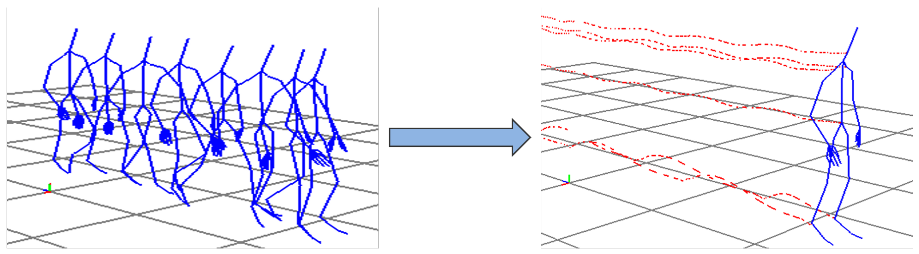

A sequence of frames: $Y = \{f_1, \dots, f_n\}$
A set of continuous functions: $Y(t) = \{p(t), q_1(t), \dots, q_m(t)\}$

FIGURE 3.8: Represent a sequence of frames as a set of continuous functions. $p(t) \in R^3$ is the root translation, $q_i(t) \in R^4$ is the i^{th} joint orientation represented by quaternions.

Figure 3.9 shows the reconstruction error between B-spline representation and original discrete data. The reconstruction error is measured by evaluating functional data at canonical frames and comparing it with the original data. Generally, we are interested in using fewer basis functions while keeping the functional data as close to the original data as possible. It seems that some complex motions, for instance, picking and placing, require more basis functions to achieve the same reconstruction error, however, the number of canonical frames is different for different types of motions. Usually, the length of an aligned motion clip in complex motions is longer than in simple motions. So the reduction rate of functional representation is similar for most motions, which indicates that the cubic B-spline basis works well for our motion data.

3.3.4 Scaled Functional Principal Component Analysis

We now introduce how to apply a scaled version of Functional Principal Component Analysis (FPCA) for motion data. FPCA reduces the dimensionality of data by finding a subspace that accounts for as much of the variability in the data as possible. For functional data, the subspace is defined by a set of orthonormal eigenfunctions $\mathbf{V}(t)$, which minimize the reconstruction error in feature space:

$$\arg\min_{\mathbf{V}(t)} \frac{1}{N} ||\mathbf{Y}(t) - \mathbf{V}(t)\mathbf{V}^T(t)\mathbf{Y}(t)||_F^2 \tag{3.3}$$

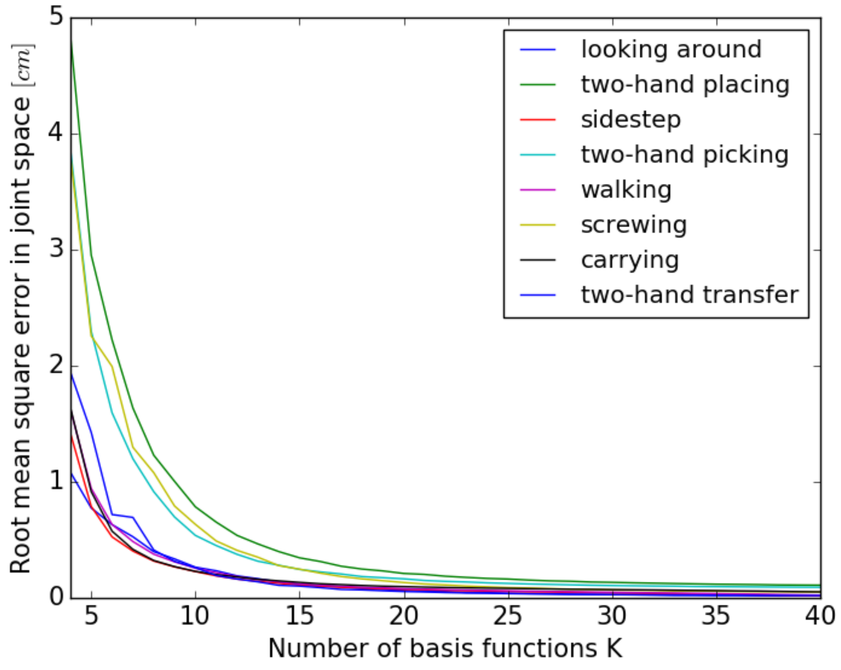


FIGURE 3.9: Evaluation of functional data representation for different motions. Root mean squared error (RMSE) between functional data and discrete data is measured in joint space. RMSE for each motion type is plotted with an increasing number of basis functions K.

$$\mathbf{V}^T(t) = \begin{pmatrix} \boldsymbol{\xi}_0(t) \\ \vdots \\ \boldsymbol{\xi}_p(t) \end{pmatrix} = \begin{pmatrix} \xi_{0,\mathbf{p}}(t) & \cdots & \xi_{0,\mathbf{q}_m}(t) \\ \vdots & \ddots & \vdots \\ \xi_{p,\mathbf{p}}(t) & \cdots & \xi_{p,\mathbf{q}_m}(t) \end{pmatrix} \tag{3.4}$$

where $\mathbf{Y}(t)$ is the functional representation of input motion clips, $\mathbf{V}(t)$ are eigenfunctions of $\mathbf{Y}(t)$, and N is the number of motion clips.

The inner product of two vector of functions $\boldsymbol{\xi}(t)$ and $\boldsymbol{y}(t)$ is defined as:

$$\boldsymbol{\xi}(t)\boldsymbol{y}(t) = \sum_{i=1}^{m} \int \xi_i(t) y_i(t) \mathrm{d}t \tag{3.5}$$

each eigenfunction $\xi_i(t)$ is a vector of eigenfunctions for each dimension of scaled functional data $\mathbf{WY}(t)$.

However, for motion data, joints could have different variances under different parameterizations. Figure 3.4 compares the variance of each joint for the motion of two-hand picking measured in different spaces. The variances by using these two parameterizations are significantly different for different motion representations. FPCA tends to reduce the information with low variance in the feature space.

This could cause a big difference in the observation space. Based on this observation, we propose to measure the reconstruction error between the original motion and the reconstructed motion using forward kinematics in observation space rather than feature space (Eq. 3.6), which corresponds to human visual observation.

$$\arg\min_{\mathbf{V}(t)} \frac{1}{N} ||f_k(\mathbf{Y}(t)) - f_k(\mathbf{V}(t)\mathbf{V}^T(t)\mathbf{Y}(t))||_F^2 \tag{3.6}$$

where $f_k(\mathbf{Y}(t))$ is the forward kinematics that maps functional motion data $\mathbf{Y}(t)$ to joint position in Euclidean space.

For motion modeling and synthesis task, dimension reduction methods need to have a small reconstruction error, since human is sensitive to human motion quality. So a good dimension reduction mapping $g(x)$ should minimize pose error as follows:

$$||f_k(\mathbf{X}) - f_k(g^{-1}(g(\mathbf{X})))||^2 \tag{3.7}$$

Applying FPCA on $\mathbf{Y}(t)$ can find optimal solution $\mathbf{V}(t)$ for Eq. 3.3, however, $\mathbf{V}(t)$ is not the optimal solution for Eq. 3.6.

Finding an analytical optimal solution for Eq. 3.6 is nontrivial due to the complexity of forward kinematics. In our work, we apply a simple scaling method to further reduce the reconstruction error defined by Eq. 3.6, similar to scaled GPLVM [Gro+04]. We adopt a similar idea to our problem with two modifications. Firstly, we apply to scale on functional data rather than discrete data, which significantly reduces the number of weights to be estimated by the factor of the number of frames and reduces the risk of overfitting. Directly scaling each dimension of the motion clip is not practical in our work. For example, for two-hand placing reach, each motion clip has 145 frames and each frame has 79 variables. If we sequentially concatenate frames, each motion clip will have 11455 dimensions. In this case, optimizing the weight for each dimension is not only computationally expensive but overfits the data as well. Secondly, the weights are calculated by minimizing the average squared error in the Euclidean joint space. We use this prior knowledge to guide the calculation of the weights. An optimal functional space $\mathbf{V}'(t)$ is found by employing FPCA on scaled functional data $\mathbf{WY}(t)$. The weight $\mathbf{W}$ is a diagonal matrix

$diag(w_1, ..., w_d)$. So Eq. 3.6 can be rewritten as:

$$\arg\min_{\mathbf{W}} \frac{1}{N} ||f_k(\mathbf{Y}(t)) - f_k(\mathbf{W}^{-1}\mathbf{V}'(t)\mathbf{V}'^T(t)\mathbf{W}\mathbf{Y}(t))||_F^2 \tag{3.8}$$

The optimal weight **W** can be found by minimizing Eq. 3.8. We apply numerical optimization algorithm L-BFGS-B [Byr+95] to iteratively find optimal value for weights of different number of principal components. At each evaluation of Eq 3.8 in optimization, the eigenfunctions $\mathbf{V}^k(t)$ is updated by applying FPCA on updated scaled motion data $\mathbf{W}^k\mathbf{Y}(t)$.

The experiment results demonstrate that our approach works well in practice and can achieve a better result than standard FPCA for reconstruction error in joint space. In addition, since the result of $f_k(\mathbf{Y}(t))$ does not depend on the choice of parameterization method, our scaling approach can be automatically adapted to different motion data representations.

The motion clip is represented as a vector of continuous functions and we scale each function with one weight, rather than one weight per dimension per frame. It makes the scaling more generative and significantly reduces the computational efforts to calculate the weights since the weight vector is much shorter for functional data than discrete data. The optimal weights can be found by minimizing Eq. 3.8. Here, L-BFGS-B algorithm [Byr+95; Zhu+97] is applied to find the optimal value for the weights. The initial weights are set heuristically to accelerate the optimization and reduce the chance of getting stuck in a local minimum. We first compare the reconstruction error of Eq. 3.7 using raw functional motion data and normalized functional motion data (using Z-score normalization) and take the one with a smaller error as the initial guess.

3.3.5 Experiments

In this section, we provide a quantitative analysis of our approach to a fairly large motion capture database for multiple actions. Table 3.1 shows the detailed information of part of motion primitives used in our experiments. We compare our method with two baseline approaches: FPCA and normalized FPCA. Normalized FPCA normalizes the input features to have 0 mean and 1 standard deviation before applying

FPCA. For FPCA, normalized FPCA and our method, we all use a cubic B-spline basis. The same number of basis functions is used for each motion primitive. In addition to linear dimension reduction methods, an advanced nonlinear method Discriminative Prior Bayesian GPLVM is evaluated as well.

Discriminative Prior Bayesian GPLVM combines two variants of standard GPLVM: Bayesian GPLVM [LJ06] and discriminative GPLVM [UD07]. Bayesian GPLVM is robust to overfitting, especially for our case the dimension is much higher than the number of samples. Discriminative GPLVM integrates the discriminative prior over latent space $p(\mathbf{X})$ into likelihood function $p(\mathbf{Y}|\mathbf{X})$ to optimize. Therefore it can learn a discriminative latent space from a small training set.

In our experiments, we implement FPCA described in [RS05]. Standard PCA is applied to the coefficients of functional data. The implementation of DPBayesianG-PLVM in GPy package [GPy12] is employed. Radial basis function (RBF) kernel is used to perform a nonlinear mapping from latent space to original space. The L-BFGS-B optimizer is chosen to optimize latent parameters. For the discriminative prior over latent space, we automatically label the training motion clips according to their pose similarity. Motion clips are parameterized as joint position splines, and clustered by k-means++ algorithm [AV07]. The number of clusters is set heuristically. For example, although motion clips in "walk leftStance" contain a rich variation, the main visual styles are three: walk to the left, walk to the right and walk forward. In this case, 3 clusters are set for walk leftStance.

We evaluate the latent space constructed from different approaches based on the aforementioned criteria. First, the low dimension representation in latent space should be able to reconstruct original motion without an observable visual difference. So we evaluate the average reconstruction Euclidean error between original motion data and reconstructed motion data in two spaces: Feature space and observation space. The reconstruction error in feature space can tell us how good the dimension reduction method is in reconstructing input data, the reconstruction error in observation space tells how good the dimension reduction method is in reconstructing data for visual observation. Secondly, the motion clips which are similar in observation space should stay close in latent space as well. So we evaluate the distribution of pre-labeled samples in latent space.

TABLE 3.1: Examples of motion primitives

Motion primitive	No. motion clips	No. basis	No. low dimensions
walking leftStance	1366	10	42
two-hand picking reach	374	25	45
two-hand carrying leftStance	280	15	28
two-hand transferring	82	20	22

Reconstruction Errors

Fig 3.10 compares the reconstruction error of all methods for two motion primitives: walk leftStance and two-hand picking reach. We select these two motion primitives because they are representative of our database: One stands for locomotion, and the other stands for manipulation motion. For locomotion, e.g. walking, and carrying, the dominant variance is root trajectory (root translation), and the variance of the poses is less. In contrast, for manipulation motion, e.g. picking, placing, transfer, and so on, the pose variation is usually larger than the root translation.

DPBayesianGPLVM achieves comparably small reconstruction error for most dimensions compared to FPCA based approaches. However, there are some exceptions for walk leftStance. The reconstruction errors go up with the increase in the number of dimensions, and the values are fairly large compared to other cases. For FPCA and its two scaled variants, the reconstruction error monotonically decreases with the increase of the number of dimensions. FPCA yields a lower reconstruction error than SFPCA and normalized FPCA in feature space. The reconstruction error is very close to 0 when the number of dimensions is larger than 10. SFPCA gets the lowest reconstruction errors in Euclidean space for the most number of dimensions, except for the first few dimensions, which are larger than DPBaeysianGPLVM. Normalized FPCA has the largest reconstruction errors for most dimensions however, it is noticeable that the reconstruction error for two-hand picking in observation space is smaller than FPCA. For both motion primitives, it is sufficient for FPCA and DPBayesianGPLVM to use 10 dimensions for achieving a reconstruction error less than 0.1 in feature space, however, the reconstruction error in joint space is significantly different for two motion primitives.

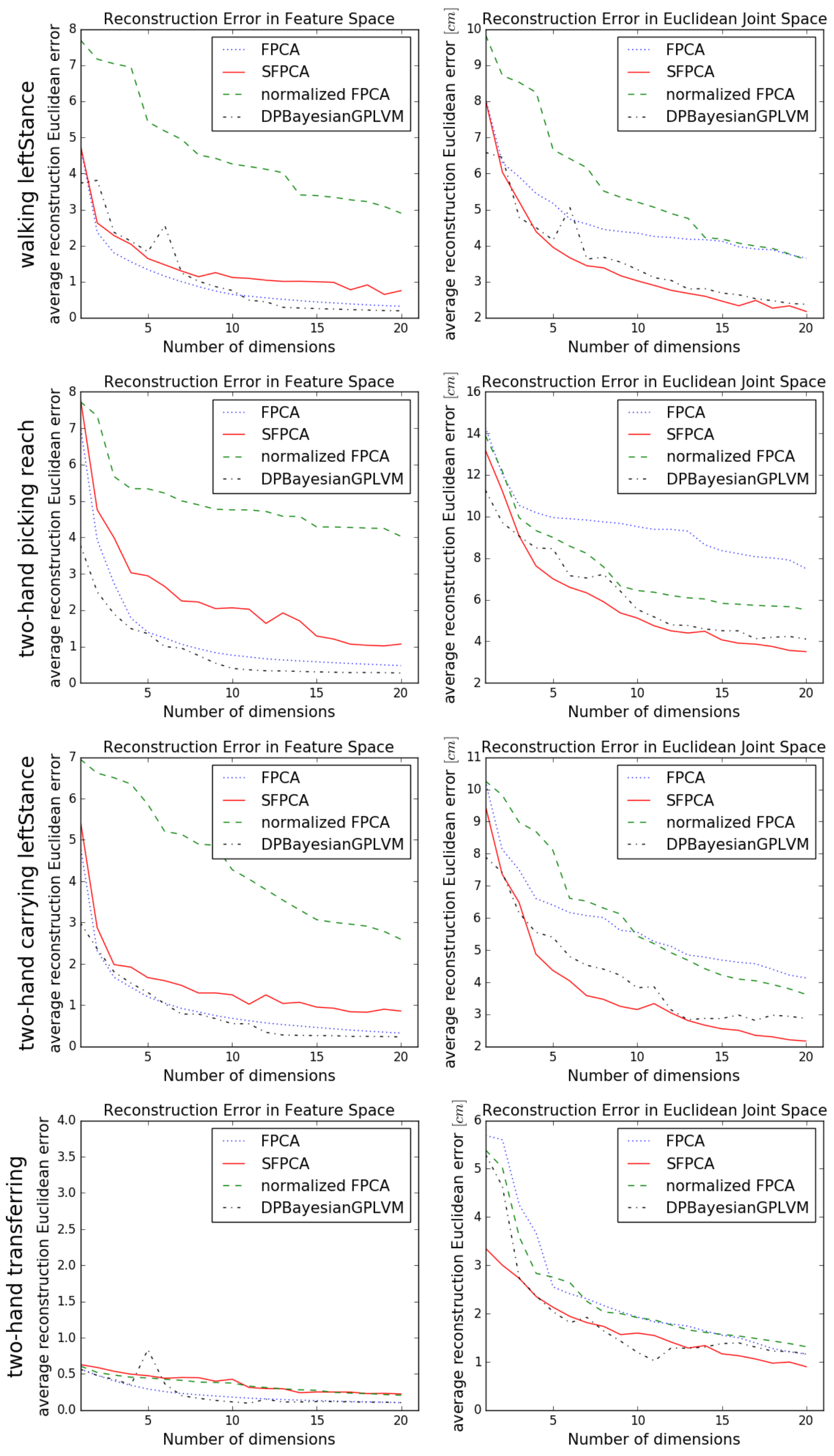


FIGURE 3.10: Comparison of four dimension reduction methods: Each curve shows the change of average reconstruction Euclidean error with the increasing of number of dimensions.

Latent Space Analysis

Figure 3.11 shows the distribution of pre-labeled motion clips in the latent space for two motion primitives: walking leftStance and two-hand picking reach. The original motion data is reduced to 20 dimensions using four testing methods separately. Here, we choose 20 dimensions to make the reconstruction errors fairly small in both feature space and observation space for both motion primitives. The samples in latent space are visualized in 2D space by t-SNE algorithm [VH08], which is a widely used approach to keep the local structure of original data. For walking leftStance, FPCA and SFPCA both keep clusters in latent space, although the boundary is not separated. DPBayesian GPVLM mixes the samples of walk to the right and walk forward. Normalized FPCA yields six clusters in latent space. Each cluster contains samples from pre-labeled classes. For both-hand picking reach, SFPCA clearly separates three types of picking. FPCA mislocates some samples of picking from the bottom into picking forward. DPBayesianGPLVM and normalized FPCA can keep the main structure in latent space, while a few samples are mislocated.

3.3.6 Discussion

Overall, FPCA shows better performance than normalized FPCA and our method for reconstruction error in feature space since FPCA can find an optimal linear mapping to minimize the average squared Euclidean error. Normalized PCA and our method both scale the motion data, so the linear mappings are optimal for scaled space, but not for original feature space. DPBayesianGPLVM, for most cases, outperforms PCA since it takes PCA result as the initial guess, and optimizes latent variables to maximize the posterior $p(\mathbf{Y}|\mathbf{X})$. However, we notice that since the optimization method does not guarantee to find a global optimum, it may converge fast due to the local optimum. That's why for walk LeftStance, the reconstruction error sometimes goes up with the increasing number of principal components.

For motion synthesis, we are interested in minimizing the reconstruction error in observation space, which corresponds to visual observation. Since the error measure is actually in a nonlinear mapped space from motion data, so standard PCA is worse than SFPCA. For walking, the PCA result is still better than normalized PCA. This

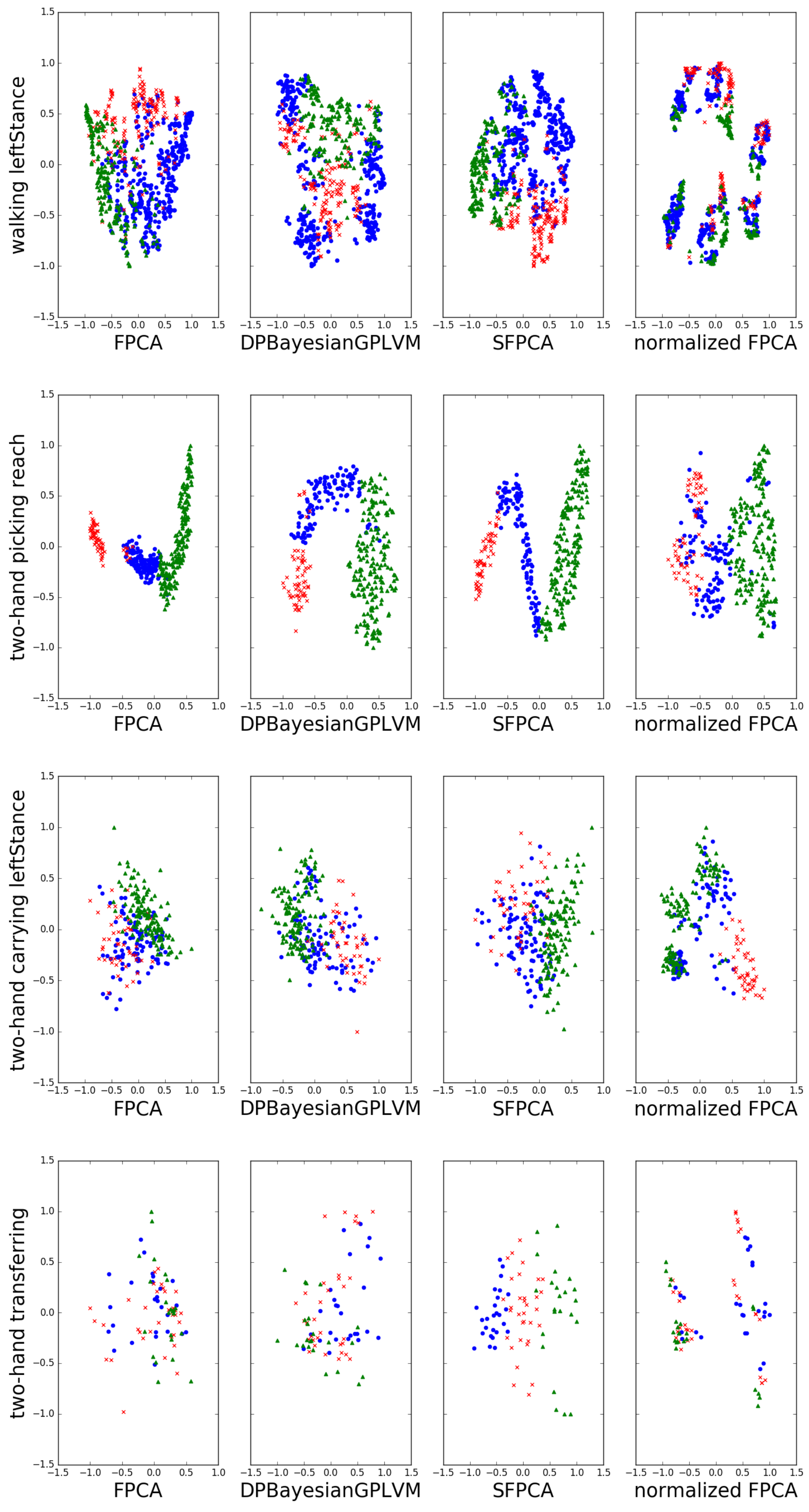


FIGURE 3.11: Clusters of latent space for walk leftStance and two-hand picking reach using FPCA, Discriminative Prior Bayesian GPLVMSFPCA, SFPCA and normalized FPCA separately. For walk leftStance, red cross is walk forward, blue circle is walk to the left and green triangle is walk to the right. For two-hand picking reach, red cross is picking from bottom, blue circle is picking forward and green triangle is picking from top.

is because Hips translation is the major variance in both feature space and Cartesian space. PCA put most effort to keep this variance in feature space, so it can also achieve a better result than normalized PCA in Cartesian space. However, in the case of picking, the variance of Hips translation is less compared to the variance of other joints, for instance, hand waving. PCA still throws away these 'unimportant variances' in feature space, but normalized PCA equally keeps the variance of each dimension, so the pose variation is better reconstructed. DPBayesianGPVLM also works better than PCA and normalized PCA for Cartesian error. Our method can achieve a comparable result as DPBayesianGPLVM. This is because we bring the nonlinear mapping from feature space to Cartesian space as prior knowledge for weights optimization. In addition, we notice that the result of our method is more stable than DPBayesianGPLVM, even though we also have the risk to get a local optimum for weights optimization.

For walking leftStance, SFPCA and FPCA achieve similar results. Motion clips which are close in Cartesian space are still close to each other in latent space. This is because SFPCA does not gain much against FPCA for reconstruction error in Cartesian space, and the number of clusters is set based on the direction of trajectory, which is well reconstructed by FPCA. However, the boundary is not clearly separated for both. We noticed that we have a very rich variation of walking leftStance due to the largest amount of samples for this motion primitive, so the change of trajectory seems continuous. The data is not clearly separated by the direction of trajectory as we assume. But, in general, the clusters in observation space are preserved well by SFPCA and FPCA. Although DPBayesianGPLVM achieves good reconstruction error in both feature space and Cartesian space, it fails to keep to local structure in latent space for walking leftStance.

For two-hand picking reach, our SFPCA outperforms other approaches. The samples in latent space are clearly separated and well clustered. The pose variation of picking motion is significantly large than walking, and motion clips are more distinguishable. So for SFPCA, the boundary of each cluster is better separated than walking leftStance. FPCA is more sensitive to variation of body trajectory than pose since the root translation dominates the variance in feature data. Normalized FPCA emphasizes pose variation, however, the naive scaling overemphasizes the importance of each joint. The contribution of variation of each joint to visual variation of the pose is clearly different. DPBayesianGPLVM works well for this case, however, the boundary of each cluster is not as good as SFPCA.

3.3.7 Conclusion

In this work, we analyze different motion data parameterization approaches and their corresponding visual effects observed from Euclidean joint space. A latent space for statistical motion modeling is presented based on Scaled Functional Principal Component Analysis. We conclude that functional data analysis can represent motion data in a more generative and compact way by reducing the redundancy of the data in the time domain. Scaling motion data based on visual similarity in Euclidean joint space can guide the mapping of motion data from high dimensional space to latent space in a more meaningful sense. We demonstrate that taking this prior knowledge into dimension reduction results in an improved latent space in terms of reconstruction error in observation space and local structure preserving compared to no and naive scaling.

In principle, our approach is general for different motion parameterizations. We construct our approach on top of PCA, however, we believe that our idea to measure the reconstruction loss in observation space using forward kinematics can be easily adapted to other nonlinear dimension reduction approaches for motion data. They can be benefited from functional data representation and taking visual similarity in Euclidean space as prior for scaling.

Chapter 4

Morphable Graph for Human Motion Modeling and Synthesis

In this chapter, we introduce our generative model named Morphable Graph for digital human modeling and simulation in assembly workshops. Digital simulation tools are widely used in the planning and verification of assembly processes. The digital human model serves as one of the key components of these simulation tools. It is valuable to utilize workers' knowledge of executing manual assembly tasks and include it in the digital tools used to support the design, verification, validation, modification, and continuous improvement of human-centered, flexible assembly workplaces. The current trend towards greater customization and personalization necessitates that digital human models be able to be rapidly adapted and seamlessly expanded. In addition, ergonomics analysis in assembly workshops becomes more and more important and an effective evaluation requires the generated motions to be high-quality and realistic. Therefore, our goal is to provide an animation system that can produce high-quality human motions with large variations in a compact and scalable manner.

Our framework relies on a graph-based model for human motion modeling and synthesis. Graph-based motion synthesis approaches [KGP02; MC12] have demonstrated their ability to convert unstructured motion capture data into structured data that can be efficiently searched and reused. A directed graph is used to model and generate human motions. Our model builds on previous successful work Motion Graphs++ [MC12]. Similar to [MC12], we decompose the complex motion variation as high-level variation and low-level variation. The high-level structure, which is named Motion Primitive, is finite and represented as the nodes in the graph. The low-level structure is modeled as a continuous distribution using a statistical model. Each edge indicates the possible transitions between motion primitives.

In [MC12], the transitions between motion primitives are also modeled as statistical models using Gaussian Processes. In our work, we find out that the GP-based transition models do not scale well with a large amount of data. It also suffers a large memory cost compared to our motion primitive models. In addition, in the scope of the assembly workshop simulation, we need to model the transitions between many different actions. Some of them do not have many transition examples. This could lead to a GP model with large variation, which might produce motions with noticeable artifacts. Therefore, we extend [MC12] to the assembly workshop scenarios by constructing a more compact low-dimensional motion space (Section 3.3) and a best-fit simulation strategy for general and fast motion synthesis. We formulate the important features for a smooth transition between motion primitives such as last posture, root velocity, and so on as the constraints for the next motion primitives. So the smooth transition becomes the best-fit search in the continuous low-dimensional feature space. Our idea is most similar to Motion Matching [Cla16], which is a simple but powerful approach widely used in the game industry. In order to achieve fast motion synthesis, we discretize our statistical model by generating a very dense set of samples, then construct a space partitioning tree for fast searching.

Our framework is implemented and applied for the EU Project INTERACT [INT] Interactive Manual Assembly Operations for the Human-Centered Workplaces of the Future, which is funded by the European Union Seventh Framework Programme. Massive evaluation has been conducted for different use cases in the INTERACT project. The techniques and ideas presented here are based on previous publications [Du+16b; Her+17; Man+18].

4.1 Introduction

Today's fierce competition on the global scale requires automotive manufacturers to enrich the variety of products and reduce the cost, meanwhile maintaining high quality and efficiency [GEE06]. Product variants usually lead to large numbers of process variants, such as tools to be used, sequence of tasks, forces/torques to be applied, and so on. These requirements pose challenges to the traditional way of production analysis and verification. For instance, in today's automotive industry, the most widely used method to design and verify a new product and its corresponding assembly system is based on a physical prototype [NWW12]. A physical prototype is an early release of a product built to test concepts or processes of the product [Har+06]. This sets out the prerequisites for completing the entire assembly process, including determining the optimal walking path for workers, positioning the workbench, selecting components to be utilized, conducting time analysis and ergonomic evaluations, and various other considerations. However, giving clear answers to these questions is not a trivial task, due to it is very hard to consider all the possible cases at a very early stage of development. Therefore, a successful design of a physical prototype usually undergoes several design cycles and variations until a final version is reached [Kul+11]. This procedure is very time-consuming and costly. And the cost for modification of the prototype increase when the product development cycle nears its end [FWW83]. In addition, there are no guarantees for future extension of later products in all respects, because it is nearly impossible to simulate all possible situations by physical simulation at the very early stage [MBF08]. Another problem is that each prototype is specific to one product, which is extremely unsuitable for today's requirement of a large variety of products. So in order to meet the requirements of a customer-driven market, novel ways to design and verify products and their corresponding assembly systems are highly demanded.

The most popular approach to replacing physical prototypes is to use virtual prototyping (VP) [Kul+11; Pan+05; Zor+03]. Virtual prototyping employs digital design and simulation tools for production analysis and verification. CAE/CAD tools, such as AutoCAD, and CATIA, are the early virtual design prototyping. Their application reduces the design cycle time and improves quality standards [McL93]. One of the essential components of digital simulation tools is digital human models. The demands for variability in performing an assembly task and ergonomics analysis require virtual characters can produce realistic movements with a large diversity. However, in the early version of virtual prototyping, the modeling of human activities is either not included or roughly modeled. The increasing number of product variants requires more actions and variations are included in the simulation tools. The requirement cannot be easily solved by manual or procedural motion simulation approaches due to the infeasible efforts to include a huge number of examples and find smooth weights to blend them.

This limitation does not mean that including rich and realistic variations in digital human models is impossible. Data-driven motion synthesis approaches provide an appealing solution for efficiently generating a wide range of natural-looking motions with large variations. However, one of the core challenges is how to efficiently reuse recorded motion capture data and synthesize new motions that can be easily controlled [WN15; Kov04b].

In order to deal with the aforementioned challenge, graph-based motion synthesis, and modeling approaches have been proposed as a powerful data structure to define the transitions between unstructured motion data. Motion Graph [KGP02] automatically constructs a graph structure to allow transition between each pair of frames, and converts the motion synthesis problem into a graph search problem. Motion Graph works well for a small dataset containing thousands of frames. However, it does not scale well for larger datasets containing millions of frames. In addition, motion graphs cannot generate new motions that are not in the motion database. For large motion databases, a high-level structured motion data representation is required. Motion Graphs++ [MC12] assumes that although human motion appears to have infinite variations, the fundamental high-level structures are always finite. For example, normal walking can be regarded as a sequence of alternating left and right stances, and picking can be decomposed as reaching and retrieving. They construct a generative statistical graph model using structurally similar motion clips and validate the ability of their model to interactively generate controllable, natural-looking motion on a large database.

For the task of motion synthesis, we propose a best-fit simulation approach. Our Morphable Graph framework converts the motion synthesis task into a graph search problem. First, a graph walk is generated to match the high-level input, such as the decomposed action sequence. Then we search the optimal sample in each motion primitive to best fit the transition and target constraints. A novel space partitioning tree of motion data is proposed to efficiently search the latent motion space and find the best candidates.

Some motion synthesis approaches such as [Kov04a] and [HG07a] make use of space partitioning data structures for motion retrieval. Space partitioning data structures on low-dimensional feature representations have been explored extensively for the application of retrieving high-dimensional time series data such as motion capture data. Krüger et al. [Krü+10] evaluate different feature representations for pose retrieval using kd-trees and present a fast motion retrieval method based on graphs constructed using the result of the pose retrieval queries of individual frames. Kapadia et al. [Kap+13] present a motion retrieval approach that allows the definition of arbitrary key features by experts which are then used to construct a novel trie-tree data structure. A node in the trie-tree represents a key and the leaves of the tree contain references to motion clips that fit the sequence of keys that is obtained by traversing the tree from the root to the leaf. Bernard et al. [Ber+13] present a visual motion database exploration tool that makes use of k-Means clustering of poses represented as feature vectors containing 3D positions of important joints to generate interactive dendrograms of pose accumulations at different levels of detail.

In order to accelerate our best-fit simulation strategy, we evaluate the use of space partitioning data structures on the low-dimensional space generated from statistical motion models. Similar to motion retrieval methods presented by [Krü+10] and [Ber+13], we construct a space partitioning data structure on a motion feature representation. However, in contrast to these methods, we do not search for a motion similar to an input motion but a motion fitting to arbitrary user-defined spatial constraints based on an objective function. We give an overview introduction of our system in Section 4.2. Section 4.3 describes the motion data used in our project and the details of the preprocessing pipeline. Section 4.4 and Section 4.5 provide the technical details about motion modeling and synthesis. We present our experimental results in Section 4.6.

4.2 System Overview

Figure 4.1 shows the overview of our system. It contains two parts: Offline motion modeling and online motion synthesis. For offline modeling, we analyze the input

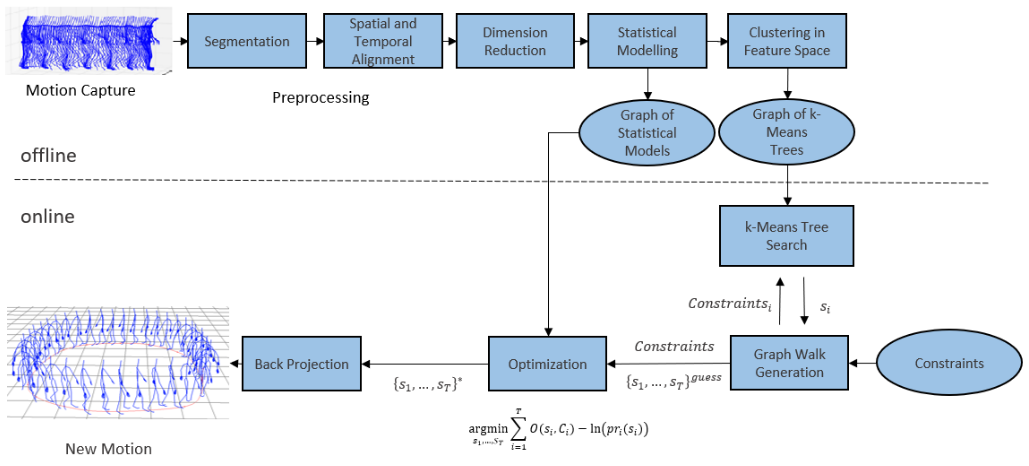
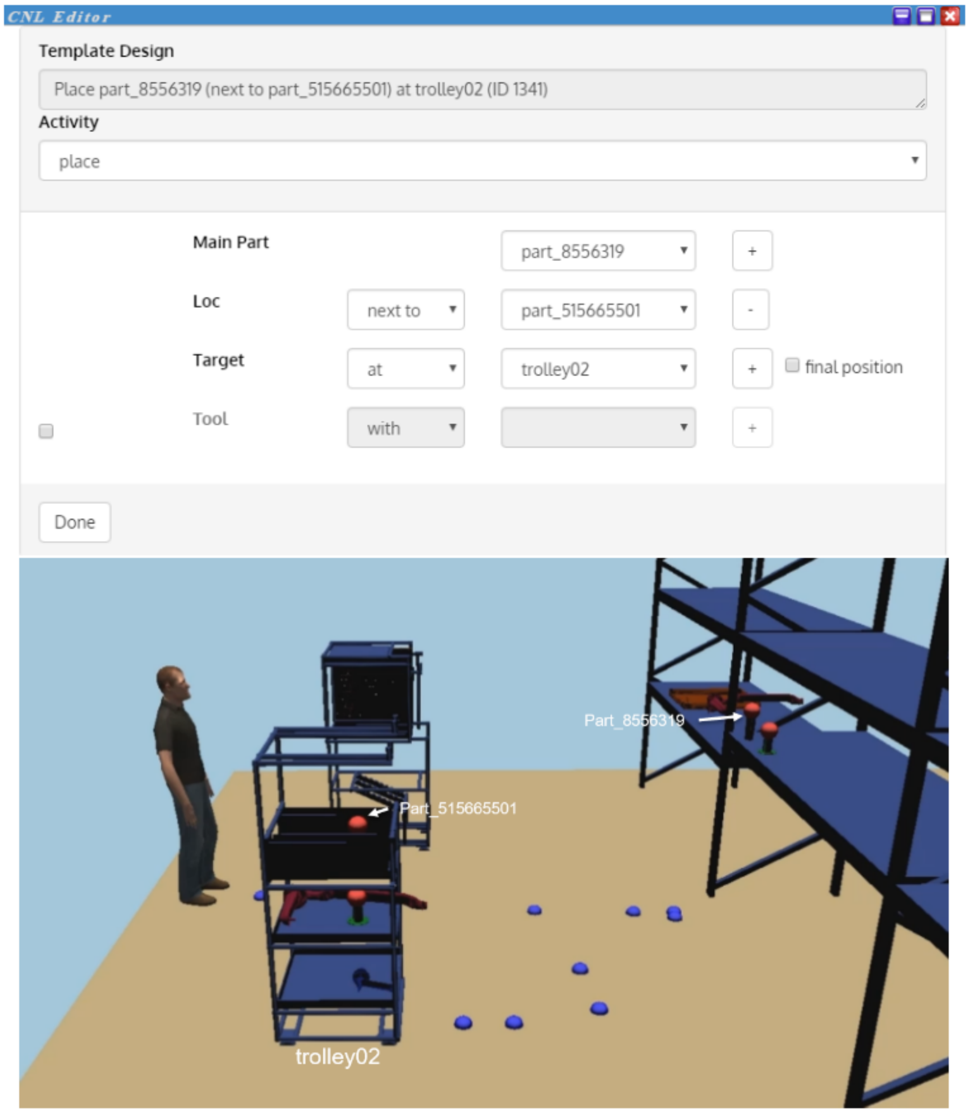


FIGURE 4.1: Overview of Morphable Graph pipeline. Here s_i represents the motion vector in latent space. [Her+17]

actions and predefine the high-level structures named motion primitives for each action. A directed graph is deployed to describe the transitions between motion primitives. The input motions are segmented and categorized into each motion primitive. For motion clips in each motion primitive, a low-dimensional representation space is learned (Section 3.3) and the distribution of samples in the low-dimensional space is modeled using statistical models. In addition, in order to accelerate the synthesis tasks, we discretize the continuous distribution into a dense sample space. We sample the motions in low-dimensional space with a high likelihood that the quality of the motions is guaranteed to be good.

For online motion synthesis, our system can support different kinds of user-defined constraints. The assembly tasks described by Controlled Natural Language (CNL) [BSH16] can be directly used in our framework (Figure 4.2). The users can also create new motions by simply sketching the target trajectory or position through our graphical interface (Figure 4.3). A graph walk is generated by traversing our Morphable Graph to reach the constraints. The best-fit sample for each motion primitive in the graph walk is found and optimized. We do not directly perform optimization on random samples from statistical models, which could lead to motions with noticeable artifacts. Instead, an efficient data structure is proposed to assist the searching for the best-fit motion. The best-fit candidate is used as the initialization for further optimization to reach the constraints exactly.



Template Design

Place part_8556319 (next to part_515665501) at trolley02 (ID 1341)

Activity

place

Main Part		part_8556319	+
Loc	next to	part_515665501	-
Target	at	trolley02	+ ☐ final position
☐ Tool	with		+

Done

FIGURE 4.2: The interface of Morphable Graph system. The task is described by controlled natural language.

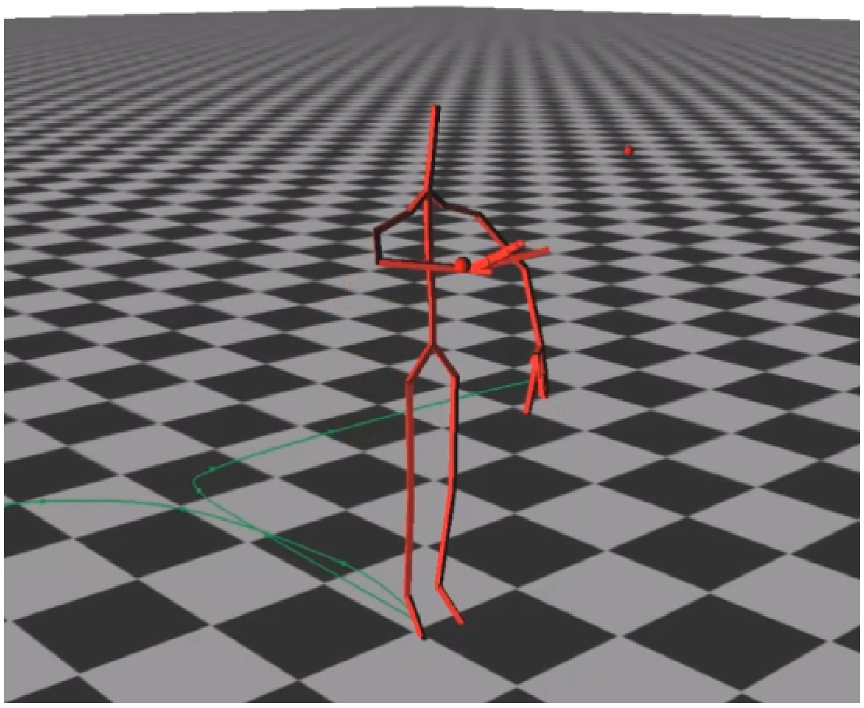


FIGURE 4.3: The graphical interface of Morphable Graph. The green line on the ground is the trajectory constrain and the red dot is the keyframe constraint for hand joints.

4.3 Data Acquisition and Preprocessing

Our motion capture data is recorded by using Optitrack [Opt18] and Captury [Cap]. Our database contains 10 distinctive elementary actions performed in different assembly workshop scenarios. We cannot record our motion data in real assembly workshops due to technical difficulties, for instance, occlusions and electromagnetic interference. Hence, our data is recorded in a simulated laboratory environment for better quality and controllability. Table 4.1 shows the details of recorded actions used in our work. The recorded motion data from different capturing systems have different skeletons. We retarget all the motion data to the game engine skeleton from MakeHuman [MAK]. The game engine skeleton contains 19 joints. We use the same parameterization method as Section 3.3.2. The root translation is represented by (x, y, z) vector and relative joint rotation is represented by unit quaternions. Finally. we get a vector of 79 dimensions for each frame.

Motion Data Preprocessing. Figure 4.4 illustrates the detailed steps for data preprocessing. The long input recordings are decomposed into small clips by defining

TABLE 4.1: Motion capture data quantity.

Action Type	No. samples	No. motion primitives
Walk	1154	6
Carry	2481	6
Pick	859	6
Place	512	6
Retract	678	2
Side step	466	2
Look at	90	1
Transfer	82	1
Insert	692	3
Screw (pistol driver)	1304	3

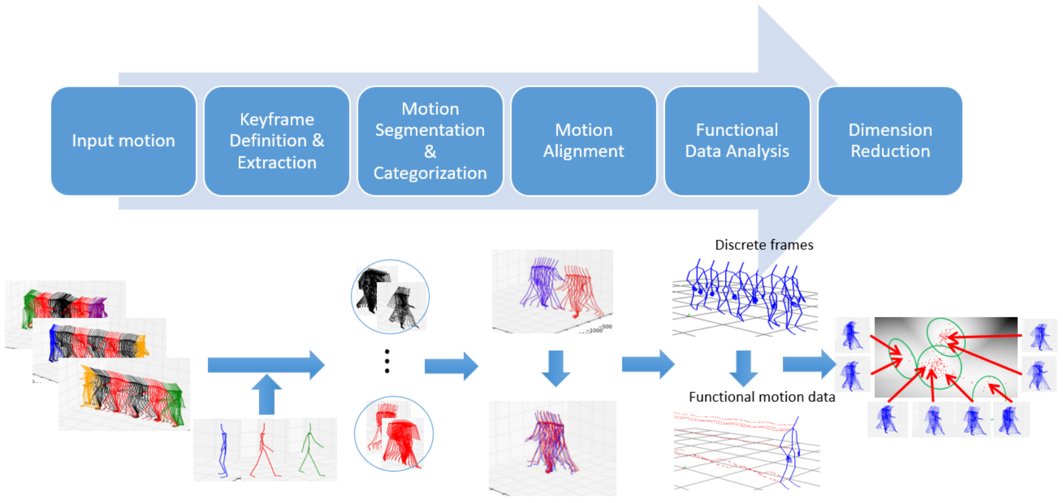


FIGURE 4.4: The pipeline of motion data preprocessing.

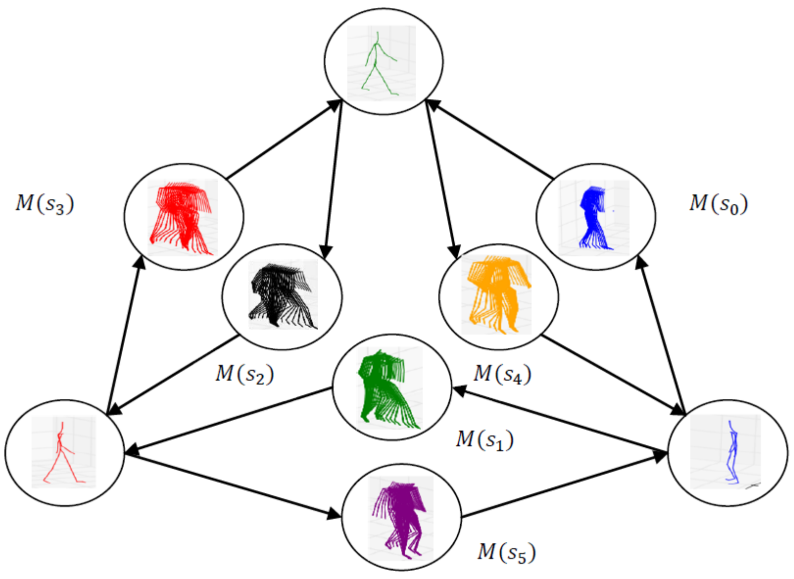


FIGURE 4.5: Constructing motion primitives $M(s_i)$ for walking using keyframes.

keyframes in the motion sequences and cutting the long motions based on those keyframes. We define the keyframes as frame instances when a contact state transition occurs. For example, during walking, foot contact with the ground alternatively changes (e.g. left foot strike). The decoupled motion clips that share the same starting and ending keyframes are categorized into the same motion primitive (Figure 4.5). Hence, the unstructured motion data is converted to structurally similar, contact-consistent motion clips. A good decomposition should make the dissimilarity very small for motion segments within the same motion primitive, and very large for motion segments in different motion primitives. Motion chips that are within the same motion primitive, could have different root positions, orientations, and number of frames. We rotate motion clips in the same motion primitive about the vertical axis and translate them on the ground to make sure they have the same starting position and orientation. The aligned frame sequences are represented as functional data with the same number of basis functions. A low-dimensional space is constructed using SFPCA (see Section 3.3).

4.4 Motion Modeling

Our Morphable Graph framework is a graph-based generative model. The directed graph is manually designed and represents the high-level variations. Figure 4.6 shows part of the whole Morphable Graph used in our work. Each node in the graph is a statistical model called Motion Primitive. It represents a class of structurally and semantically similar motion clips. Each edge in the graph indicates the potential transition between motion primitives.

4.4.1 Learning Statistical Models

For modeling motion primitives, , we apply GMM to learn the distribution of motion clips in low-dimensional space. For the temporal variations, we use dynamic time warping [MR81] to record the temporal variations. Every motion clip in one motion primitive is warped to a canonical timeline. The warping function, which is a monotonically increasing function, represents the temporal variations. We use the median-length motion clip as a canonical timeline. The warping function is represented as a spline $\mathbf{f}(t)$ with the same basis functions as spatial data. We concatenate spatial variations $\mathbf{Y}(t)$ and temporal variations $\mathbf{f}(t)$ together as $\mathbf{Y}'(t) = \{\mathbf{p}(t), \mathbf{q_1}(t), ..., \mathbf{q_m}(t), \mathbf{f}(t)\}$. A latent space $\mathbf{s}$ is learned using SFPCA. Each motion clip $\mathbf{y'}_i(t)$ is projected as a low-dimensional vector $\mathbf{s}_i$. The probability $p(\mathbf{s}_i)$ of motion clip s_i in motion primitive $\mathbf{M}$ is:

$$p(\mathbf{s}_i) = \sum_{k=1}^{K} w_k N(\mathbf{s}_i | \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k) \tag{4.1}$$

where the parameter w_k is the weight of each Gaussian. $\boldsymbol{\mu}_k$ and $\boldsymbol{\Sigma}_k$ define the shape of each Gaussian. These parameters are estimated using Expectation Maximization algorithm [DLR77]. The optimal number of Gaussian K is automatically determined by using Bayesian Information Criterion (BIC) [Sch78]. BIC score has better performance in penalizing the complexity of the model than other model selection methods, such as Akaike Information Criterion (AIC) and cross validation [Wea99]. This is important for our work because good interpolation and extrapolation quality are required for motion synthesis in latent space.

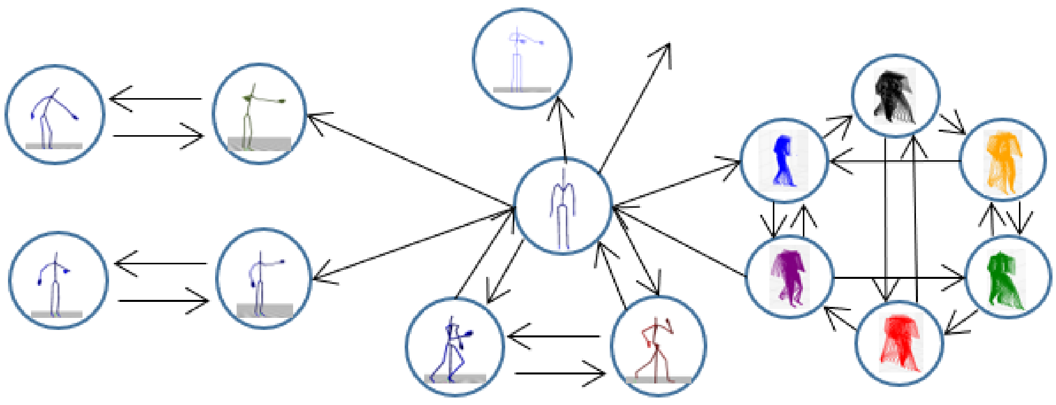


FIGURE 4.6: Part of the Morphable Graph

4.4.2 Space Partitioning Tree

We notice that the stability and quality of synthesized motions using optimization largely rely on the initial input for optimization. If the random sample from motion primitive is far away from the target constraints, sometimes it cannot produce the natural-looking motion that satisfied the user constraints exactly. Therefore, if we can provide a high-quality initial guess that is close to the target constraints, the optimization can converge fast and the resultant motion is anticipated to be of good quality. In order to support fast motion synthesis, we discretize the statistical models and build a space partitioning data structure for each motion primitive in discretized space for efficient searching and optimization. Our motivation is to accelerate the motion synthesis by quickly discarding a part of the latent space based on the observation that samples close in the latent space are also close in the observation space. 10000 samples are generated for each motion primitive to form a dense latent space and we apply the k-means algorithm recursively on the latent space. Figure 4.7 shows an example of our space partitioning tree for picking. The cluster in latent space is corresponding to the cluster of structurally and semantically similar motion clips in observation space. The latent samples are stored in the leaf nodes. For other nodes in the tree, they represent their child nodes using the mean latent motion space parameter vector of the cluster. The closest sample can be efficiently found by comparing the mean of each cluster than going through all the samples.

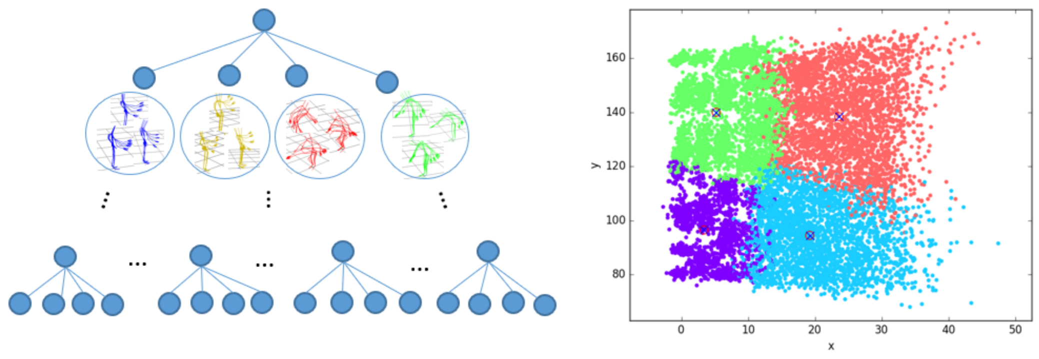


FIGURE 4.7: Space partitioning tree for picking. Four subdivisions are used for each level. Left: Tree structure. The samples in each node are close in observation space. Right: The samples in each node are close in latent space.

4.5 Motion Synthesis

The motion synthesis module consists of finding an optimal graph walk in the Morphable Graph and the optimal motion parameters for each step in order to increase the likelihood in the transition models of the whole motion sequence and reduce errors based on geometric and/or time constraints.

4.5.1 Graph Walk Generation

Our motion synthesis module supports two types of constraints: trajectory and keyframe constraints. High-level user input for instance CNL will be translated into trajectory constraints and keyframe constraints. Trajectory constraints define the position of a joint during an entire elementary action. Keyframe constraints allow constraining the position or orientation of a joint on one frame of the motion. Furthermore, they can be used to constrain the time on which the constraint must be reached. For the definition of keyframe constraints, we semantically annotate the canonical timeline of motion primitives with semantic labels such as start contact and find the target frame in the runtime that meets the semantic labels associated with the keyframe constraint.

The motion synthesis algorithm takes constraints $\mathbf{C}$ on elementary action level and breaks them down to motion primitive level $\boldsymbol{C} = \{\boldsymbol{c}_1, \boldsymbol{c}_2, ..., \boldsymbol{c}_n\}$. The breakdown uses edges in Morphable Graph and starts from the motion primitive that is labeled as start nodes. The graph walk is created heuristically by traversing the transition between motion primitives and ending at the motion primitives which are labeled as end nodes. We iteratively enumerate all optional transitions from the current motion primitive looking at a fixed number of steps T ahead. We use the mean motion of each motion primitive to quickly evaluate the error to constraints using Equation 4.2 as an objective function and select the best next motion primitive with minimal error. An overview of the sequential motion synthesis algorithm is shown in Algorithm 1.

Algorithm 1: Graph Walk Generation

```
Input  : user constraints
Output: a sequence of motion primitives
for (action, constraints) in elementary action list do
    while state is not the end of action do
        transition to new state
        generate state constraints from action constraints
        generate latent parameters for current state
        optimize latent parameters
        add current state with optimized parameters to graph walk
    end
    optimize all steps of current elementary action;
end
```

For motions that follow trajectory constraints on the hip joint, such as walking and carrying, we have to generate constraints for each step until the end of the trajectory has been reached. The constraints of individual steps consist of the position and orientation of the projected hip joint at the end of each step and are generated based on a heuristic using the median step length of the motion primitive. Due to structural differences in the motion, we have to separate motion primitives such as a sidestep and a standard step. In order to choose the appropriate motion primitive at run-time, the path-following algorithm can evaluate multiple options per step to select the best option given the current step constraints.

After the generation of the graph walk with optimal low dimensional parameters, the low dimensional parameters of the graph walk are back-projected into motion splines, concatenated, and discretized into frames for visualization. During this discretization, we generate an annotation of the keyframes that are used in the simulator for scene manipulation.

4.5.2 Space Partitioning Search

During motion generation, the space partitioning tree is traversed to find the optimal sample $\boldsymbol{s}$ based on an objective function that evaluates spatial constraints on joints at one or more frames of the motion primitive. The objective function is shown in Equation 4.2.

$$O(\boldsymbol{s},\boldsymbol{c}) = \sum_{i=0}^{N} \sqrt{(f_k(M(\boldsymbol{s})) - \boldsymbol{c}_i)^2} \tag{4.2}$$

where $\boldsymbol{s}$ denotes the motion vector represented in latent space, $\boldsymbol{s}$ denotes a set of constraints on the joints of the motion at the last frame of the canonical timeline, N is the number of constraints, f_k is the forward kinematics and $M(\boldsymbol{s})$ is the projection from latent space to motion space.

By utilizing this approach, the number of evaluations is reduced from $O(N)$ to $O(l \times k \times c)$ where l is the number of levels, k the number of subdivisions and c is the number of candidates. The search result is a low dimensional vector s that is back-projected into a warped motion spline as described in Section 3.3.

4.5.3 Constrained Motion Synthesis

In this section, we describe our best-fit simulation strategy to generate constrained motions. We use a list of actions with constraints on joints of the skeleton as input for the motion synthesis algorithm, e.g. walk, pick, carry, and place. The constraints **c** can be either defined in a graphical user interface or generated via a text-based CNL user interface in combination with an annotated 3D scene.

The list of actions is sequentially converted into a complete motion by generating a graph walk. The graph walk has two parts: The sequence of motion primitives $\{\boldsymbol{s}_1, \boldsymbol{s}_2, ..., \boldsymbol{s}_n\}$ and the corresponding constraints $\{\boldsymbol{c}_1, \boldsymbol{c}_2, ..., \boldsymbol{c}_n\}$. In our work, we formulate the constrained motion synthesis problem in a Maximum A Posteriori (MAP) framework:

$$\arg \max_{\{\boldsymbol{s}_1, \boldsymbol{s}_2, ..., \boldsymbol{s}_n\}} p(\boldsymbol{s}_1, \boldsymbol{s}_2, ..., \boldsymbol{s}_n | \boldsymbol{c}_1, \boldsymbol{c}_2, ..., \boldsymbol{c}_n) \tag{4.3}$$

According to chain rule, the joint distribution can be decomposed as:

$$\arg\max_{\boldsymbol{s}} p(\boldsymbol{s}_1|\boldsymbol{c}) \prod_{i=2}^{N} p(\boldsymbol{s}_i|\boldsymbol{s}_{i-1}, \boldsymbol{c}) \tag{4.4}$$

where $\boldsymbol{s}$ denotes the sequence of motion primitives $\{\boldsymbol{s}_1, \boldsymbol{s}_2, ..., \boldsymbol{s}_n\}$ and $\boldsymbol{c}$ denotes the constraints $\{\boldsymbol{c}_1, \boldsymbol{c}_2, ..., \boldsymbol{c}_n\}$. Since the constraints in our case are independent in each motion primitive, the Eq. 4.4 can be simplified as:

$$\arg\max_{\boldsymbol{s}} p(\boldsymbol{s}_1|\boldsymbol{c}_1) \prod_{i=2}^{N} p(\boldsymbol{s}_i|\boldsymbol{s}_{i-1}, \boldsymbol{c}_i) \tag{4.5}$$

We find that for a smooth transition between motion primitives $p(\boldsymbol{s}_i|\boldsymbol{s}_{i-1})$, the discontinuities of frames usually cause the major artifacts. Therefore, we propose to model the transition as keyframe constraints. There is no necessity to model the transition distribution additionally. In practice, different actions are usually recorded in separate recordings. Some actions might not have enough smooth transition examples for training a good statistical model. Our approach can put motion synthesis in a more general framework and better reuse the captured data. After the simplification, the final motion synthesis task can be formulated as:

$$\arg\max_{\boldsymbol{s}} \prod_{i=1}^{N} p(\boldsymbol{s}_i|\boldsymbol{c}_i) \tag{4.6}$$

where the constraints $\boldsymbol{c}_i$ contain additional keyframe constraint from previous state $\boldsymbol{s}_{i-1}$ if the state is not the starting state.

According to Bayes' theorem, the posteriori distribution can be decomposed as:

$$p(\boldsymbol{s}_i|\boldsymbol{c}_i) \propto p(\boldsymbol{c}_i|\boldsymbol{s}_i)p(\boldsymbol{s}_i) \tag{4.7}$$

The conditional probability for control term can be modeled as a Gaussian distribution with the mean squared error objective function Eq. 4.2 and a standard deviation of σ_c:

$$p(\boldsymbol{c}_i|\boldsymbol{s}_i) \propto exp(\frac{-||f(M(\boldsymbol{s}_i)) - \boldsymbol{c}_i||}{2\sigma_c^2}) \tag{4.8}$$

Maximizing the likelihood $p(\boldsymbol{s}|\boldsymbol{c})$ is equal to minimizing the negative log likelihood:

$$\arg\min_{\boldsymbol{s}} ln(p(\boldsymbol{s}|\boldsymbol{c})) = \sum_{i=1}^{N} (O(\boldsymbol{s}_i, \boldsymbol{c}_i) - ln(p(\boldsymbol{s}_i))) \tag{4.9}$$

where $p(\boldsymbol{s}_i)$ is modeled by GMM, $O(\boldsymbol{s}_i, \boldsymbol{c}_i)$ is our objective function defined in Eq. 4.2. The optimal value for each motion primitive $\boldsymbol{s}_i$ is found by optimizing the objective function Eq. 4.9 numerically using Levenberg-Marquardt algorithm [Mar63]. In the optimization, we do not use random samples as the initial guess. We use our space partitioning tree described in Section 4.5.2 to find a closer candidate as the initial guess. Our approach can make the optimization converges rapidly and produce more stable results.

4.6 Experiments

In this section, we demonstrate some of the results of our system. We first study the performance of our statistical motion primitive models in terms of motion diversity and model compactness. We then show the effectiveness of our proposed space partition tree for constrained motion synthesis. Finally, we present some qualitative results to show the ease of use and extensibility of our system for different assembly tasks. Our morphable graph framework has been widely tested and deployed in the FP7 EU project INTERACT [INT].

4.6.1 Motion Primitive Evaluation

The statistical motion primitive models are the foundation of our system. In order to test the usability of our work, we apply our approach in different pilot cases. 10 distinctive elementary actions performed in the assembly workshop are included in our system. Figure 4.8 shows some of the motion primitives modeled in our work. Our experimental results demonstrate that the motion primitive models can provide a compact representation for motion variations. Any number of new samples can be generated from the models and converge the space of example motions well.

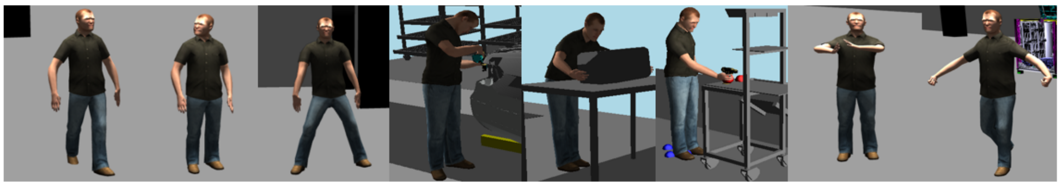


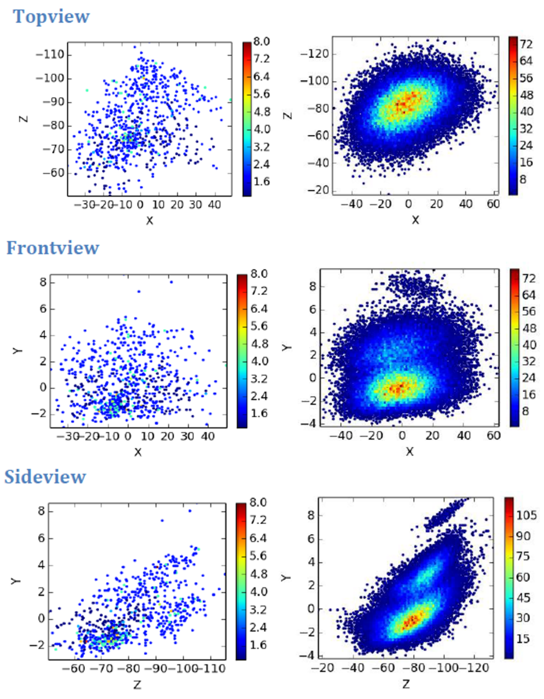
FIGURE 4.8: Generated motion clips from our statistical motion database. From left to right: walking leftStance, looking around, walking sidestep, screwing, two-hand placing, right-hand picking, two-hand transferring and two-hand carrying.

Motion Diversity

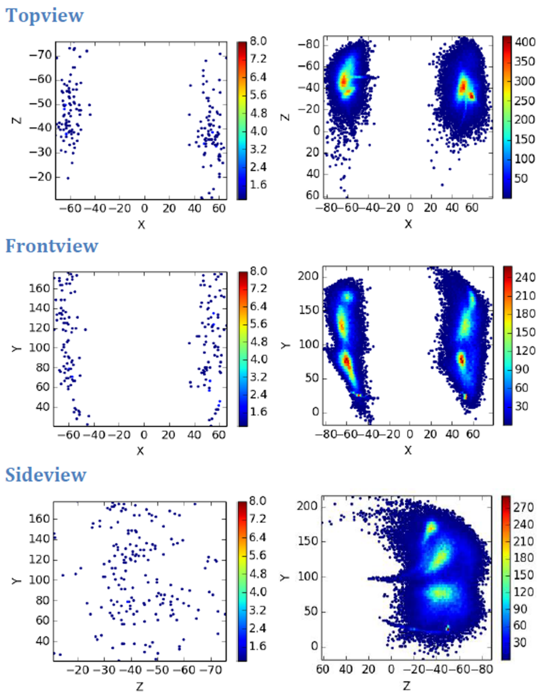
For constrained motion synthesis, it is important to give the constraints in a reachable area. If the targets are given outside of the reachable area, the quality of generated motions is usually poor. The diversity of generated motions from the statistical model defines a reachable area. We sample a large number of random motions from each motion primitive and plot the joint position for some important joints, for example, hands for picking and placing, in Euclidean space. Figure 4.9 shows 60000 random motions generated from two motion primitives: Walking leftStance and two-hand picking. The density distribution learned from random samples demonstrates that our motion primitives can well cover the variations from training data. Figure 4.10 demonstrates our motion synthesis results for different constraints. Given any targets within the valid variation range, our system can generate natural-looking motions to reach the targets.

Compactness

The size of the parametric statistical model is invariant to the number of samples. Our system does not require storing any motion capture data for the task of motion synthesis. Once the model is trained, it can serve as a compact representation of the variations contained in the recorded motions. In our work, we construct 36 motion primitives for 10 elementary actions. The total size of 4.2 GB of motion data is recorded for modeling. Our motion primitive models only require 41 MB, which is 100 times smaller.



(A) The kernel density estimation plot for left foot of walking leftStance. The joint position is calculated from the last frame of generated motion clip. The 3D points are projected to 2D from different perspectives. Left: motion capture data. Right: 60000 synthetic data.



(B) The kernel density estimation plot for left and right hands of two-hand picking. The joint position is calculated from the last frame of generated motion clip. The 3D points are projected to 2D from different perspectives. Left: motion capture data. Right: 60000 synthetic data.

FIGURE 4.9: The evaluation of motion diversity. Two types of motions: Locomotion and Operation are analyzed. A color-coded density is plotted using the R library 'MASS'.

4.6.2 Space Partitioning Evaluation

In our work, we formulate the constrained motion synthesis task as an optimization problem. A good initialization is critical to numerical optimization to reach the global minimum efficiently and stably. We construct a space partitioning tree using k-Means to find the optimal initial guess for optimization. In order to evaluate the efficiency and accuracy of our method, we compare our k-Means tree with the other two popular space partitioning methods: GMM tree and Median tree. GMM tree is constructed using GMM to cluster samples iteratively. The Median tree splits the data according to the median of the dimension with the maximum variance. We also include random initialization in our evaluation. 100 random samples are used

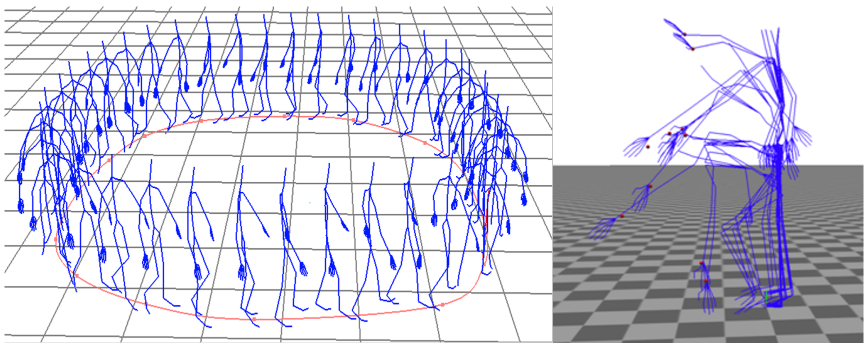


FIGURE 4.10: Constrained motion generation using Morphable Graph. Left: Path following for locomotion. Right: Reaching different targets (right dots) using right-hand picking motion models.

in parallel. In addition, we use brute-force search to find the closest sample to input constraints in the discretized latent space as the ground truth for our evaluation.

We conduct our test on four motion primitives: walk leftStance, right-hand pick reach, right-hand place reach, and screw reach. These are the most commonly performed actions in assembly workshops. Additionally, we design a synthetic picking motion primitive by sampling our statistical motion primitive model. A dense and equally distributed grid is used as the target position for picking (Figure 4.11). The goal is to test motion synthesis approaches in a large and equally distributed dataset.

For the evaluation with the motion models, we generate 1000 random position and orientation constraints for one joint and one keyframe of each motion primitive by sampling from the motion models. Additionally, we evaluate 1000 random pose constraints for each motion primitive that constrain the hip, both shoulders, both hands, and both feet of one frame. The position constraints are defined by a point in the global coordinate system. The orientation constraints are defined by a quaternion representing a rotation in the global coordinate system. The distance for the position constraint is measured using the Euclidean distance. In the case of the pose constraints, the distance is divided by the number of constrained joints. The distance between orientations is measured by the angle between reference vectors rotated by the global orientation of the joint. The constrained joints were selected based on the use case of the simulation of manual assembly workers.

Figure 4.12 shows the results of the experiment that compares the k-Means tree search with different space partitioning methods and random sampling. A brute force search is used as ground truth in the experiments. We measure the average

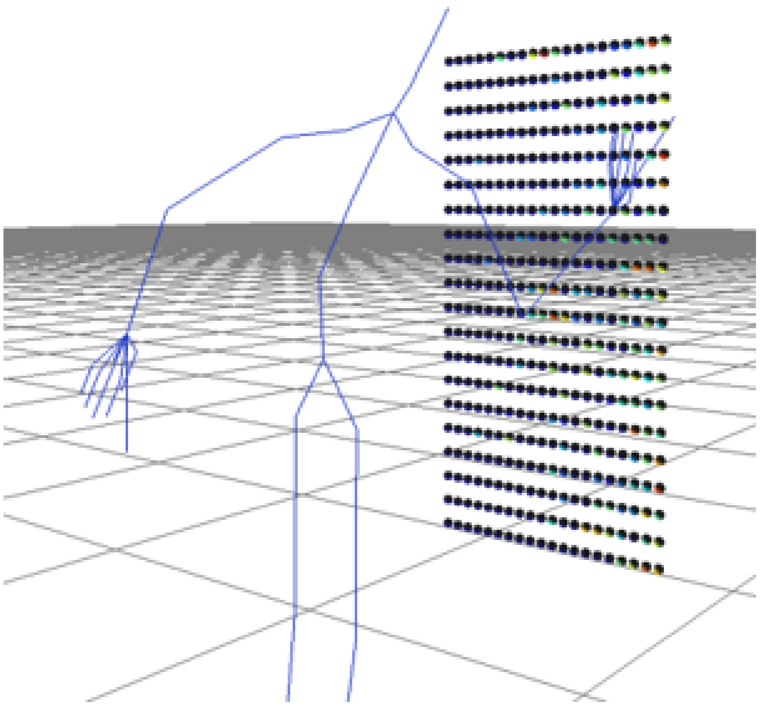


FIGURE 4.11: Synthetic picking motion generation using an equal-distributed, dense grid as the target positions for picking.

distance from generated motions to the constraints and the average number of samples evaluated in each test. The results show that the k-Means tree search reduces the median error for all evaluated motion primitives in comparison with random sampling by more than half while requiring less than 120 evaluations in most cases. For position constraints, the search result also reaches close to the ground truth, which was estimated by the comparison of the 10.000 samples used to construct the tree. The search in the k-Means tree also results in an error close to the ground truth for the pose constraint, which constrains ten joints at once. This shows that the search in the same k-Means tree supports different spatial constraints. However, the results of the experiments also show that the accuracy of our method differs depending on the motion primitives and constraint types. The accuracy of the search in the GMM tree is close to the accuracy of the search in the k-Means tree, however, using k-Means the median and the standard deviation are slightly lower for all tested motion primitives. Contrary to that, the result using the median split strategy is closer to random sampling than to k-Means.

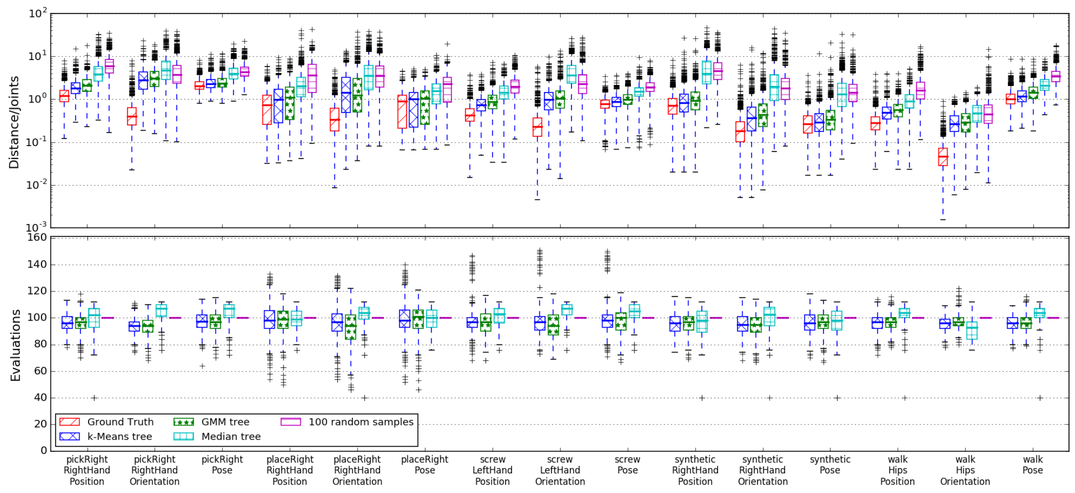


FIGURE 4.12: The search outcome for position, orientation, and pose constraints are compared between space partitioning trees with random sampling and brute force search. The trees are generated using 10,000 samples in the latent feature space. Each box depicts two quantiles of the search outcomes for 1,000 random constraints, while the bold dashes represent the median values of the search results. [Her+17]

4.6.3 Constrained Motion Synthesis

Our system has been applied and evaluated in some digital assembly workshop scenarios. In this section, we discuss how to use our system to generate work task simulation for virtual characters in a digital assembly workshop.

Ease of Use Our 3D scenes are set up using XML3D [Son+10] in order to support directly using from a web browser (chrome) without any installation or dependencies. The 3D objects in the scene are stored and can be queried from the scene knowledge base. A task can be easily defined using CNL from a text-based interface (Fig. 4.2). For example, Fig. 4.13 shows the simulation result for the task description "carry the middle console from trolley to work table". Users do not need to specify the exact walking path or how to pick and place the middle console. Once the actions and the target objects are given, the position of the target objects in the scene will be queried and the trajectories will be automatically generated. In order to increase the variations, our system processes the conditions in a less constrained manner. Instead of using a dense curve as the trajectory, we use the target position and orientation as the goal for each motion primitive. In Fig. 4.13, the blue points on the ground are the trajectory constraints. During the motion synthesis phase, if the system uses random samples as initialization, the optimal result for each motion primitive could be different for different simulations. It can generate diverse animations given the same constraints.

Extensibility One of the challenges for constrained motion synthesis is to adapt existing action controllers to different types of actions. For instance, the controller for locomotion like walking and carrying could be different from the controller for an operation like screwing and picking. One of the key advantages of our system is that we formulate the controlled motion synthesis in a general framework. We test our system in different scenarios for complex assembly tasks. Figure 4.14a shows a small-scale and complex warehouse pilot case. The task for the virtual character is to take the target tools from the shell and put them in the correct trolley or vice versa. Transfer action is included in our Morphable Graph based on the observation that sometimes people like to transfer objects from right hand to left hand and then pick the next one. Figure 4.14b shows a large-scale automotive assembly workshop. The virtual character takes the screwdriver and screws from the trolley and screws them into the correct place on the car. The screw action can be modeled and generated using target constraints. The experimental results show that our system can add new actions in a general manner and produce new variant motions with competitive quality to captured data. In addition, the statistical motion models provide a general framework for integrating new constraints by using the maximum a posterior (MAP) method.

4.7 Discussion

Our motion primitive modeling pipeline is not fully automatic. The keyframes need to be defined and annotated from frame sequences manually. We explore some rule-based frame recognition approaches that detect some kinds of signals such as foot contact on the ground to automatically search keyframes and segment raw capture data. However, some keyframes are semantically similar but quite different in geometry, especially for complex actions like screw, and transfer. So for extracting key frames from complex actions, manual work is still required. It is very time-consuming due to the large number of samples in the motion data required for modeling. In order to make our motion synthesis prototype more practical for industry usage, deep learning-based human motion recognition approaches could be a promising option to investigate (e.g. [Che+19]). The goal would be to extract structurally and semantically similar key frames from recorded motion sequences, so the motion primitive construction can be fully automatic.

Individual motion primitives can inherently generate realistic motions inside of the range of the training data. If the target constraints are out of the reachable area, the optimization could produce unnatural motions. Even though the model likelihood item in Eq. 4.9 provides a penalty for unnaturalness, some "funny-looking" motions can still be generated if the targets are too far away and the error overwhelms the objective function. In order to make sure that the generated motions are high-quality, it is important to provide appropriate constraints for each motion primitive in the graph walk. However, the variation of the target joint is unknown for the procedure of constraint decomposition. The motion diversity analysis in Section 4.6.1 can be provided as hyperparameters for constraint decomposition. We also notice that the point cloud density distribution of some joints is not continuous. There are holes or gaps between clusters. This may indicate that the captured data has a bias. Our laboratory capturing was manually designed and the randomness in multiple repetitions could be dominated by the individual actors' own habitual tendencies. Thereby, the captured data could not be natural. This could produce a gap between the end user's expectations and the model variation. We believe that the motion models can be improved by using the data captured from the real assembly workshop, which could represent the natural distribution of the motion better. In addition, it is not clear how many examples would be enough to define a dense distribution for our motion primitives.

4.8 Conclusion

The Morphable Graph framework constructed in INTERACT provides a compact representation for the variation in motion capture data. A statistical human motion modeling pipeline is designed and implemented. Using this pipeline, a statistical motion database that contains 10 elementary actions is created. The database is compact compared to the original motion capture data, requiring only 42MB for memory. However, the original motion capture data takes 4.2 GB for storage. Any number of motion variations with different styles can be efficiently generated by simply sampling the motion primitive models we construct.

Given a list of constraints for different elementary actions, such as walking and picking, a constrained motion can be synthesized. The controlled motion synthesis algorithm breaks down the constraints to the motion primitive level to generate a sequence of motion primitives. The latent model parameters of each motion primitives are first optimized individually to fit the constraints. For path following motions, multiple steps are additionally optimized together. The optimization of individual steps is accelerated using space partitioning data structures in the latent space of the statistical motion models. Our system is successfully tested in practice using motion capture data that is recorded for manual assembly tasks.

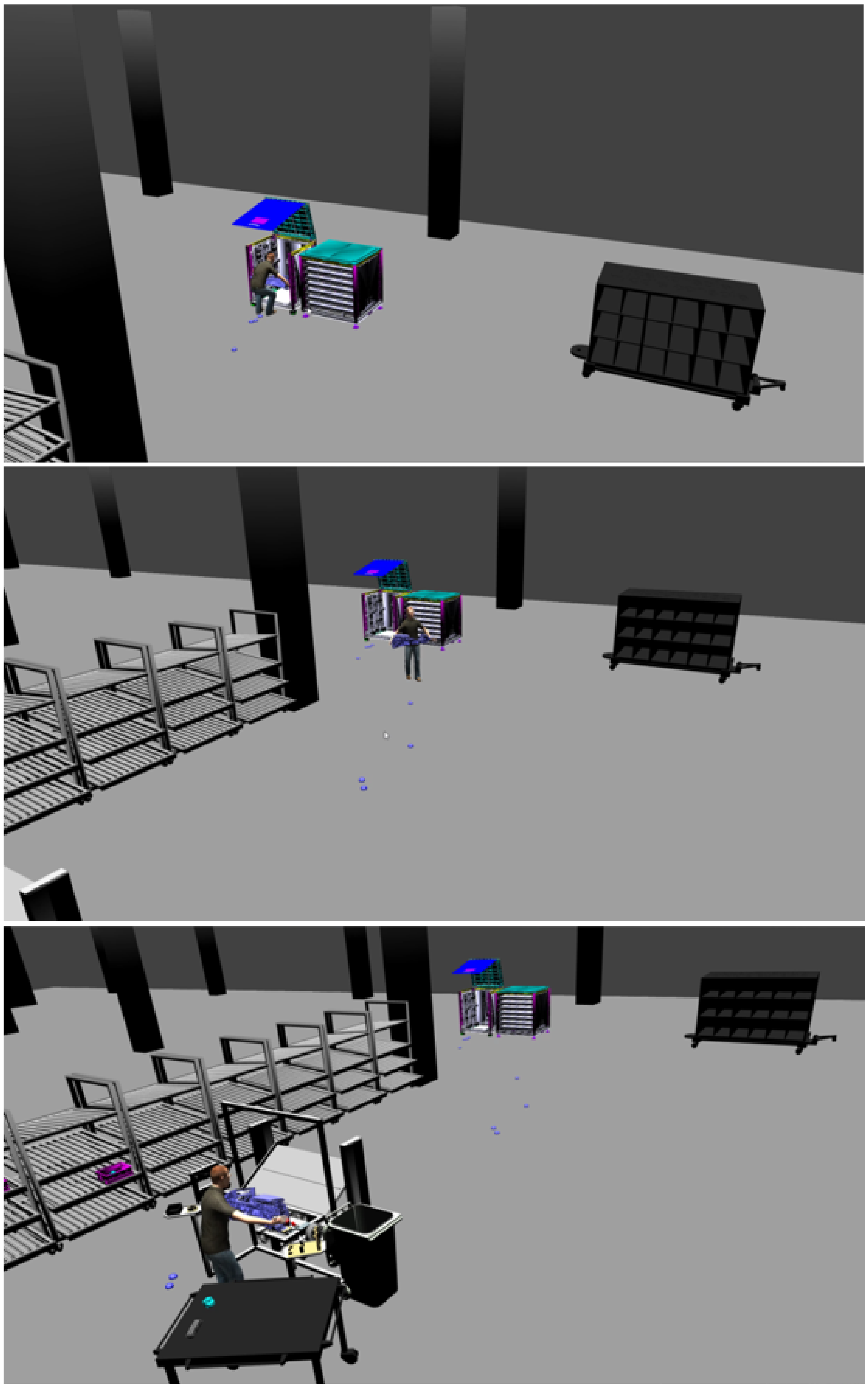
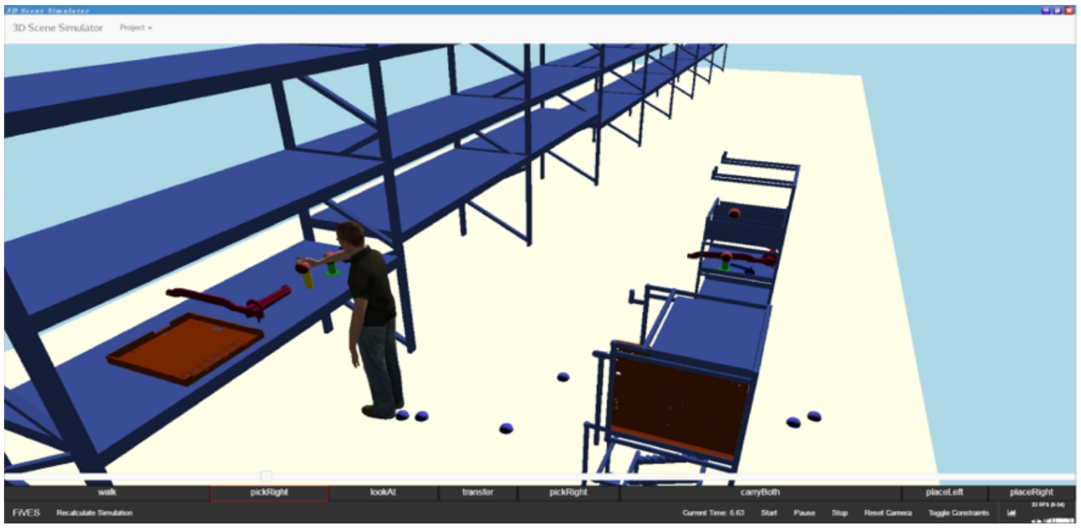
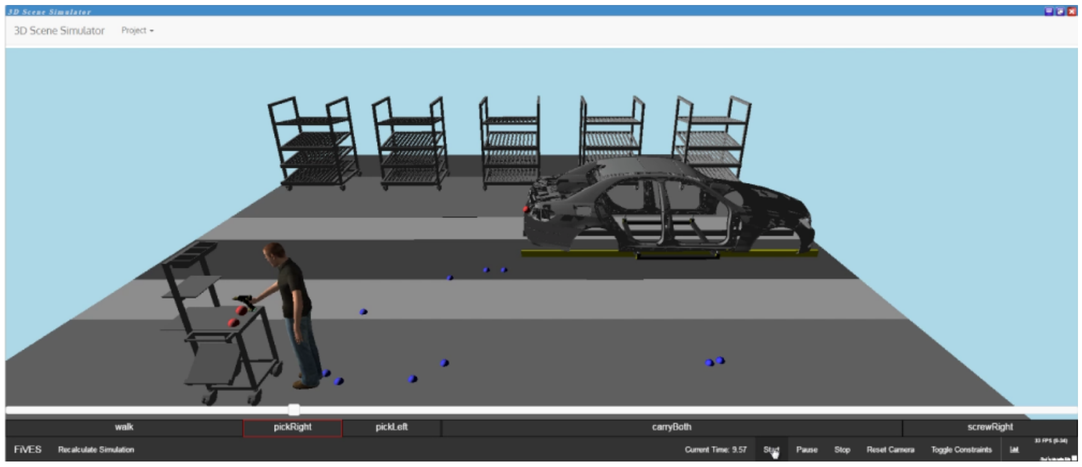


FIGURE 4.13: A simulation result for the task "Carry the middle console from trolley to work table".



(A) Warehouse pilot case



(B) Automotive assembly workshop pilot case

FIGURE 4.14: The simulation results using our system for different virtual scenarios.

Chapter 5

Modeling Motion Style using Variational Autoencoder

In the previous chapters, we discuss our efforts in modeling the variations of human motions using the canonical generative model. Although our Morphable Graph framework can generate controlled motions with rich variations, the motion style is not included. Constructing style models is a challenging task due to the lack of example motions. The style or personality of human motion is a very broad concept. It can be the differences in aging, gender, mood, and so on. In our work, we define motion variation such as repeating the same action differently multiple times as within-class variations, and the style which is shared in the same class but different between other classes as between-class variations.

The synthesis of human motion with large variations including styles and personalities has a growing demand for simulation applications such as crowd simulation of pedestrians for autonomous driving. However, data-driven motion synthesis approaches usually require a significant amount of example data for training. It is even more challenging for style modeling because a similar size of data needs to be recorded for each style. In addition, we notice that the variation and quality of synthesized motions highly depend on the size of the training data.

Recently, deep learning achieves great success in computer vision and natural language processing. It also provides us with powerful tools to construct advanced generative models on a more general latent space. We utilize an autoencoder to learn a general latent space for the whole motion database. In addition, we improve our Morphable Graph by replacing Gaussian Mixture Models with variational autoencoder to determine the motion primitives' distribution. The Gaussian Mixture Model is a powerful tool while it is a parametric model and has a strong assumption about the structure of the distribution. One of the advantages to use a variational autoencoder is that it does not have any assumption on the shape of the distribution.

We propose a novel approach to create generative models for distinctive styles of locomotion for humanoid characters. Our approach only requires a single or a few style examples and a neutral motion database. We are inspired by the observation that human styles can be easily distinguished from a few examples. However, learning a generative model for natural human motions that can display huge amounts of variations and randomness would require a lot of training data. Furthermore, it would require considerable effort to create such a large motion database for each style. One solution for that is motion style transfer, which provides the possibility of converting the content of the motion from one style to the other. Typically style transfer focuses on transferring the content motion to the target style explicitly. We propose a variational generative model to combine the large variation in the neutral motion database and style information from a limited number of examples. We formulate the style motion modeling as a conditional distribution learning problem and style transfer is implicitly applied during the model learning process. A conditional variational autoencoder (CVAE) is applied to learn the distribution and stylistic examples are used as constraints. We demonstrate that our approach can generate any number of natural-looking, various human motions with a similar style to the target. The work presented here is based on previous publications [Du+19a; Du+19b].

5.1 Introduction

Constructing a representative data-driven motion model usually needs a lot of motion capture data. For instance, for pedestrian simulation, lots of repetitions of walking are necessary to cover different trajectories and speeds. In addition, capturing multiple characters is also required to contain different styles. As a result, data-driven approaches still face the challenge of effectively reusing recorded motion data for diverse scenarios [MC12; WN15].

Many previous approaches try to learn the distribution of human motion from motion capture data. Statistical modeling methods such as Hidden Markow Model (HMM) [BH00a; Bow00], Gaussian Mixture Model (GMM) [Lee+02; MC12; Du+16b] have been used to model the distribution of motion capture data. Nevertheless, utilizing parametric models mandates a robust assumption regarding the nature of the distribution.. Recently, generative models from deep learning demonstrate state-of-the-art performance in image processing and speech synthesis [GEB15; JAF16]. Variational autoencoder and other generative models have been effectively utilized to understand the distribution of motion capture data [Hab+17; MHM18]. These techniques are more universal since they do not necessitate any prior knowledge or assumptions about the distribution.

Motion style transfer provides the possibility of creating a synthetic stylized motion database from existing capturing without additional motion capturing efforts. Some special styles are hard to capture. For instance, a drunk walking model would be interesting for pedestrian simulation of autonomous driving. However, capturing a lot of drunk walking is much harder than normal walking. A significant amount of effort has been spent on the problem of style transfer for human motion data. One set of approaches is trying to separate style components from the content of the motion. Brand and Hertzmann [BH00b] applied HMMs to learn a set of style-specific models and a generic model to encode style from motion content and generate new stylistic motions. Urtasun et al. [Urt+04] treated motions as a linear combination of principal components, and extrapolate principal component weights to change the content of motion while keeping the original style. Inspired by the successful work of analyzing the genres of an audio signal in the spectrum domain, Fourier analysis is also employed for motion data. Unuma et al. [UAT95] used Fourier analysis to motion signal and alter the style of motion by manipulating the coefficients of different frequencies. Yumer and Mitra [YM16] observed that the magnitude of the spectrum is more relevant to the style of action and the phase is more relevant to the content. They formulate motion style transfer as an optimization problem and achieve style transfer between different actions in the spectral domain. Our work is inspired by [Hol+17], the Gram matrix is used as the style similarity measure for motion data to train a style transfer network in an unsupervised fashion.

In this work, we present a novel approach to combining statistical motion modeling and style transfer. Figure 5.1 shows an example of our motion modeling and synthesis pipeline for walking. We follow the assumption from [MC12], which is although human motion has infinite variations, the high-level structures (motion primitive) are finite. A motion primitive contains structurally and semantically similar motion clips. For instance, normal walking can be considered as an alternating between left-stance and right-stance. Each left- or right-stance could have infinite variations. Taking this assumption, an action can be modeled by a directed graph. Each node in the graph is a motion primitive and the edges are possible transitions between motion primitives. Each motion primitive is modeled as a generative model. Inspired by [HSK16], we train a single-layer convolutional autoencoder on a large motion database to find a general, representative feature space for motion data. We formulate the style motion primitive modeling as a conditional distribution learning problem. A large amount of neutral motions which contain a rich of variations is used as training data, and the limited number of style examples are taken as constraints. The content and style motions are encoded by a pre-trained convolutional autoencoder and fed to train conditional variational autoencoder (CVAE) to model the conditional distribution.

Our contribution can be summarized as:

- we propose a novel approach to implicitly combine style transfer and statistical motion modeling by using CVAE to formulate the style-conditioned distribution of human motion.
- our approach can produce any number of new stylistic motions with rich variations given a single or a few style examples.

5.2 Motion Data Acquisition

Our motion database is recorded by an OptiTrack system [Opt18] with three male actors. The whole database contains 40 minutes of walking. In addition, we also include walking, running from CMU [Dat18] and HDM05 [Mül+07] motion capture database.

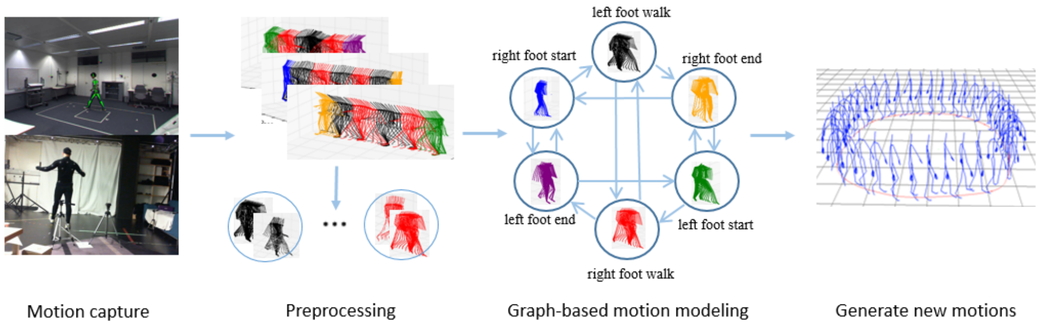


FIGURE 5.1: Motion modeling and synthesis pipeline for walking.

For the style data, we use stylistic walking data from [Xia+15]. A retargeting approach [Mon+00] is implemented to retarget all motion data to Makehuman [MAK] game engine skeleton.

All motions in our motion dataset are parameterized as 3D joint positions. We employ joint position since it is more consistent with visual motion observation, making it an appropriate choice for training purposes. In our work, we use 21 joints for representing the human skeleton. In addition to the joint position, the global speed on the 2D ground and the rotation velocity about the vertical axis is computed and added to each frame. So in our dataset, each frame is represented as a vector of length 66.

All frames are normalized in the way that they have the same global position on the 2D ground and face the same direction. The motion preprocessing is similar to the previous work described in Section 4.3, we decompose the long motion recordings into small motion clips by automatically detecting foot contact frames in the frame sequences. We use the foot contact frames as keyframes to cut the motions and categorize the structurally and semantically similar clips into different motion primitives.

5.3 Statistical Motion Primitive Modeling

Figure 5.2 shows the outline of constructing the statistical model for each motion primitive. Our goal is to learn a conditional distribution $P(\mathbf{X}|\mathbf{x}_s)$ based on a large number of content motion clips $\mathbf{X}$ and a style constraint $\mathbf{x}_s$. If there are no style constraints given, the model simply learns the distribution of content motion clips

$P(\mathbf{X})$ for each motion primitive. If there are style examples, the distribution can be deformed to a new distribution based on different style input $\mathbf{x}_s$. Similar to [HSK16], we use a single layer convolutional autoencoder to encode motions into the feature space $\mathbf{Y} = \Phi(\mathbf{X})$. The feature space provides a continuous space for motion editing. Style is encoded in feature space using the Gram matrix. The distribution of motion clips in feature space is learned by the VAE. For the stylistic model, we formulate the Gram matrix as a style constraint term in the loss function of the VAE. Therefore, a conditional distribution can be learned by training the model.

5.3.1 Style Feature Extraction

Using Gram matrix to extract styles from features produced by convolutional neural networks has achieved great success in image style transformation [GEB15; JAF16]. Holden et al. [Hol+17] adopt this idea to human motion data. In our work, we do not directly work on the motion data. Instead we take similar idea to [HSK16; Hol+17], learning a general, continuous motion manifold from a large set of heterogeneous motion dataset. We construct a single layer convolutional autoencoder to perform 1d convolution on a large set of motion data.

The convolutional encoder is:

$$\mathbf{Y} = \Phi(X) = \Psi(ReLU(\mathbf{X} * \mathbf{W} + \mathbf{b})) \tag{5.1}$$

We use 256 kernels to perform 1D convolution along the time axis. The size of the kernel is 15, which is a quarter of the canonical timeline. $\mathbf{W}$ denotes the kernel weights and $\mathbf{b}$ is the bias. Ψ denotes average pooling, which we find average pooling has good performance in content reconstruction. The convolutional decoder is:

$$\mathbf{X} = \Phi'(Y) = \Psi'(\mathbf{Y} * \mathbf{W}' + \mathbf{b}') \tag{5.2}$$

where $\mathbf{W}'$, $\mathbf{b}'$ have the same shape as $\mathbf{W}$ and $\mathbf{b}$. Ψ' denotes the average depooling layer.

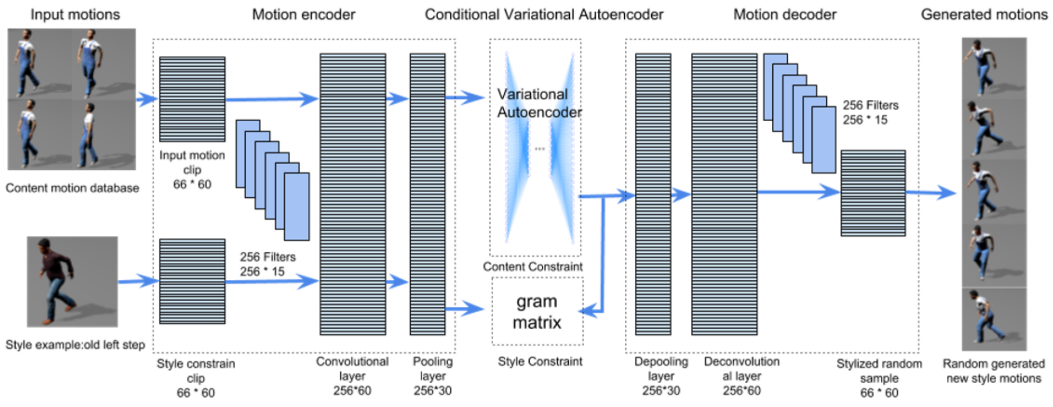

Input motions
Motion encoder
Conditional Variational Autoencoder
Motion decoder
Generated motions
Content motion database
Input motion clip 66 * 60
256 Filters 256 * 15
Variational Autoencoder
Content Constraint
gram matrix
Style example:old left step
Style constrain clip 66 * 60
Convolutional layer 256*60
Pooling layer 256*30
Style Constraint
Depooling layer 256*30
Deconvolutional layer 256*60
256 Filters 256 * 15
Stylized random sample 66 * 60
Random generated new style motions

FIGURE 5.2: The overview of stylistic motion primitive modeling. The input is a large set of content motions with rich variations and a single style example. The input motions are encoded into feature space by motion decoder, which is trained on all motion clips. The conditional variation autoencoder (CVAE) is used to model the conditional distribution. New motion can be generated from the CVAE and back projected to motion space via the motion decoder.

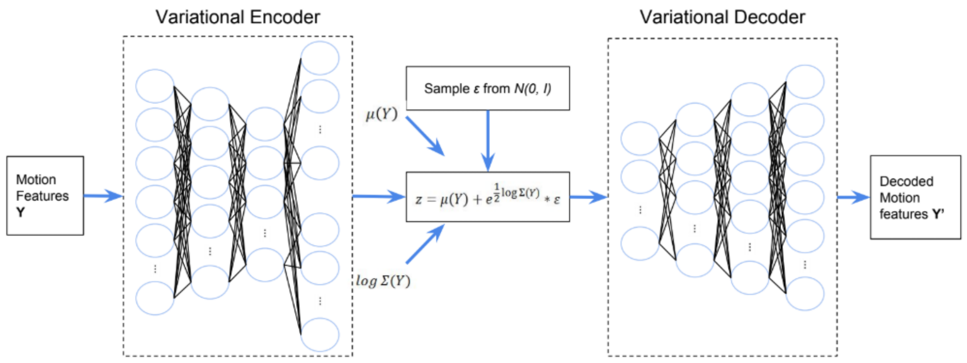


FIGURE 5.3: The architecture of variational autoencoder used in our model.

5.3.2 Variational Autoencoder for Human Motion Model

In this section, we will briefly review variational autoencoder (VAE) proposed by [KW13]. VAE assumes that data $\mathbf{Y}$ can be represented in a low-dimensional latent space $\mathbf{z}$ and the distribution $P(\mathbf{Y})$ can be computed by the integral of the marginal likelihood as:

$$P(\mathbf{Y}) = \int P_\theta(\mathbf{Y}|\mathbf{z})P(\mathbf{z})d\mathbf{z} \tag{5.3}$$

Since the true posterior $P_\theta(\mathbf{Y}|\mathbf{z}) = P(\mathbf{z}|\mathbf{Y})P(\mathbf{z})/P(\mathbf{Y})$ is intractable, [KW13] defines a variational estimator $Q_\phi(\mathbf{z}|\mathbf{Y})$ to approximate $P(\mathbf{z}|\mathbf{Y})$. We want to maximize the log likelihood of the density of data $P(\mathbf{Y})$ while minimizing the difference between $P(\mathbf{z}|\mathbf{Y})$ and $Q_\phi(\mathbf{z}|\mathbf{Y})$, which can be formulated as:

$$\begin{aligned} &log(P(\mathbf{Y})) - D(Q_\phi(\mathbf{z}|\mathbf{Y})||P_\theta(\mathbf{z}|\mathbf{Y})) = \\ &E_z[log(P_\theta(\mathbf{Y}|\mathbf{z}))] - D[Q_\phi(\mathbf{z}|\mathbf{Y})||P(\mathbf{z})] \end{aligned} \tag{5.4}$$

where D is the Kullback-Leibler divergence to measure the dissimilarity between two distributions. The details of the derivation of Eq. (5.4) can be found in [KW13]. The optimal estimation of $P(\mathbf{Y})$ can be achieved by optimizing the right-hand side of Eq. (5.4). $Q_\phi(\mathbf{z}|\mathbf{Y})$ can be regarded as the encoder to encode $\mathbf{Y}$ into $\mathbf{z}$ and $P_\theta(\mathbf{Y}|\mathbf{z})$ can be regarded as the decoder to decode $\mathbf{z}$ back to $\mathbf{Y}$. The latent distribution $P(\mathbf{z})$ is approximated by a multivariate Gaussian with $\mu(\mathbf{Y})$ and diagonal covariance matrix $\Sigma(\mathbf{Y})$, which are deterministic functions learned from the encoder. The sampling of $P(\mathbf{z})$ is achieved by using parameterization trick, which samples a normal distribution $\varepsilon \sim N(0, I)$ and transforms it to $\mathbf{z}$:

$$z = \mu(\mathbf{Y}) + \Sigma^{1/2}(\mathbf{Y}) * \varepsilon; \varepsilon \sim N(0, I) \tag{5.5}$$

Figure 5.3 illustrates the network architecture of VAE used in our work. The encoder and decoder are both modeled by a four-layer feed-forward network. The structure of the encoder and decoder are symmetric except for the last layer of the encoder, which has two branches to produce mean and covariance separately. The hidden units for the encoder are 512, 256, 128, and 32, and for the decoder are 32, 128, 256, and 512. Tanh is used as an activation function for all layers except for the output layer in both the encoder and decoder.

For training the network, we optimize the parameters of encoder and decoder networks via stochastic gradient descent. The loss function $\mathbf{L}$ is defined as follow:

$$\begin{aligned} \mathbf{L} &= \mathbf{L}_{content} + \mathbf{L}_{KL} \\ &= -E_z[log(P_\theta(\mathbf{Y}|\mathbf{z}))] + D[Q_\phi(\mathbf{z}|\mathbf{Y})||P(\mathbf{z})] \end{aligned} \tag{5.6}$$

For the first reconstruction term, $E_\mathbf{z}$ can be approximated by sampling $\mathbf{z}$:

$$\begin{aligned} -E_z[log(P_\theta(\mathbf{Y}|\mathbf{z}))] &= \frac{1}{L}\sum_{l=1}^{L}(\frac{1}{N}\sum_{i=1}^{N} log(e^{-\frac{1}{2}||decoder(\mathbf{z}_{i,l})-\mathbf{y}_i||^2})) \\ &= \frac{1}{2NL}\sum_{i=1}^{N}\sum_{l=1}^{L}||decoder(\mathbf{z}_{i,l})-\mathbf{y}_i||^2 \end{aligned} \tag{5.7}$$

For each $\mathbf{y}_i$, we draw $\mathbf{L}$ samples from $P(\mathbf{z})$ to train the decoder $P_\theta(\mathbf{Y}|\mathbf{z})$. Theoretically, the true distribution can be better approached if more samples are used. But it is also computationally expensive. In our work, we use $\mathbf{L} = 10$ for all our models. $\mathbf{z}_{i,l} = \mu(\mathbf{y}_i) + \Sigma^{1/2}(\mathbf{y}_i) * \varepsilon_l$. $\mathbf{y}_i$ is the encoded features from motion clip $\mathbf{x}_i$.

The KL divergence serves as a regularization term. The analytical solution can be computed since P_θ and Q_ϕ are both multivariate Gaussian.

$$\begin{aligned} D[N(\mu(\mathbf{Y}),\Sigma(\mathbf{Y}))||N(0,I)] &= \\ \frac{1}{2}(\mu(\mathbf{Y})^T\mu(\mathbf{Y}) + tr(\Sigma(\mathbf{Y})) - k - log(det(\Sigma(\mathbf{Y})))) \end{aligned} \tag{5.8}$$

5.3.3 Conditional Variational Autoencoder

Our goal is to combine motion style transfer with statistical modeling of motions. Creating a generative model for a certain style can be formulated as a conditional distribution modeling $P(\mathbf{Y}|\mathbf{y_s})$. The stylistic example $\mathbf{x_s}$ is first encoded into feature space $\mathbf{y_s}$ using convolutional encoder Φ. The style is extracted using the Gram matrix, which is defined as the sum of the inner product of features over the temporal axis.

$$Gram(\mathbf{y}_s) = \sum_i \mathbf{y_{s,i}}\mathbf{y_{s,i}^T} \tag{5.9}$$

As the Gram matrix sums over frames, it does not require frame alignment between content motion and style motion. So for style constraints, single or multiple style examples can be used. The distribution should deform based on different style inputs. This is achieved by adding the style constraint into the loss function of VAE.

$$\mathbf{L}_{style} = \alpha||Gram(decoder(\mathbf{z})) - Gram(\mathbf{y}_s)|| \tag{5.10}$$

where α is the style weight to control how stylistic sampled motions. We empirically set α to 200. Our final loss function for the conditional VAE model is:

$$\mathbf{L} = \mathbf{L}_{content} + \mathbf{L}_{KL} + \mathbf{L}_{style} \tag{5.11}$$

5.4 Experiments

In this section, we give the details about how we train the generative models for each motion primitive and show some of our results with a quantitative analysis of our approach. We evaluate our method on six distinctive styles, which are: depressed, proud, old, childlike, female, and angry.

5.4.1 Model Training

We first train a convolutional autoencoder (motion encoder and decoder in Figure 5.2) to extract the features from motion data. In order to find a general representation for all motion data, we train a convolutional autoencoder on the whole training data, not only in the segmented motion clips. We use an overlapping window to slide over motion data to decompose long motions into equal-size clips. The overlapping window has a size of 60 and is overlapped by half of the size, which is the same size as our canonical timeline (see Section 4.4.1). The convolutional autoencoder is trained on all motion clips with 300 epochs and a training rate of 0.0001 on an NVIDIA GeForce GTX 760. The training takes roughly 10 hours.

For neutral walking and running, we create six motion primitives for each action (Figure 5.1). Each motion primitive is modeled by VAE. We use the pre-trained convolutional autoencoder to encode motion clips into feature space. The motion clips are pre-classified into each motion primitive. Eq. (5.6) is used as a training objective function.

For stylistic motion primitives, motion primitives are trained using conditional variational autoencoder (CVAE). For each motion primitive, we use the same motion clips as the corresponding neutral motion primitive as the content and one style example as the condition. The style example is also encoded using a convolutional autoencoder and serves as a constraint in the objective function Eq. (5.10). The style

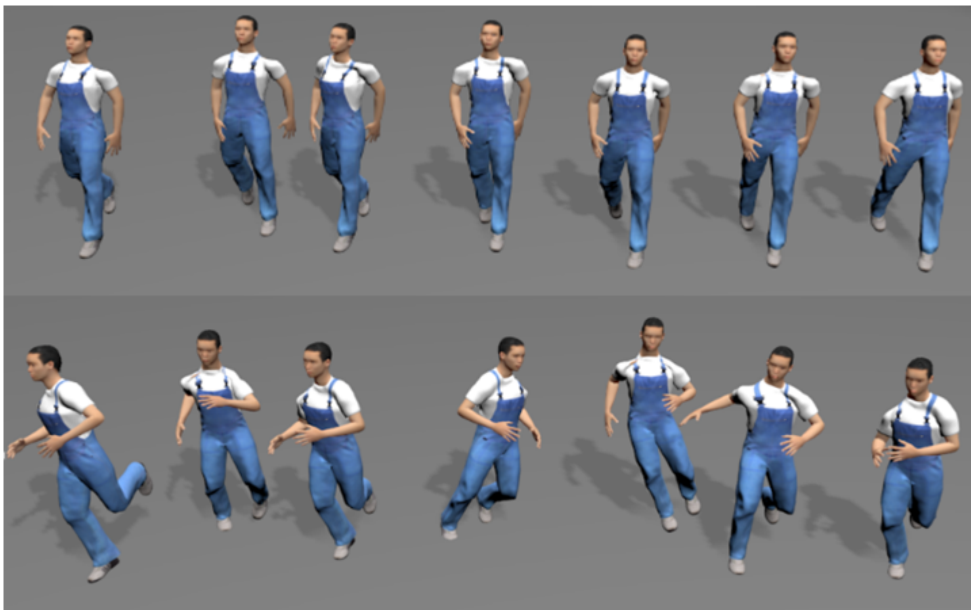


FIGURE 5.4: Random samples from motion primitives trained on a neutral dataset. From top to down: walk left stance, run right stance.

example does not need to have the same size as motion clips. And more style examples can be used as constraints. It means more stylistic frames are provided and will be averaged by the calculation of gram matrix Eq. (5.9).

For VAE and CVAE model training, We set epochs to 1000 with a training rate of 0.00001. The training time depends on the samples in each motion primitive. All networks are implemented using Tensorflow.

5.4.2 Motion Primitive Evaluation

Motion primitives serve as the core of our motion synthesis framework. The quality and variation of generated motions from each motion primitive will decide the quality of the final motion. Figure 5.4 shows some random samples generated from three motion primitives: walk left stance, run right stance, and pick reaching. They are modeled using VAE on a neutral database without style constraints. All characters start in a line with equal spacing. The last frame of each clip is displayed in Figure 5.4. The samples show good variation which is important to satisfy different constraints. We test the modeling ability of VAE on picking to show our approach is general for different kinds of actions, not only for locomotion.

5.4.3 Stylistic Motion Primitive Evaluation

In order to measure whether the samples generated from stylistic models are similar to the target style example or not, we use Eq. (5.12) to measure the similarity distance between motion primitives and style examples.

$$\mathbf{D}(m,s) = \frac{1}{N}\sum_{i}^{N} ||Gram(encoder(x_i) - Gram(encoder(s)))|| \tag{5.12}$$

where x_i is i_{th} sample of model m, s denotes the style example. We encode the samples and the style example using Gram matrix, and measure the mean squared distance between the samples and the style example. We use 1000 samples for each motion primitive. For neutral, we use the mean motion as the style example. Table 5.1 shows the evaluation of seven different style motion primitives for walk left stance. From the diagonal of the confusion matrix, we can see that the samples from each stylistic model are more close to the target style than other styles.

Figure 5.5 shows random samples from six stylistic variants of walk left and right stance. The left columns are the style input. All the style examples are selected to walk in a straight line to minimize the variations between style examples. From the sampling results, we can see that the stylistic models have good variations in both poses and trajectories.

TABLE 5.1: The style similarity distance between different styles and their corresponding models.

	neutral	proud	depressed	angry	female	childlike	old
neutral	**0.0149**	0.1096	**0.0345**	0.2731	**0.0377**	0.1792	0.1712
proud	0.1937	**0.0658**	0.1292	0.0916	0.0818	0.0957	0.1414
depressed	0.0825	0.0883	0.0357	0.1502	0.0448	0.1179	0.1030
angry	0.1769	0.0906	0.1081	**0.0697**	0.0832	0.0959	0.1183
female	0.1694	0.0842	0.1137	0.1156	0.0697	0.1148	0.1457
childlike	0.0977	0.0718	0.0939	0.1405	0.0719	**0.0456**	0.1257
old	0.1728	0.1118	0.0987	0.1344	0.1071	0.1395	**0.0624**

5.4.4 Controlled Motion Synthesis

Our model can generate controlled motion given user control inputs. Once the high-level parameters, such as target trajectory or destination are given, offline motion planning is processed and a graph walk is generated in a graph-based motion model.

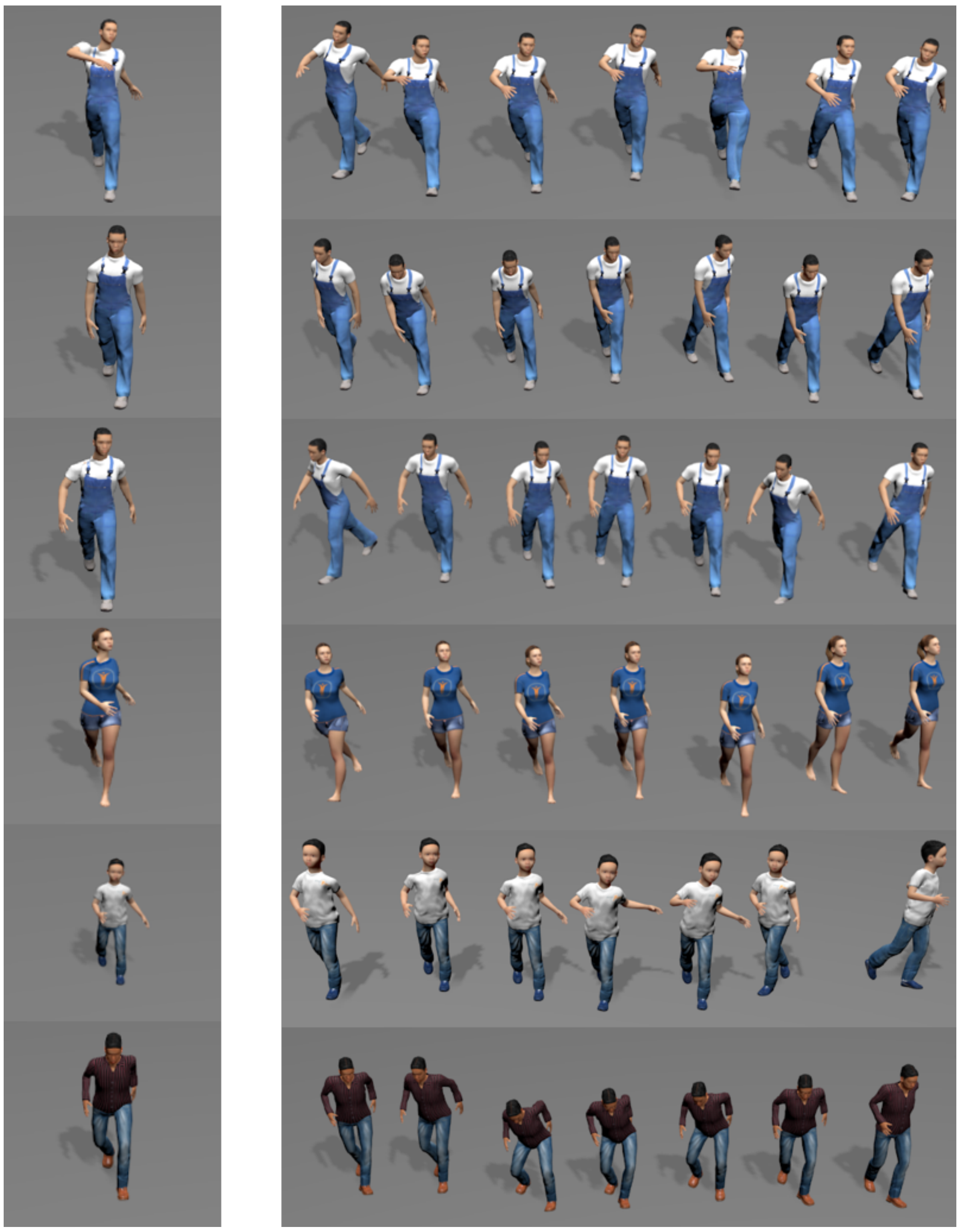


FIGURE 5.5: Random samples from six stylistic motion primitives deformed from neutral walk Left stance. The left side is the style constraints. For top to bottom, the styles are: proud, depressed, angry, female, childlike and old. The right side is the samples generated from each stylistic motion primitive.

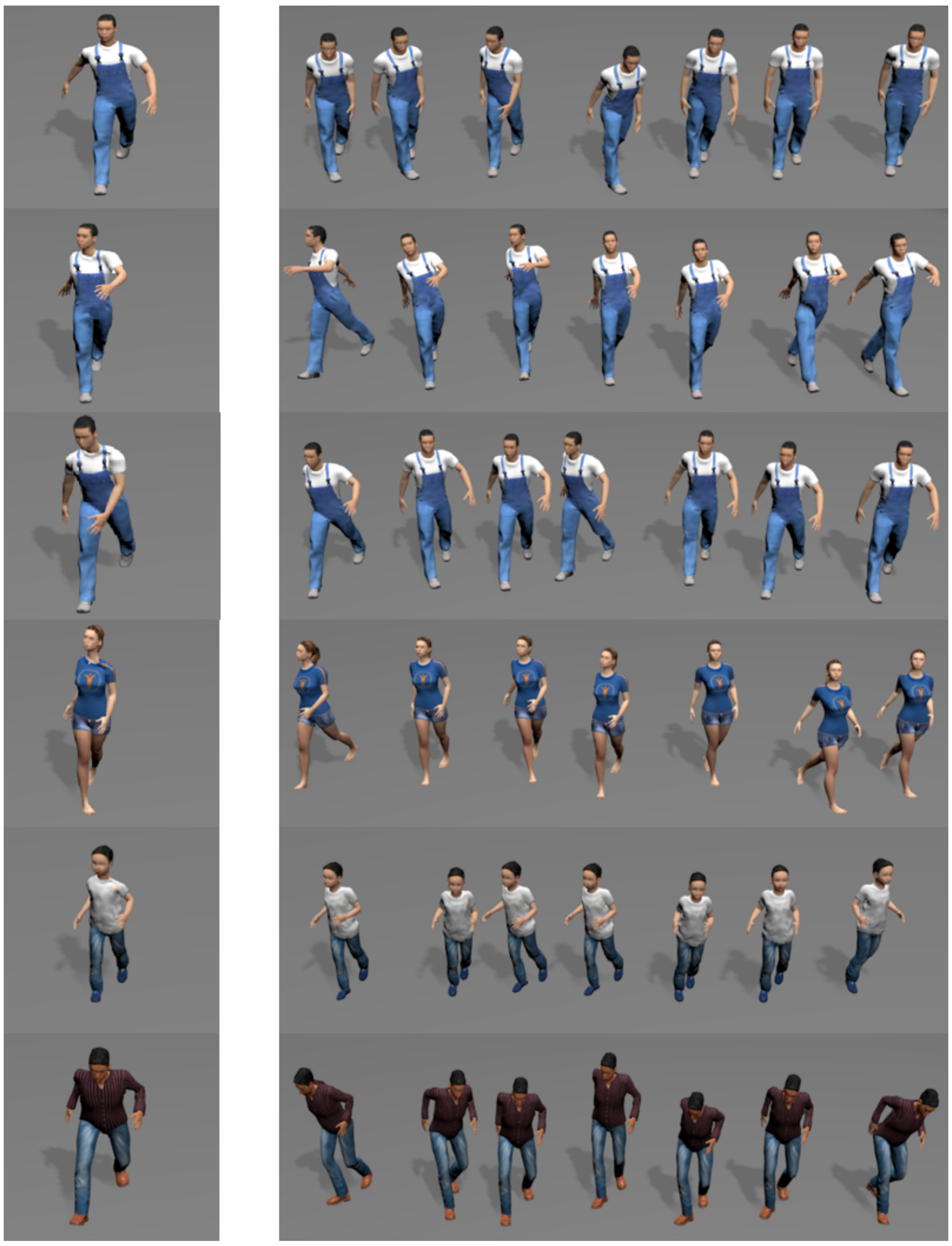


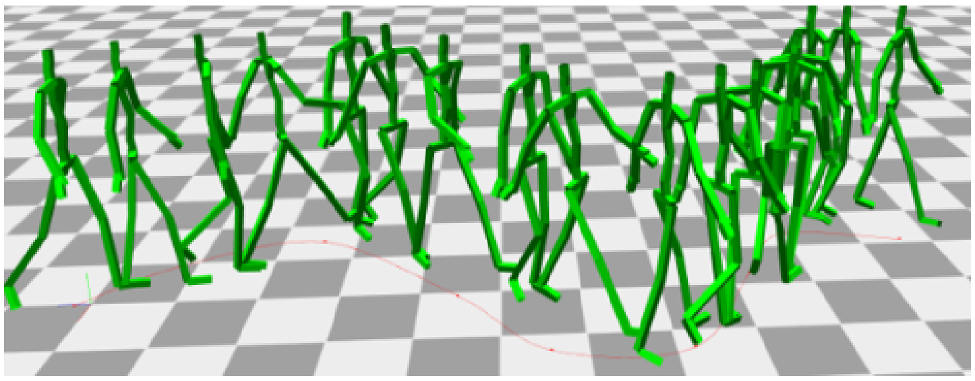
FIGURE 5.6: Random samples from six stylistic motion primitives deformed from neutral walk right stance. The left side is the style constraints. For top to bottom, the styles are: depressed, proud, angry, female, childlike and old. The right side is the samples generated from each stylistic motion primitive.

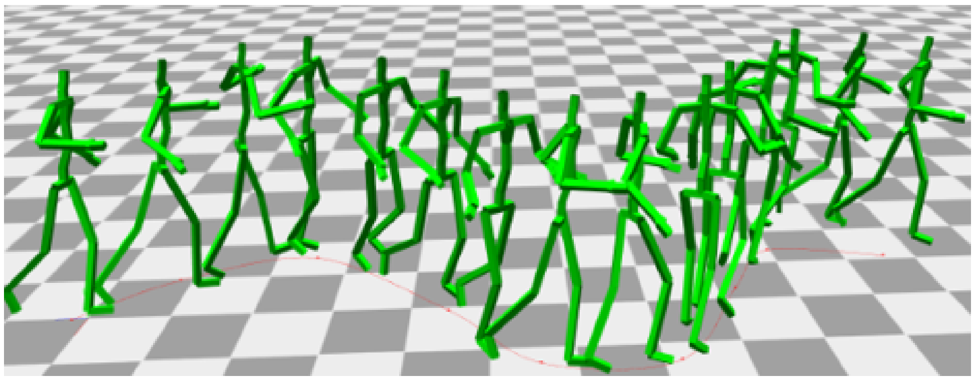
For each step, the motion synthesis task is formulated as a maximum a posteriori (MAP) problem to find the best sample satisfying constraints in the prior distribution learned by variational autoencoder. In order to make a smooth transition between two motion primitives, we take the last pose and the average speed of the sample from the previous motion primitive as the constraints for the next motion primitive. Our focus of this work is to learn and evaluate style deformable generative models for each motion primitive in a graph-based motion synthesis framework. So for the controlled motion synthesis pipeline, we use a similar approach as [MC12].

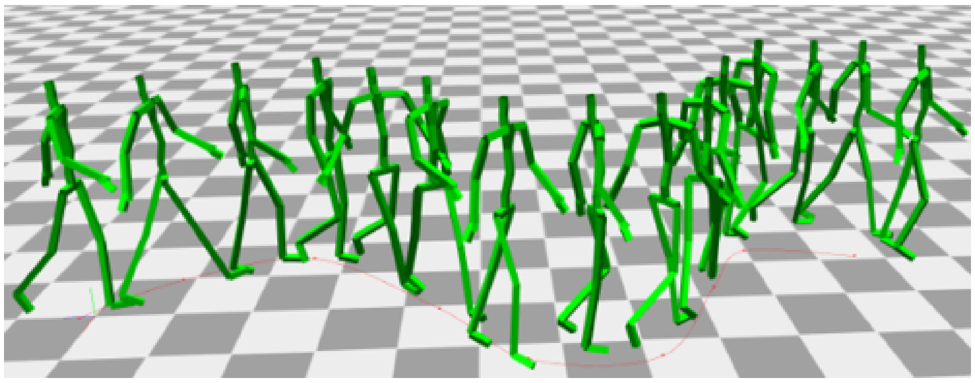
Figure 5.7 shows long animations following the given trajectory. (a) demonstrates that the neutral walking generated from VAE can follow the given trajectory well since our motion database has a large variation in trajectories. For the three stylistic walkings generated from conditional VAE, we notice that they can follow the target trajectory as well. And each of them has stylistic spatial variations, not identical to each other. All the stylistic motion primitive models are trained by taking one style example. All style examples are chosen to be roughly straight-line walking. From our experiments, we find that providing style examples without directional bias as constraints can produce better stylistic models. The results of three stylistic walkings demonstrate that conditional VAE can produce comparable variations to the neutral VAE model. In addition to spatial variation, we also notice that the style variants under the same constraint enjoy significant temporal variations. For instance, for completing the same length trajectory, the depressed walk takes longer than other styles due to the small step size.

5.4.5 Motion Diversity Analysis

It is important that the stylistic motion primitives still have the same variations in motion content as the neutral motion database. Therefore, we provide some quantitative analysis of the diversity of our models. In order to compare the variation between different motion primitives, we measure the standard deviation of a pose, speed, and step length of the samples generated from the motion primitives. The definition of the three measurements can be found as follows:


(A) neutral


(B) childlike


(C) female

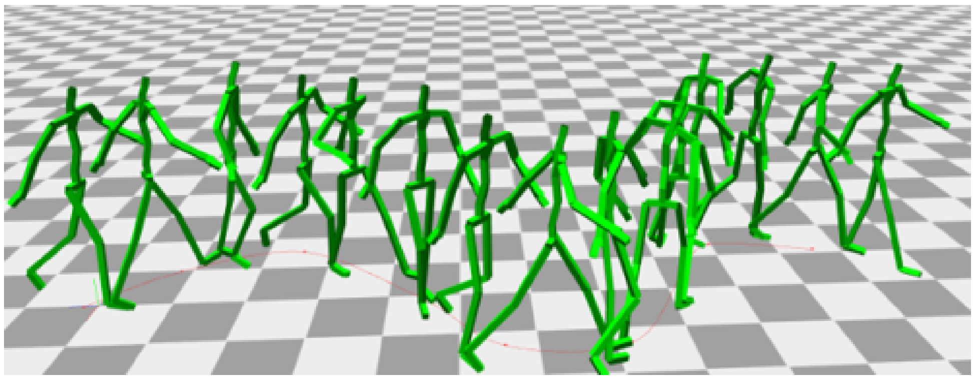

(D) angry

FIGURE 5.7: Results of generated motions following a given trajectory. (a) shows the neutral result. The motion is generated from a graph-based motion model using variational autoencoder. (b) - (d) are the stylistic results generated from a graph-based motion model using conditional variation autoencoder.

$$pose_{SD} = \sqrt{\frac{\sum_i^N (p_i - \bar{p})^2}{N-1}} \tag{5.13}$$

$$V_{SD} = \sqrt{\frac{\sum_i^N [(v_{trans,i} - \bar{v}_{trans})^2 + (v_{r,i} - \bar{v}_r)^2]}{N-1}} \tag{5.14}$$

$$step_{SD} = \sqrt{\frac{\sum_i^N (l_i - \bar{l})^2}{N-1}} \tag{5.15}$$

where p_i denotes the normalized pose, each pose is normalized to have the same root position and orientation. $v_{trans,i}$ is the translation speed on the ground and $v_{r,i}$ is the rotation speed about vertical axis. l_i is the step length for i-th motion clip. The step length is calculated as the sum over relative root translation.

Table 5.2 and Table 5.3 show the quantitative results of three criteria on the two dominant motion primitives in locomotion, which are left and right steps. We measure the variance in training data as the baseline. The neutral represents the unconditioned model learned from the neutral database. The six styles are corresponding to six stylistic variants of the original VAE model. 10,000 samples are randomly generated from each model and used to calculate three variance criteria. In general, the VAE-based generative model can well represent the variation in the training data, since the $pose_{SD}$ and V_{SD} are both very close to training data. However, the variance of step length is decreased for both models. This might be caused by the KL divergence term in loss function Eq. (5.11), which penalizes the large variations. The six stylistic motion primitives generated from a single style example demonstrate comparable variance in the pose, speed, and step length, compared to both training data and original motion primitive. For pose variance, all stylistic models are decreased. This is because the style is mainly defined by poses. Therefore, the style constraint term in Eq. (5.11) reduces the variance of poses. However, for speed and step size, most of the stylistic models actually get a large variance than the training data. This indicates that the loss function does some extrapolation during training.

TABLE 5.2: Variance evaluation of walk left stance

	$pose_{SD}$	V_{SD}	$Step_{SD}$
training data	52.8613	0.4816	10.1233
neutral	52.6766	0.4197	7.6634
proud	50.6545	0.5782	11.2539
depressed	48.3101	0.5892	9.3205
angry	49.9446	0.6611	11.7906
female	49.7224	0.5974	10.7240
childlike	51.1589	0.5058	9.6104
old	46.0082	0.5383	9.9513

TABLE 5.3: Variance evaluation of walk right stance

	$pose_{SD}$	V_{SD}	$Step_{SD}$
training data	53.0372	0.4985	9.9842
neutral	52.8817	0.4610	7.5081
proud	51.1320	0.6147	11.8916
depressed	48.5037	0.6031	9.9183
angry	50.2103	0.7035	12.7569
female	50.3722	0.6669	10.7090
childlike	51.3179	0.5389	8.8596
old	45.5661	0.5998	10.2023

5.5 Conclusion

In this work, a new approach to creating generative models for stylistic locomotion has been presented. Our approach does not require a large number of stylistic motions for training. This is because our model makes use of motion style transfer to implicitly convert the neutral motion to the target style during training. In order to demonstrate our method, we test six different styles of walking. For each style, a walking motion graph with six stylistic motion primitives is constructed. We quantitatively evaluated the variance of deformed styles compared to training data and the neutral model. Any number of stylistic samples with rich variations can be generated from our models. Our experiments result demonstrate that our model can produce stylistic motions with both spatial and temporal variations.

As a common problem in motion style transfer [Hol+17; Xia+15] our approach works best if a content-similar style motion is provided as a style constraint for model training. However, since we do not directly learn style transfer from content-similar training data, instead of using Gram matrix as style measurement, content similarity is not necessary for our approach. For instance, by adapting the style weight in Eq. (5.11), our model can still produce good stylistic begin or end walk steps using a left or right step as a style constraint. From experiments, we find that our method does not work well if the motion content is far too different from the stylistic constraint and neutral content motions. For example, we cannot deform a neutral picking model to an old picking by taking an old walking as a style constraint.

To generate a long sequence of motion, a smooth transition between motion primitives is usually required for graph-based motion synthesis approaches. The smooth transitions can either be learned by specific transition models or formulated as constraints. In this work, we simply take the last pose and average speed of the previous motion primitive as the constraints for the next one. However, we noticed that these constraints sometimes cannot be completely satisfied by optimization. Therefore, some artifacts like foot skating can be found at the transition points. In neural network-based motion synthesis approaches, the transition problem is automatically solved by the partial support of convolution. We would like to apply our approach to some deep learning-based approaches to further improve our results.

The work we presented focuses on stylistic locomotion modeling and synthesis. However, our approach is not limited to locomotion. We believe the approach of combining the variation in a large neutral motion database and the style from a few stylistic examples to learn generative models is meaningful to all kinds of actions. In the paper, we demonstrate that the VAE-based generative model is a promising approach for modeling heterogeneous actions. Our approach can be easily adapted to other actions if content-similar stylistic motions are provided.

Chapter 6

Conclusion and Future Prospects

Virtual character animations have great potential in the future of augmented reality (AR) and virtual reality (VR) applications. An immersive feeling is important for end users to take virtual simulation seriously. Robotic and deterministic motions can hardly give users a natural feeling because it is unnatural for humans to perform identical movements every time. Therefore, how to bring these randomness and variations observed in the daily activities into character animation systems correctly still poses challenges. In this thesis, we focus on modeling the variations of natural human motions with generative models and present a series of methods and tools for different simulation scenarios such as digital workers in assembly workshops and pedestrian simulation for autonomous driving. Our goal in the scope of this work is to develop methods and tools to learn and apply the natural variations from motion capture data into digital human models automatically.

The journey starts with the task to integrate experienced workers' knowledge into digital simulation tools for planning, verification, and ergonomic analysis. Most existing consumer solutions like EMA manually add knowledge to the configurations of digital avatars. We want to automatically learn this knowledge through data-driven approaches using motion capture data. The fast development of motion capture techniques makes it possible to capture multi-person in a large area conveniently. We develop our Morphable Graph framework that can create compact statistical motion models. In conclusion, two concrete contributions are made during the work: (1) a novel dimension reduction method based on the Spatio-temporal properties of the motion data, and (2) a fast search algorithm based on a space partitioning tree for real-time motion synthesis. Our work bridges the gap between industrial applications and academic results.

Recently, we have also seen many great successes in applying neural network models to human motion modeling and synthesis. We improve our Morphable Graph framework with a variational autoencoder. A general latent motion space is learned from a large, heterogeneous motion dataset.

Last but not least, we explore the possibility to use advanced style transfer technologies developed in image style transfer using neural networks for the style transfer of human motion. We develop a style generative model based on a conditional variational autoencoder which can implicitly perform the style transfer on distribution instead of a single sample.

We believe our contributions are solid both conceptually and practically. We carefully study the properties of motion capture and extend some simple but powerful methods like PCA and space partitioning to work better on the motion data. We explore the meaning of variations in the recorded motions, from the low-level structural variations such as walking path, reaching endpoints, and so on to the high-level semantic variations for example styles. Novel deep generative models are developed to model the variations in the motion data and transfer the domain features such as styles. The developed framework and tools have been used in several projects and evaluated by end users.

Acknowledgements

First of all, I would like to express my sincerest gratitude to my advisors Prof. Philipp Slusallek and Dr. Klaus Fischer for their great supports of my PhD study. Their deep academic insights and rich knowledge helped me a lot in the research work during my PhD. More importantly, their attitudes and encouragement will be always inspiring to me in my future work. I would also like to thank all my colleagues in Agents and Simulated Reality, in particular Erik Herrmann, Janis Sprenger, Noshaba Cheema, and Somayeh Hosseini. They provide me with a lot of help during my study and I really enjoy the creative and collaborative work with them. In addition, I would like to thank Prof. Martin Manns for introducing me to the topic I worked on for my study. Finally, I also would like to thank my parents and my friends for their support all these years.

Bibliography

[ADH13] Mazen Al Borno, Martin De Lasa, and Aaron Hertzmann. "Trajectory optimization for full-body movements with complex contacts". In: *IEEE transactions on visualization and computer graphics* 19.8 (2013), pp. 1405–1414.

[Ala+13] Andrea Alaimo et al. "Comparison between Euler and quaternion parametrization in UAV dynamics". In: *AIP Conference Proceedings*. Vol. 1558. 1. American Institute of Physics. 2013, pp. 1228–1231.

[Alb+05] Irene Albrecht et al. "Mixed feelings: expression of non-basic emotions in a muscle-based talking head". In: *Virtual Reality* 8.4 (2005), pp. 201–212.

[Ali+20] Sadegh Aliakbarian et al. "A stochastic conditioning scheme for diverse human motion prediction". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 5223–5232.

[AV07] David Arthur and Sergei Vassilvitskii. "K-means++: The Advantages of Careful Seeding". In: *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*. SODA '07. ACM. New Orleans, Louisiana: Society for Industrial and Applied Mathematics, 2007, pp. 1027–1035. ISBN: 978-0-898716-24-5.

[Bel66] Richard Bellman. "Dynamic programming". In: *Science* 153.3731 (1966), pp. 34–37.

[Ber+13] Jürgen Bernard et al. "MotionExplorer: Exploratory Search in Human Motion Capture Data Based on Hierarchical Aggregation". In: *IEEE Transactions on Visualization and Computer Graphics (Proc. VAST)* (Dec. 2013).

[BH00a] Matthew Brand and Aaron Hertzmann. "Style Machines". In: *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '00. New York, NY, USA: ACM Press/Addison-Wesley Publishing Co., 2000, pp. 183–192. ISBN: 1-58113-208-5. DOI: 10.1145/344779.344865.

[BH00b] Matthew Brand and Aaron Hertzmann. "Style machines". In: *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*. ACM Press/Addison-Wesley Publishing Co. 2000, pp. 183–192.

[Bis05] Alessandro Bissacco. "Modeling and learning contact dynamics in human motion". In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*. Vol. 1. IEEE. 2005, pp. 421–428.

[BKL18] Emad Barsoum, John Kender, and Zicheng Liu. "Hp-gan: Probabilistic 3d human motion prediction via gan". In: *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*. 2018, pp. 1418–1427.

[Bow00] Richard Bowden. "Learning statistical models of human motion". In: *IEEE Workshop on Human Modeling, Analysis and Synthesis, CVPR*. Vol. 2000. 2000.

[Bre97] Christoph Bregler. "Learning and recognizing human dynamics in video sequences". In: *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE. 1997, pp. 568–574.

[BS02] M. Balasubramanian and E.L. Schwartz. "The Isomap algorithm and topological stability". In: *Science* 295(5552) (2002), p. 7.

[BSH16] S. Busemann, J. Steffen, and E. Herrmann. "Interactive Planning of Manual Assembly Operations: From Language to Motion". In: *In Procedia CIRP* 41 (2016), pp. 224–229.

[BSH99] Bobby Bodenheimer, Anna V Shleyfman, and Jessica K Hodgins. "The effects of noise on the perception of animated human running". In: *Computer Animation and Simulation'99*. Springer, 1999, pp. 53–63.

[Byr+95] Richard H Byrd et al. “A limited memory algorithm for bound constrained optimization”. In: *SIAM Journal on scientific computing* 16.5 (1995), pp. 1190–1208.

[Cap] Captury. URL: `https://captury.com/`.

[CBP10] Stelian Coros, Philippe Beaudoin, and Michiel van de Panne. “Generalized biped walking control”. In: *ACM Transactions on Graphics (TOG)*. Vol. 29. 4. ACM. 2010, p. 130.

[CH05] Jinxiang Chai and Jessica K Hodgins. “Performance animation from low-dimensional control signals”. In: *ACM Transactions on Graphics (TOG)*. Vol. 24. 3. ACM. 2005, pp. 686–696.

[CH07] Jinxiang Chai and Jessica K Hodgins. “Constraint-based motion optimization using a statistical dynamic model”. In: *ACM Transactions on Graphics (TOG)*. Vol. 26. 3. ACM. 2007, p. 8.

[Che+19] Noshaba Cheema et al. “Fine-Grained Semantic Segmentation of Motion Capture Data using Dilated Temporal Fully-Convolutional Networks”. In: *arXiv preprint arXiv:1903.00695* (2019).

[Cla16] Simon Clavet. “Motion Matching and The Road to Next-Gen Animation”. In: *GDC 2016* (2016).

[Dat18] CMU Graphics Lab Motion Capture Database. 2018. URL: `http://mocap.cs.cmu.edu/`.

[DLR77] Arthur P Dempster, Nan M Laird, and Donald B Rubin. “Maximum likelihood from incomplete data via the EM algorithm”. In: *Journal of the Royal Statistical Society: Series B (Methodological)* 39.1 (1977), pp. 1–22.

[Doe16] Carl Doersch. “Tutorial on variational autoencoders”. In: *arXiv preprint arXiv:1606.05908* (2016).

[Du+16a] Han Du et al. “Joint angle data representation for data driven human motion synthesis”. In: *Procedia CIRP* 41 (2016), pp. 746–751.

[Du+16b] Han Du et al. “Scaled functional principal component analysis for human motion synthesis”. In: *Proceedings of the 9th International Conference on Motion in Games*. ACM. 2016, pp. 139–144.

[Du+19a] Han Du et al. “Stylistic locomotion modeling and synthesis using variational generative models”. In: *Motion, Interaction and Games*. 2019, pp. 1–10.

[Du+19b] Han Du et al. “Stylistic Locomotion Modeling with Conditional Variational Autoencoder.” In: *Eurographics (Short Papers)*. 2019, pp. 9–12.

[Fit54] Paul M Fitts. “The information capacity of the human motor system in controlling the amplitude of movement.” In: *Journal of experimental psychology* 47.6 (1954), p. 381.

[Fra+15] Katerina Fragkiadaki et al. “Recurrent network models for human dynamics”. In: *Proceedings of the IEEE International Conference on Computer Vision*. 2015, pp. 4346–4354.

[FWW83] Ann F Friedlaender, Clifford Winston, and Kung Wang. “Costs, technology, and productivity in the US automobile industry”. In: *The Bell Journal of Economics* (1983), pp. 1–20.

[FXS12] Andrew W Feng, Yuyu Xu, and Ari Shapiro. “An example-based motion synthesis technique for locomotion and object manipulation”. In: *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*. ACM. 2012, pp. 95–102.

[Gai+18] Felix Gaisbauer et al. “Presenting a Deep Motion Blending Approach for Simulating Natural Reach Motions.” In: *Eurographics (Posters)*. 2018, pp. 5–6.

[GBD08] Emmanuel Guigon, Pierre Baraduc, and Michel Desmurget. “Optimality, stochasticity, and variability in motor behavior”. In: *Journal of computational neuroscience* 24.1 (2008), pp. 57–68.

[GEB15] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. “A neural algorithm of artistic style”. In: *arXiv preprint arXiv:1508.06576* (2015).

[GEE06] J Gausemeier, P Ebbesmeyer, and R Eckes. “Virtual Production–Computer Model-Based Planning and Analyzing of Manufacturing Systems”. In: *Reconfigurable Manufacturing Systems and Transformable Factories*. Springer, 2006, pp. 743–759.

[Gho+20] Saeed Ghorbani et al. "Probabilistic character motion synthesis using a hierarchical deep latent variable model". In: *Computer Graphics Forum*. Vol. 39. 8. Wiley Online Library. 2020, pp. 225–239.

[Gol+02] Ary L Goldberger et al. "Fractal dynamics in physiology: alterations with disease and aging". In: *Proceedings of the national academy of sciences* 99.suppl 1 (2002), pp. 2466–2472.

[Goo+14] Ian Goodfellow et al. "Generative adversarial nets". In: *Advances in neural information processing systems* 27 (2014).

[Goo+20] Ian Goodfellow et al. "Generative adversarial networks". In: *Communications of the ACM* 63.11 (2020), pp. 139–144.

[GP12] Thomas Geijtenbeek and Nicolas Pronost. "Interactive Character Animation Using Simulated Physics: A State-of-the-Art Review". In: *Computer Graphics Forum*. Vol. 31. 8. Wiley Online Library. 2012, pp. 2492–2515.

[GPy12] GPy. *GPy: A Gaussian process framework in python*. `http://github.com/SheffieldML/GPy`. since 2012.

[Gra98] F Sebastian Grassia. "Practical parameterization of rotations using the exponential map". In: *Journal of graphics tools* 3.3 (1998), pp. 29–48.

[Gro+04] K. Grochow et al. "Style-based Inverse Kinematics". In: *In ACM Transactions on Graphics (Proc. of SIGGRAPH 2004)* (2004), pp. 522–531.

[Guo+20] Chuan Guo et al. "Action2Motion: Conditioned Generation of 3D Human Motions". In: *Proceedings of the 28th ACM International Conference on Multimedia (MM '20)*. 2020.

[Hab+17] Ikhsanul Habibie et al. "A Recurrent Variational Autoencoder for Human Motion Synthesis". In: *BMVC17* (2017).

[HAB20] Gustav Eje Henter, Simon Alexanderson, and Jonas Beskow. "Moglow: Probabilistic and controllable motion synthesis using normalising flows". In: *ACM Transactions on Graphics (TOG)* 39.6 (2020), pp. 1–14.

[Häm+14] Perttu Hämäläinen et al. "Online Motion Synthesis Using Sequential Monte Carlo". In: *ACM Trans. Graph.* 33.4 (July 2014), 51:1–51:12. ISSN: 0730-0301. DOI: 10.1145/2601097.2601218. URL: http://doi.acm.org/10.1145/2601097.2601218.

[Har+06] Björn Hartmann et al. "Reflective physical prototyping through integrated design, test, and analysis". In: *Proceedings of the 19th annual ACM symposium on User interface software and technology*. 2006, pp. 299–308.

[Har+20] Félix G Harvey et al. "Robust motion in-betweening". In: *ACM Transactions on Graphics (TOG)* 39.4 (2020), pp. 60–1.

[Her+17] Erik Herrmann et al. "Accelerating statistical human motion synthesis using space partitioning data structures". In: *Computer Animation and Virtual Worlds* 28.3-4 (2017), e1780.

[HG07a] Rachel Heck and Michael Gleicher. "Parametric Motion Graphs". In: *Proceedings of the 2007 Symposium on Interactive 3D Graphics and Games*. I3D '07. Seattle, Washington: ACM, 2007, pp. 129–136. ISBN: 978-1-59593-628-8. DOI: 10.1145/1230100.1230123. URL: http://doi.acm.org/10.1145/1230100.1230123.

[HG07b] Rachel Heck and Michael Gleicher. "Parametric motion graphs". In: *Proceedings of the 2007 symposium on Interactive 3D graphics and games*. ACM. 2007, pp. 129–136.

[HKS17] Daniel Holden, Taku Komura, and Jun Saito. "Phase-functioned Neural Networks for Character Control". In: *ACM Trans. Graph.* 36.4 (July 2017), 42:1–42:13. ISSN: 0730-0301. DOI: 10.1145/3072959.3073663.

[Hol+17] Daniel Holden et al. "Fast Neural Style Transfer for Motion Data". In: *IEEE computer graphics and applications* 37.4 (2017), pp. 42–49.

[HSK16] Daniel Holden, Jun Saito, and Taku Komura. "A deep learning framework for character motion synthesis and editing". In: *ACM Transactions on Graphics (TOG)* 35.4 (2016), p. 138.

[HW98] Christopher M Harris and Daniel M Wolpert. "Signal-dependent noise determines motor planning". In: *Nature* 394.6695 (1998), pp. 780–784.

[INT] INTERACT. *Interactive Manual Assembly Operations for the Human-Centered Workplaces of the Future*. URL: `http://www.interact-fp7.eu/`.

[Ion+14] Catalin Ionescu et al. "Human3.6M: Large Scale Datasets and Predictive Methods for 3D Human Sensing in Natural Environments". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36.7 (July 2014), pp. 1325–1339.

[JAF16] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. "Perceptual losses for real-time style transfer and super-resolution". In: *European Conference on Computer Vision*. Springer. 2016, pp. 694–711.

[Joh03] Michael Patrick Johnson. "Exploiting quaternions to support expressive interactive character motion". PhD thesis. Massachusetts Institute of Technology, 2003.

[Kap+13] Mubbasir Kapadia et al. "Efficient motion retrieval in large motion databases". In: *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*. ACM. 2013, pp. 19–28.

[KGB19] Jogendra Nath Kundu, Maharshi Gor, and R Venkatesh Babu. "Bihmp-gan: Bidirectional 3d human motion prediction gan". In: *Proceedings of the AAAI conference on artificial intelligence*. Vol. 33. 01. 2019, pp. 8553–8560.

[KGP02] Lucas Kovar, Michael Gleicher, and Frédéric Pighin. "Motion Graphs". In: *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques*. SIGGRAPH '02. San Antonio, Texas: ACM, 2002, pp. 473–482. ISBN: 1-58113-521-1. DOI: `10.1145/566570.566605`. URL: `http://doi.acm.org/10.1145/566570.566605`.

[KHS03] Kolja Kähler, Jörg Haber, and Hans-Peter Seidel. "Reanimating the dead: reconstruction of expressive faces from skull data". In: *ACM Transactions on Graphics (TOG)* 22.3 (2003), pp. 554–561.

[KN12] Yejin Kim and Michael Neff. "Component-based locomotion composition". In: (2012).

[Kov04a] Lucas Kovar. “Automated Methods for Data-driven Synthesis of Realistic and Controllable Human Motion”. AAI3155067. PhD thesis. Madison, WI, USA, 2004. ISBN: 0-496-15824-4.

[Kov04b] Lucas Kovar. “Automated methods for data-driven synthesis of realistic and controllable human motion”. PhD thesis. Citeseer, 2004.

[Krü+10] Björn Krüger et al. “Fast local and global similarity searches in large motion capture databases”. In: *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. Eurographics Association. 2010, pp. 1–10.

[KSR15] Igor Kviatkovsky, Ilan Shimshoni, and Ehud Rivlin. “Person identification from action styles”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 2015, pp. 84–92.

[Kul+11] A Kulkarni et al. “Virtual prototyping used as validation tool in automotive design”. In: *19th International Congress on Modelling and Simulation, Perth, Australia*. 2011, pp. 419–425.

[KW13] Diederik P Kingma and Max Welling. “Auto-encoding variational bayes”. In: *arXiv preprint arXiv:1312.6114* (2013).

[Law04] N.D. Lawrence. “Gaussian Process Latent Variable Models for Visualisation of High Dimensional Data”. In: *Proc. of NIPS 2003* (2004).

[Law07] Neil D Lawrence. “Learning for larger datasets with the Gaussian process latent variable model”. In: *Artificial intelligence and statistics*. PMLR. 2007, pp. 243–250.

[LBK09] Manfred Lau, Ziv Bar-Joseph, and James Kuffner. “Modeling spatial and temporal variation in motion data”. In: *ACM Transactions on Graphics (TOG)* 28.5 (2009), pp. 1–10.

[Lee+02] Jehee Lee et al. “Interactive Control of Avatars Animated with Human Motion Data”. In: *ACM Trans. Graph.* 21.3 (July 2002), pp. 491–500. ISSN: 0730-0301. DOI: `10.1145/566654.566607`.

[Lee+10] Yongjoon Lee et al. “Motion fields for interactive character locomotion”. In: *ACM Transactions on Graphics (TOG)*. Vol. 29. 6. ACM. 2010, p. 138.

[Li+17] Zimo Li et al. "Auto-Conditioned LSTM Network for Extended Complex Human Motion Synthesis". In: *CoRR* abs/1707.05363 (2017). URL: `http://arxiv.org/abs/1707.05363`.

[Lin+20] Hung Yu Ling et al. "Character controllers using motion vaes". In: *ACM Transactions on Graphics (TOG)* 39.4 (2020), pp. 40–1.

[Liu+10] Libin Liu et al. "Sampling-based contact-rich motion control". In: *ACM Transactions on Graphics (TOG)*. Vol. 29. 4. ACM. 2010, p. 128.

[LJ06] N.D. Lawrence and J.Q.Candela. "Local Distance Preservation in the GP-LVM through Back Constraints". In: *Omnipress* 23 (2006).

[LLL18] Kyungho Lee, Seyoung Lee, and Jehee Lee. "Interactive character animation by learning multi-objective control". In: *ACM Transactions on Graphics (TOG)* 37.6 (2018), pp. 1–10.

[LMH10] Martin de Lasa, Igor Mordatch, and Aaron Hertzmann. "Feature-based locomotion controllers". In: *ACM Transactions on Graphics (TOG)*. Vol. 29. 4. ACM. 2010, p. 131.

[Lop+15] Matthew Loper et al. "SMPL: A skinned multi-person linear model". In: *ACM transactions on graphics (TOG)* 34.6 (2015), pp. 1–16.

[LWS02] Yan Li, Tianshu Wang, and Heung-Yeung Shum. "Motion texture: a two-level statistical model for character motion synthesis". In: *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*. 2002, pp. 465–472.

[LYG15] Libin Liu, KangKang Yin, and Baining Guo. "Improving Sampling-based Motion Control". In: *Computer Graphics Forum*. Vol. 34. 2. Wiley Online Library. 2015, pp. 415–423.

[Ma+10] Wanli Ma et al. "Modeling style and variation in human motion". In: *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. 2010, pp. 21–30.

[MAK] MAKEHUMAN.

[Man+18] Martin Manns et al. "A new approach to plan manual assembly". In: *International Journal of Computer Integrated Manufacturing* 31.9 (2018), pp. 907–920.

[Mar63] Donald W Marquardt. "An algorithm for least-squares estimation of nonlinear parameters". In: *Journal of the society for Industrial and Applied Mathematics* 11.2 (1963), pp. 431–441.

[MBF08] Sven Mandel, Thomas Bar, and Alexander Fay. "Concept for proactive ramp-up validation of body-in-white lines". In: *2008 IEEE International Conference on Emerging Technologies and Factory Automation*. IEEE. 2008, pp. 693–696.

[MC12] Jianyuan Min and Jinxiang Chai. "Motion graphs++: a compact generative model for semantic motion analysis and synthesis". In: *ACM Transactions on Graphics (TOG)* 31.6 (2012), p. 153.

[MCC09] Jianyuan Min, Yen-Lin Chen, and Jinxiang Chai. "Interactive generation of human animation with deformable motion models". In: *ACM Transactions on Graphics (TOG)* 29.1 (2009), p. 9.

[McL93] Charles R McLean. "Computer-Aided Manufacturing System Engineering." In: *APMS*. Citeseer. 1993, pp. 341–348.

[Meh+20] Dushyant Mehta et al. "XNect: Real-time multi-person 3D motion capture with a single RGB camera". In: *ACM Transactions on Graphics (TOG)* 39.4 (2020), pp. 82–1.

[MHM18] Yuichiro Motegi, Yuma Hijioka, and Makoto Murakami. "Human Motion Generative Model Using Variational Autoencoder". In: (2018).

[MLC10] Jianyuan Min, Huajun Liu, and Jinxiang Chai. "Synthesis and editing of personalized stylistic human motion". In: *Proceedings of the 2010 ACM SIGGRAPH symposium on Interactive 3D Graphics and Games*. ACM. 2010, pp. 39–46.

[MM+01] Maddock Meredith, Steve Maddock, et al. "Motion capture file formats explained". In: *Department of Computer Science, University of Sheffield* 211 (2001), pp. 241–244.

[Mon+00] Jean-Sébastien Monzani et al. "Using an intermediate skeleton and inverse kinematics for motion retargeting". In: *Computer Graphics Forum*. Vol. 19. 3. Wiley Online Library. 2000, pp. 11–19.

[MR81] C.S. Myers and L.R. Rabiner. "A comparative study of several dynamic time-warping algorithms for connected word recognition". In: *In The Bell System Technical Journal* 60.7 (1981), pp. 1389–1409.

[Mül+07] M. Müller et al. *Documentation Mocap Database HDM05*. Tech. rep. CG-2007-2. Universität Bonn, June 2007.

[NWW12] Shimon Y Nof, Wilbert E Wilhelm, and H Warnecke. *Industrial assembly*. Springer Science & Business Media, 2012.

[OBB20] Ahmed A A Osman, Timo Bolkart, and Michael J. Black. "STAR: A Sparse Trained Articulated Human Body Regressor". In: *European Conference on Computer Vision (ECCV)*. 2020, pp. 598–613. URL: `https://star.is.tue.mpg.de`.

[Opt18] OptiTrack. *OtiTrack Motion Capture Systems*. June 2018. URL: `https://optitrack.com/`.

[Osh08] Masaki Oshita. "Smart motion synthesis". In: *Computer Graphics Forum*. Vol. 27. 7. Wiley Online Library. 2008, pp. 1909–1918.

[Pan+05] Y Pang et al. "Assembly Design and Evaluation in an Augmented Reality Environment". In: (2005).

[Pav+20] Dario Pavllo et al. "Modeling human motion with quaternion-based neural networks". In: *International Journal of Computer Vision* 128 (2020), pp. 855–872.

[PB00] Katherine Pullen and Christoph Bregler. "Animating by multi-level sampling". In: *Proceedings Computer Animation 2000*. IEEE. 2000, pp. 36–42.

[PB02] Katherine Pullen and Christoph Bregler. "Motion capture assisted animation: Texturing and synthesis". In: *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*. 2002, pp. 501–508.

[PBV21] Mathis Petrovich, Michael J. Black, and Gül Varol. "Action-Conditioned 3D Human Motion Synthesis with Transformer VAE". In: *International Conference on Computer Vision (ICCV)*. Oct. 2021, pp. 10985–10995.

[Pea01] K. Pearson. "On lines and planes of closest fit to systems of points in space". In: *Philosophical Magazine* 2.6 (1901), pp. 559–572.

[Per85] Ken Perlin. "An image synthesizer". In: *ACM Siggraph Computer Graphics* 19.3 (1985), pp. 287–296.

[Per95] Ken Perlin. "Real time responsive animation with personality". In: *IEEE transactions on visualization and Computer Graphics* 1.1 (1995), pp. 5–15.

[PG96] Ken Perlin and Athomas Goldberg. "Improv: A system for scripting interactive actors in virtual worlds". In: *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*. 1996, pp. 205–216.

[PRM00] Vladimir Pavlovic, James M Rehg, and John MacCormick. "Learning switching linear models of human motion". In: *NIPS*. Vol. 2. 3. 2000, p. 4.

[RMW14] Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. "Stochastic backpropagation and approximate inference in deep generative models". In: *International conference on machine learning*. PMLR. 2014, pp. 1278–1286.

[RS00] S.T. Roweis and L.K. Saul. "Nonlinear Dimensionality Reduction by Locally Linear Embedding". In: *Science* 290.5500 (2000), pp. 2319–2323.

[RS05] Jim O. Ramsay and Bernard W. Silverman. *Functional Data Analysis*. Springer, 2005.

[RT02] Michael A Riley and Michael T Turvey. "Variability and determinism in motor behavior". In: *Journal of motor behavior* 34.2 (2002), pp. 99–125.

[RWV19] Sarah Ribet, Hazem Wannous, and Jean-Philippe Vandeborre. "Survey on style in 3d human body motion: Taxonomy, data, recognition and its applications". In: *IEEE Transactions on Affective Computing* 12.4 (2019), pp. 928–948.

[Sch78] Gideon Schwarz. "Estimating the dimension of a model". In: *The annals of statistics* (1978), pp. 461–464.

[Shi+20] Mingyi Shi et al. "Motionet: 3d human motion reconstruction from monocular video with skeleton consistency". In: *ACM Transactions on Graphics (TOG)* 40.1 (2020), pp. 1–15.

[Sho85] Ken Shoemake. "Animating rotation with quaternion curves". In: *Proceedings of the 12th annual conference on Computer graphics and interactive techniques*. 1985, pp. 245–254.

[SHP04] Alla Safonova, Jessica K Hodgins, and Nancy S Pollard. "Synthesizing physically realistic human motion in low-dimensional, behavior-specific spaces". In: *ACM Transactions on Graphics (TOG)* 23.3 (2004), pp. 514–521.

[SMJ05] Puneet Singla, Daniele Mortari, and John L Junkins. "How to avoid singularity when using Euler angles?" In: *Advances in the Astronautical Sciences* 119.SUPPL. (2005), pp. 1409–1426.

[Son+10] Kristian Sons et al. "XML3D: interactive 3D graphics for the web". In: *Proceedings of the 15th International Conference on Web 3D Technology*. 2010, pp. 175–184.

[Spr+20] Janis Sprenger et al. "Variational Interpolating Neural Networks for Locomotion Synthesis". In: *DHM2020*. IOS Press, 2020, pp. 71–81.

[Sta+19] Sebastian Starke et al. "Neural state machine for character-scene interactions." In: *ACM Trans. Graph.* 38.6 (2019), pp. 209–1.

[Sta+20] Sebastian Starke et al. "Local motion phases for learning multi-contact character movements". In: *ACM Transactions on Graphics (TOG)* 39.4 (2020), pp. 54–1.

[Sun+21] Yu Sun et al. "Monocular, One-stage, Regression of Multiple 3D People". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 11179–11188.

[TH00] Luis Molina Tanco and Adrian Hilton. "Realistic synthesis of novel human movements from a database of motion capture examples". In: *Proceedings Workshop on Human Motion*. IEEE. 2000, pp. 137–142.

[THB06] Lorenzo Torresani, Peggy Hackney, and Christoph Bregler. "Learning motion style synthesis from perceptual observations". In: *Advances in neural information processing systems* 19 (2006), pp. 1393–1400.

[THR07] Graham W Taylor, Geoffrey E Hinton, and Sam T Roweis. "Modeling human motion using binary latent variables". In: *Advances in neural information processing systems*. 2007, pp. 1345–1352.

[THR11] Graham W. Taylor, Geoffrey E. Hinton, and Sam T. Roweis. "Two Distributed-State Models For Generating High-Dimensional Time Series". In: *J. Mach. Learn. Res.* 12 (July 2011), pp. 1025–1068. ISSN: 1532-4435. URL: `http://dl.acm.org/citation.cfm?id=1953048.2021035`.

[Toy+17] Sam Toyer et al. "Human pose forecasting via deep markov models". In: *2017 International Conference on Digital Image Computing: Techniques and Applications (DICTA)*. IEEE. 2017, pp. 1–8.

[UAT95] Munetoshi Unuma, Ken Anjyo, and Ryozo Takeuchi. "Fourier principles for emotion-based human figure animation". In: *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*. ACM. 1995, pp. 91–96.

[UD07] Raquel Urtasun and Trevor Darrell. "Discriminative Gaussian process latent variable model for classification". In: *Proceedings of the 24th international conference on Machine learning*. 2007, pp. 927–934.

[Urt+04] Raquel Urtasun et al. "Style-based motion synthesis". In: *Computer Graphics Forum*. Vol. 23. 4. Wiley Online Library. 2004, pp. 799–812.

[VH08] Laurens Van der Maaten and Geoffrey Hinton. "Visualizing data using t-SNE." In: *Journal of machine learning research* 9.11 (2008).

[Wan+20] Qi Wang et al. "Adversarial learning for modeling human motion". In: *The Visual Computer* 36.1 (2020), pp. 141–160.

[WCM16] Jane-Ling Wang, Jeng-Min Chiou, and Hans-Georg Müller. “Functional data analysis”. In: *Annual Review of Statistics and Its Application* 3 (2016), pp. 257–295.

[WCW14] Xin Wang, Qiudi Chen, and Wanliang Wang. “3D human motion editing and synthesis: A survey”. In: *Computational and Mathematical methods in medicine* 2014 (2014).

[WCX19] Zhiyong Wang, Jinxiang Chai, and Shihong Xia. “Combining recurrent neural networks and adversarial training for human motion synthesis and control”. In: *IEEE transactions on visualization and computer graphics* 27.1 (2019), pp. 14–28.

[Wea99] David L Weakliem. “A critique of the Bayesian information criterion for model selection”. In: *Sociological Methods & Research* 27.3 (1999), pp. 359–397.

[Wei+11] Thibaut Weise et al. “Realtime performance-based facial animation”. In: *ACM transactions on graphics (TOG)* 30.4 (2011), pp. 1–10.

[Wes05] Cord Westhoff. “Person identification from biological motion”. In: *Unpublished Inaugural-Dissertation, Ruhr University, Bochum* (2005).

[WFH08] Jack M Wang, David J Fleet, and Aaron Hertzmann. “Gaussian process dynamical models for human motion”. In: *IEEE transactions on pattern analysis and machine intelligence* 30.2 (2008), pp. 283–298.

[WH97] Douglas J Wiley and James K Hahn. “Interpolation synthesis of articulated figure motion”. In: *IEEE Computer Graphics and Applications* 17.6 (1997), pp. 39–45.

[WMC11] Xiaolin Wei, Jianyuan Min, and Jinxiang Chai. “Physically valid statistical models for human motion generation”. In: *ACM Transactions on Graphics (TOG)* 30.3 (2011), pp. 1–10.

[WN15] Yingying Wang and Michael Neff. “Deep signatures for indexing and retrieval in large motion databases”. In: *Proceedings of the 8th ACM SIGGRAPH Conference on Motion in Games*. ACM. 2015, pp. 37–45.

[Woo99] Robert Sessions Woodworth. “Accuracy of voluntary movement.” In: *The Psychological Review: Monograph Supplements* 3.3 (1899), p. i.

[Xia+15] Shihong Xia et al. “Realtime style transfer for unlabeled heterogeneous human motion”. In: *ACM Transactions on Graphics (TOG)* 34.4 (2015), p. 119.

[Xu+20] Jingwei Xu et al. “Hierarchical style-based networks for motion synthesis”. In: *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI 16*. Springer. 2020, pp. 178–194.

[Yan+18] Xinchen Yan et al. “Mt-vae: Learning motion transformations to generate multimodal human dynamics”. In: *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018, pp. 265–281.

[Yan+19a] Sijie Yan et al. “Convolutional sequence generation for skeleton-based action synthesis”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 4394–4402.

[Yan+19b] Yanfu Yan et al. “Feafa: A well-annotated dataset for facial expression analysis and 3d facial animation”. In: *2019 IEEE International Conference on Multimedia & Expo Workshops (ICMEW)*. IEEE. 2019, pp. 96–101.

[YLP07] KangKang Yin, Kevin Loken, and Michiel van de Panne. “SIMBICON: Simple Biped Locomotion Control”. In: *ACM SIGGRAPH 2007 Papers*. SIGGRAPH ’07. San Diego, California: ACM, 2007. DOI: 10.1145/1275808.1276509. URL: http://doi.acm.org/10.1145/1275808.1276509.

[YM16] M Ersin Yumer and Niloy J Mitra. “Spectral style transfer for human motion between independent actions”. In: *ACM Transactions on Graphics (TOG)* 35.4 (2016), p. 137.

[YT07] Qinxin Yu and Demetri Terzopoulos. *A decision network framework for the behavioral animation of virtual humans*. Vol. 68. 05. Citeseer, 2007.

[Zha+18] He Zhang et al. “Mode-adaptive neural networks for quadruped motion control”. In: *ACM Transactions on Graphics (TOG)* 37.4 (2018), pp. 1–11.

[Zho+14] Liuyang Zhou et al. “Human motion variation synthesis with multivariate Gaussian processes”. In: *Computer Animation and Virtual Worlds* 25.3-4 (2014), pp. 301–309.

[Zho+19] Yi Zhou et al. “On the continuity of rotation representations in neural networks”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2019, pp. 5745–5753.

[Zhu+97] Ciyou Zhu et al. “Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization”. In: *ACM Transactions on mathematical software (TOMS)* 23.4 (1997), pp. 550–560.

[Zor+03] Farbod Zorriassatine et al. “A survey of virtual prototyping techniques for mechanical product development”. In: *Proceedings of the institution of mechanical engineers, Part B: Journal of engineering manufacture* 217.4 (2003), pp. 513–530.

[ZSJ20] Rui Zhao, Hui Su, and Qiang Ji. “Bayesian Adversarial Human Motion Synthesis”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2020.

Acronyms

BVH	Biovision Hierarchy File Format
CNL	Controlled Natural Language
CNN	Convolutional Neural Network
DBN	Dynamic Bayesian Network
DHM	Digital Human Model
DOF	Degree of Freedom
FPCA	Functional Principal Component Analysis
GAN	Generative Adversarial Network
GMM	Gaussian Mixture Model
GP	Gaussian Process
GPDM	Gaussian Process Dynamical Model
GPLVM	Gaussian Process Latent Variable Model
HMM	Hidden Markov Model
LDS	Linear Dynamical System
LSTM	Long Short-term Memory
MAP	Maximum A Posteriori
MLP	Multilayer Perceptron
PCA	Principal Component Analysis
RBM	Restricted Boltzmann Machine
RNN	Recurrent Neural Network
SFPCA	Scaled Functional Principal Component Analysis
SLDS	Switching Linear Dynamical System
VAE	Variational Autoencoder