Human Motion Analysis and Synthesis in Computer Graphics

Yijun Shen

A Thesis presented for the degree of
Doctor of Philosophy

Department of Computer and Information Science

Faculty of Engineering and Environment

Northumbria University

January 2018

Abstract

This thesis focuses on solving a challenging problem in the field of computer graphics, namely to model and understand 3D human motion efficiently and meaningfully. This is vital to achieve the analysis (health & sports science), synthesis (character animation) and control (video game) of human movements. Though numerous studies have focused on improving the results of motion analysis, motion synthesis and motion control, only a few of these studies solved the problems from the fundamental part owing to the lack of information encoded in motion data.

In my works, the motion of human was divided into the three types, namely single human motion, multi-people interactions and crowd movement. Subsequently, I solved the problems from motion analysis to motion control in different types of motion.

In the single human motion, two types of motion graphs on the motion sequence were proposed using Markov Process. The human motion is represented as the directed graphs, which suggests the number of action patterns and transitions among them. By analyzing the graphs topologies, the richness, transitions flexibility and unpredictability among different action patterns inside the human motion sequence can be easily verified. The framework here is capable of visualizing and analyzing the human motion on the high level of action preference, intention and diversity.

For the two people interaction motion, the use of 3D volumetric meshes on the interacting people was proposed to model their movement and spatial relationship among them. The semantic meanings of the motions were defined by such relationship. A customized Earth Movers Distance was proposed to assess the topological and geometric difference between two groups of meshes. The above assessment captured the semantic similarities among different two-people interactions, which is

consistent with what humans perceive. With this interaction motion representation, the multi-people interactions in semantic level can be retrieved and analyzed, and such complex movements can be easily adapted and synthesized with low computational costs.

In the crowd movement, a data-driven gesture-based crowd control system was proposed, in which the control scheme was learned from example gestures provided by different users. The users gestures and corresponding crowd motions, representable to the crowd motions properties and irrelevant to style variations of gestures and crowd motions, were modelled into a compact low dimensional space. With this representation, the proposed framework can take an arbitrary users input gesture and generate appropriate crowd motion in real time.

This thesis shows the advantages of higher-level human motion modelling in different scenarios and solves different challenging tasks of computer graphics. The unified framework summarizes the knowledge to analyze, synthesize and control the movement of human.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

Yijun Shen

January 2018

Acknowledgements

First and foremost, I would like to express my gratitude to my principal supervisor, Dr. Hubert P. H. Shum, for giving me constant support during my PhD study. His enthusiasm and creativity greatly inspired me, and he has always given me numerous helps whenever I was stuck in research bottleneck. More importantly, he also focuses on improving my research abilities, which will be conducive to my future career.

I would like to thank Dr. Edmond S. L. Ho, Dr. Longzhi Yang and Dr. He Wang for their dedications to my study. They gave me huge support and considerable advices to my research projects, from methodology design to experimental assessments. They also greatly helped me with writing the research papers.

Also, I would be thankful to external collaborators and my fellow colleagues. I wish to thank Prof. Franck Multon and Dr. Ludovic Hoyet, for giving me an opportunity to do a short-term research session in the French Institute for Research in Computer Science and Automation (INRIA). Their precious experiences broaden the scope of my research. I want to thank Dr. Naoya Iwamoto, Dr. Lining Zhang, Jingtian Zhang, Shanfeng Hu and Gengshen Wu for their support and encouragement. It is such a good memory working with them.

Furthermore, I would like to thank the examiners, Prof. Shaun Lawson and Prof. Adrian Hilton, for reviewing my thesis. Their valuable comments will help me a lot with my thesis.

Finally, I would like to thank my parents and my friends who have been supporting and encouraging me all the time.

Dedicated to my beloved parents and my girlfriend,
who shared every moment of my happiness and sadness,
and fully supported me during my PhD study.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The modelling and the understanding of 3D human motion have become important research topics in the field of computer graphics. 3D human motion has been widely applied in CGI movies, video games, sports training and physical recovering for the disabled. This is the fundamental component to synthesize, real-time control and analyze human movement. An overall picture of human motion modelling and understanding in computer graphics is given in Fig 1.1.

In the entertainment industry, the production of high-quality animations in CGI movies and video games relies on the realistic motions on the anthropomorphic & humanoid character and the proper interactions among multiple characters [1–3].For instance, in the video games, e.g., Assassins Creed and Grand Theft Auto, the most attractive part is that the protagonist can be conducted to do some realistic motions and interact with those virtual characters inside the game by players based on their control and analysis. There are similar situations in CGI movies (e.g. Avatar). In sports and health science, a candidates motion is captured and analyzed so that the diagnosis can be provided to the candidate to improve his/her motion in sports and rehabilitate his/her body [4, 5]. Modelling and understanding human motion in the semantic level can better achieve such tasks.

Though numerous studies have been conducted on human motion synthesis and analysis, most of them focused on low level motion features (e.g. position, velocity, acceleration and torque on each joint of human body). As a result, many manual work are required to understand/represent the semantic meaning of human motion

and synthesize multi-character with complex interactions. This chapter briefly introduces the techniques in motion analysis and synthesis and highlight the difficulties to achieve such tasks. Subsequently, three problems to be addressed in this thesis are introduced, and the summary of our research is made.

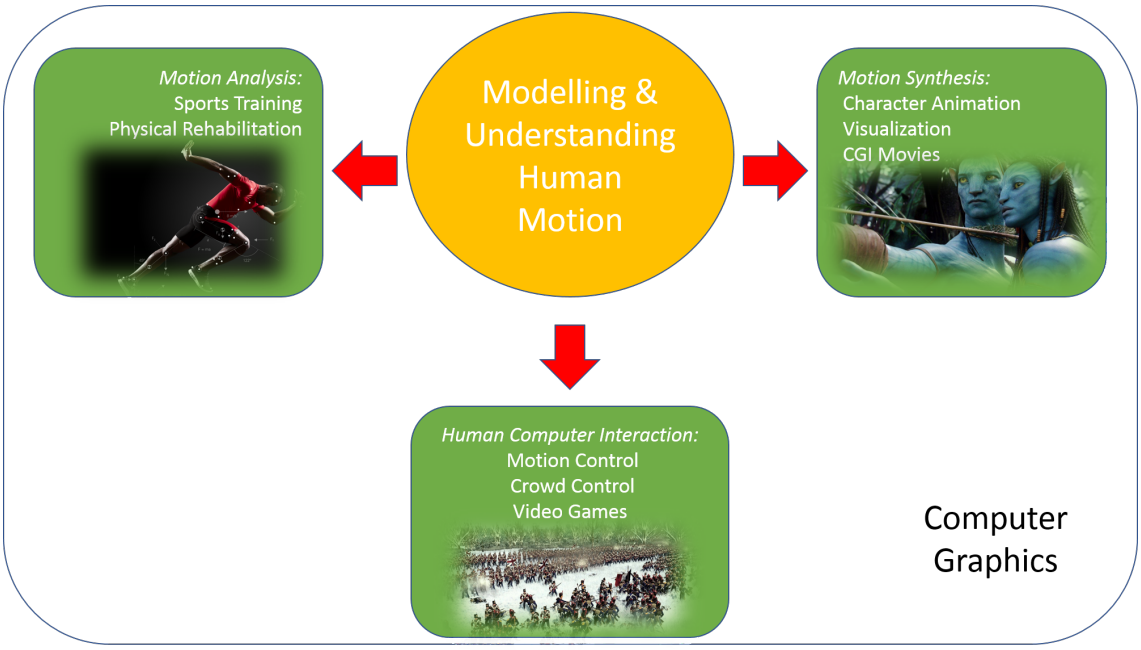


Figure 1.1: *Human motion modelling and understanding in computer graphics.*

1.1 Motivation on This Research

Human body structure and motion data are usually formatted into a hierarchical skeletal model in computer graphics Fig 1.2. The body dimension is shown as the joints and bones length, and the motion is represented by the translation and rotation data of each joint at each frame [6].Based on the mentioned information, human motions can be rendered and visualized using the character model, and the intrinsic kinematic and dynamic properties of the motion can be presented. As the Motion Capture technologies are advancing, natural and realistic human motions can be easy to record and edit for motion analysis and synthesis. However, as the motion analysis and synthesis problems are becoming increasingly complex, extracting the latent information of the movement will be more meaningful. Besides, the mentioned information is not encoded in the motion data. According to the relevant research in

recent years [7–12], modelling human motions in the higher level is vital to improve the motion analysis, action recognition and motion synthesis tasks.

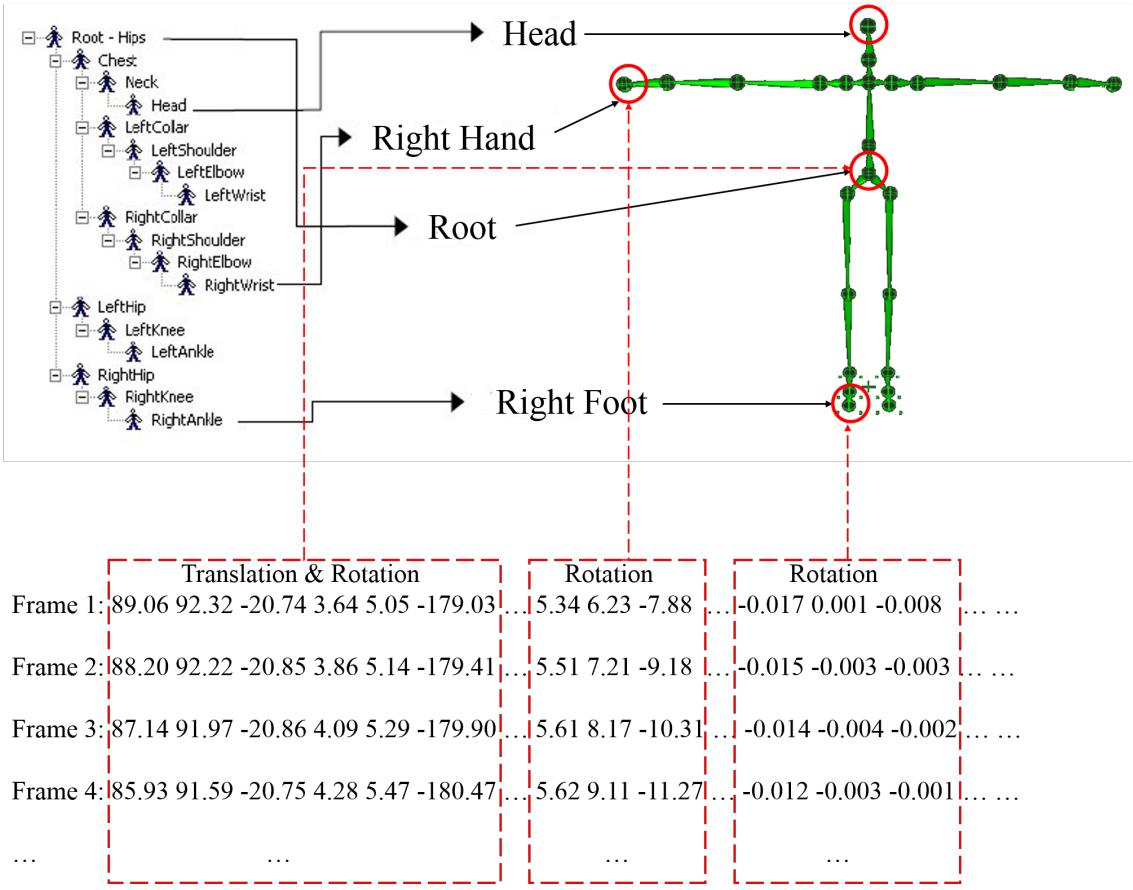


Figure 1.2: *Hierarchical skeletal structure of human model and it's motion data in computer graphics.*

In the sports motion analysis, it is critical to assess the high-level skills of motions rather than low-level kinematic details of the movement (e.g. speed and strength). For instance, in the boxing training, a boxers skill is assessed by the richness of attacking/defending actions, flexibility of transitions between attacking and defending, as well as the unpredictability of these action patterns. Such information is often used to assess the skill level of a boxer, and it is difficult to be achieved only by analyzing the kinematics information of the movement. Similar principle can be adopted to many other sports (e.g. basketball, football and fencing).

Considerable interactions are involved in the human daily movements. The important semantic meaning of many motions is defined by the information of these

interactions. For instance, in the scenarios shown in Fig. 1.3, these sitting motions might be considered different from each other because of their different postures. In human perception, these motions are considered the same type because they share the similar semantic meaning of "*a person is sitting on a chair*". This high-level semantic meaning of the sitting motion is implied by the spatial relationship between the person and the chair. Such an implicit spatial relationship helps to capture the semantics of motion, and it is unlikely to be extracted only based on human motion data. This rule can be generalized to most types of human-object and multi-people interaction motions. During the process of the motion retrieval and action recognition, introducing such spatial relationship into the system can improve the retrieval performance and recognition accuracy. For the motion synthesis, modelling such spatial relationship can efficiently avoid the collision and prevent output the animation from those artefacts (e.g. penetration of body parts between interacting characters).

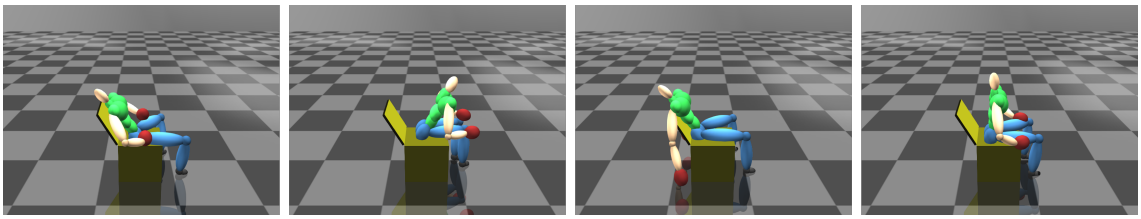


Figure 1.3: *Different styles of sitting motions.*

During the crowd simulation, people tend to prioritize the overall crowd behavior rather than the movement of each individual in the crowd. For instance, in the video game Total War series, one of the important features is to let the player to intuitively control thousands of troops for different purpose (e.g. attacking, marching, changing formation and retreating). For this end, crowd motion is always modelled as an entity, and the control schemes are designed on the top of it. Besides, the motion pattern of the crowd is hard to learn directly from the human motion data.

Here comes the common problem in the above three topics: learning an intuitive way to visualize and represent and/or synthesize (multi-) character interaction in a data-driven manner. The major motivation here is to explore the row motion data of characters movement and find the semantic level latent information of characters

motions and interactions among. Next, this information is applied to the tasks of motion analysis/retrieval, synthesis and control to improve the output results.

In general, to model human motion in a meaningful way, so that the knowledge can be generalized to motion synthesis, motion analysis and action recognition. The present thesis emphasizes the methods of modelling human motion on different application fields of computer graphics, and then produce high-quality scenes up to the industrial criteria (e.g. motion analysis according to human perception, accurate motion retrieval in complex interactions and simple/intuitive motion synthesis and control).

1.2 Problem Definition and Proposed Solution

In this section, the following three problems to be solved in the thesis are defined, and the proposed solutions to tackle the problems are briefed.

1.2.1 Skill Level Sports Motion Analysis

Automatic assessment of sports skills has been an active research area. However, most existing studies focus on low-level features e.g. movement speed and strength. In this work, we propose a framework for automatic motion analysis and visualization, which allows us to assess high-level skills e.g. the richness of actions, the flexibility of transitions and the unpredictability of action patterns. The core of our framework is the construction and visualization of the posture-based graph that focuses on the standard postures for launching and ending actions, as well as the action-based graph that focuses on the preference of actions and their transition probability. We further propose two numerical indices, the Connectivity Index and the Action Strategy Index, to assess skill level according to the graph. We demonstrate our framework with motions captured from different boxers. Experimental results demonstrate that our system can effectively visualize the strengths and weaknesses of the boxers. More details can be found in Chapter 3.

1.2.2 Interaction-based Human Motion Retrieval and Analysis

Traditional methods for motion retrieval and analysis consider features from individual characters. However, the semantic meaning of many motions is defined by the interaction between characters. There is little success in adapting interaction-based features in assessing interaction difference, as they are either topologically different across interactions or high dimensional. In this work, we propose a new unified framework for motion retrieval and analysis from the interaction point of view. We adapt the Earth Movers Distance to optimally match interaction features of different topology, which allows us to compare different classes of interactions and discover their intrinsic semantic similarity. We demonstrate how the system can retrieve interactions of similar semantic meaning filtered by user-given constraints. We also show how it can assess and visualize the body parts that contribute to the semantic difference between interactions. We construct a comprehensive kick-boxing interaction database that is open for public for research benchmark. Experimental results show that our method outperforms existing research and aligns better with human perceived interaction similarity. More details can be found in Chapter 4.

1.2.3 Data-Driven Crowd Motion Control

Controlling a crowd using multi-touch devices appeals to the computer games and animation industries, as such devices provide a high dimensional control signal that can effectively define the crowd formation and movement. However, existing works depending on the pre-defined control schemes require the users to learn a scheme that may not be intuitive. A data-driven gesture-based crowd control system is proposed, in which the control scheme is learned from example gestures provided by different users. In particular, a database with pairwise samples of gestures and crowd motions is built. To effectively generalize the gesture style of different users, e.g. the use of different numbers of fingers, a set of gesture features is proposed to represent a set of hand gesture trajectories. Likewise, to represent crowd motion trajectories of different numbers of characters over time, we propose a set of crowd

motion features that are extracted from a Gaussian mixture model. Given a run-time gesture, our system extracts the K nearest gestures from the database and interpolates the corresponding crowd motions to achieve the run-time control. Our system is accurate and efficient, making it suitable for real-time applications (e.g. real-time strategy games and interactive animation controls). For more details, see Chapter 5.

1.3 Related Publications

In this section, we list the following papers which are related to our research in the thesis. These papers have been published or under revision:

- Yijun Shen, Jingtian Zhang, Longzhi Yang, and Hubert P. H. Shum. *Depth Sensor-Based Facial and Body Animation Control*, pages 1-16. Springer International Publishing, Cham, 2016.
- Yijun Shen, He Wang, Edmond S. L. Ho, Longzhi Yang, and Hubert P. H. Shum. Posture-based and action-based graphs for boxing skill visualization. *Computers and Graphics*, 69(Supplement C):104-115, 2017.
- Yijun Shen, Longzhi Yang, Edmond S. L. Ho, and Hubert P. H. Shum. *Interaction-based Human Motion Retrieval and Analysis*, to be sumbitted to IEEE Transactions on Visualization and Computer Graphics.
- Yijun Shen, Joseph Henry, He Wang, Edmond S. L. Ho, Taku Komura, and Hubert P. H. Shum. *Data-Driven Crowd Motion Control with Multi-touch Gestures*, submitted to Computer Graphics Forum, under minor revision.

1.4 Thesis Structure

The structure of the thesis is as follow: First, we review the related research of motion analysis and synthesis in Chapter 2. This part covers the state-of-art approaches in those areas which are related to our research. In Chapter 3, we first introduce

our framework to model the single human movement, and then our method on analyzing and visualizing the skill level of the movement. In Chapter 4, we introduce our unified framework on modelling, retrieving and analyzing two people interaction motions. In Chapter 5, we introduce our data-driven method on the crowd motion control system, which enables the user to conduct arbitrary gesture to generate an appropriate crowd motion. Finally, a conclusion and discussion of our thesis are drawn in Chapter 6.

Chapter 2

Related Works

Modelling 3D human motions has been an active research area in computer graphics, computer vision and robotics in recent decades. It becomes the foundation part of motion synthesis and motion analysis techniques, which have been widely applied in entertainment industries and physical rehabilitation in the health-care area. Human motion modelling methods can be considered as the following three aspects: single person movement, interactions between multiple persons and crowd motions.

Modelling single person movements mainly focus on the detailed information of his/her motions. It represents the motions into low-level kinematics/dynamics factors (e.g. velocity, acceleration, force and power) and high-level semantic features (e.g. transition strategies from one action to another in the motion sequence). This information can be easily used for motion synthesis and motion analysis of an individual person, as discussed in Section 2.1. However, with the demand for achieving highly realistic scenarios in computer games and animations, modelling interactions between multiple people play a key role in such a purpose. It mainly focuses on modelling the spatial relationships between interacting people and this information will be used to analyze the semantic meaning of the motion. For instance, in the two people boxing motion, such spatial relationship is represented as one persons hand fast reaching the opponents body part. More details are discussed in Section 2.2.

Different from the above two aspects, crowd motion modelling does not focus on the low-level detailed information of each individual's motions and interactions

between different individuals. It formulates the whole crowd as an entity. Control schemes are designed on top of the entity to achieve high-level overall movement, e.g. controlling the whole crowd to go through the constrained environment and reach the target position. More information is available in Section 2.3.

Our approaches use a large amount of motion capture data, which need to be pre-processed due to the noise caused by devices and environments. We explore the mature techniques on refining and denoising the motion capture data in Section 2.4. Some of them are applied in our research.

2.1 Single Person Motion Analysis

Analyzing 3D human movement from motion capture system are widely used in healthcare [13–15], sports training [16, 17] and dangerous motion detection [18, 19]. Due to our research interests and the motion database we have created, we focus on the sports motion analysis and visualization in our thesis. First, we review the mainstream techniques of sports motion analysis and highlight the weakness of those approaches compared with our method in Chapter 3. Then, we review the recently human motion modelling methods with graph representations and dimensionality reduction techniques. Furthermore, we highlight how our method in Chapter 3 can outperform existing motion modelling methods in visualizing high level skill information of complex human motions.

2.1.1 Sports Analysis

Helping athletes on skill improving via the visualization of sports motions is a field that has not been fully explored in the field of sports science. Existing research [20, 21] mainly focuses on the appearance changes of motions when body and motion parameters are changed. Despite the above two research are a bit early, the criteria of their sports motion analysis are kept and wildly used in many related researches [22–25]. For instance, Yeadon [20, 21] has done research on how diving and somersault motions change when the motions are launched at different timings by using physical simulation. Though such tools are useful for the athletes to interactively visualize

possible results under different parameters, they can only assess the performance of sports that do not require complex maneuvers and strategies, such as jumping, high jumping, sky jumping, or somersaults. In many sports games, the performance depends not only on physical factors such as velocity, power and strength, but also on flexibility to switch from one motion to another and richness of the player's motions. This high-level information has not been used to visualize the skills of the athlete in previous research and it is the major difference between our work (in Chapter 3) and the afore-mentioned ones. In our research in Chapter 3, we combine the approaches of motion graph [26–28] and dimensionality reduction [29,30] to visualize high-level skills information of the athletes for the skill assessments.

2.1.2 Motion Graphs for Motion Modelling

The Motion Graph approach [7,26–28,31–33] is a method to interactively reproduce continuous motions based on a graph generated from captured motion data. Reitsma and Pollard [34] compared different motion graph techniques comprehensively. Heck et al. [35] further parametrized the motion space to control how the motions are generated by blending samples in the motion graph. Such an approach can be used for interactive character control such as that in computer games. When it comes to graph construction, [7,31] are the ones most similar to our method in Chapter 3. Min et al. [7] grouped similar postures and transitions into nodes and edges. Their focus was the motion variety of synthesized motions so they used generative models to fit the posture and motion data. Our focus is on skill visualization through the analysis of postures and motions so we can afford simpler and faster methods of analysis. Beaudoin et al. [31] cluster postures first then find motion motifs by converting the motion matching task into a string matching problem. Their priority was to find motifs that were representative while our focus is to visualize motion details and statistics to help people assess the skills. Xia et al. [36] constructed a series of local mixtures of autoregressive models (MAR) for modelling the style variations among different motions for real-time style transfer. They demonstrated style-rich motions can be generated by combining their method and motion graph.

Since the Motion Graph produces a lot of edges and nodes without any context,

it becomes difficult to control generated motion as the user wishes. Safonova and Hodgins [37] optimized the graph structure by combining motion graph and interpolation techniques to improve performance. On the other hand, works to resolve this problem by introducing a hierarchical structure were proposed [38]. These approaches add topological structures into the continuous unstructured data so that the motion synthesis can be done at a higher level. In a sport like boxing, it is possible to create a motion graph of semantic actions such as attack and defence, which is known as the action-level motion graph [39, 40]. A recent work by Hyun et al. [41] proposed *Motion Grammars* to specify how character animations are generated by high-level symbolic description. Such an approach can be used with existing animation systems which are built based on motion graphs. Ho and Komura [42] built a finite state machine (FSM) based on Topology Coordinates [43] for synthesizing two-character close interactions. The sparse graph structure can be used for controlling the movement of virtual wrestlers in computer games. The purpose of these approaches, however, is motion generation rather than the visualization of the player's skill.

In our research in Chapter 3, we adapted a hierarchical motion graph structure called the Fat Graph [38] on the action level to analyze the connectivity and the variety of a captured motion set. In a fat graph, similar nodes are grouped together as fat nodes, and similar edges are grouped as fat edges, allowing better organization of motion data. The filtered motion graph is a variation of the Fat Graph, in which the temporal relationship between poses is considered [44]. Such a structure, however, is targeted for motion reconstruction and analysis rather than visualization [45].

2.1.3 Statistical Motion Modelling

Dimensionality reduction methods have been proposed to visualize the overall structure of captured motions. Grochow et al [29] proposed a method to project the 3D motions of a human onto a 2D plane, and further reconstruct 3D motions by mapping arbitrary points from the 2D plane back onto 3D joint space. PCA [30] and ISOMAP [46] are proposed to map the motions onto 2D planes. Due to the high

variation of human motion, local PCA that considers only a relevant subset of the whole motion database to generate a locally linear space is proposed [47,48]. One can generate motions from arbitrary points on the plane by interpolating the postures of the original motion. Meanwhile, non-linear methods [49, 50] and Deep Learning [51] have also been used to reduce the dimensionality of motions. The Gaussian Process [52] and the mixture of Gaussian Processes [52] can be used to represent a set of human postures with a small number of Gaussian parameters. However, such methodologies do not consider the connectivity structure of the motions. We apply dimensionality reduction to our graph structure to visualize the connectivity structure of captured motions on a 2D plane.

Other researchers have focused on the connectivities of motion/actions by methods such as Markov models. Hidden Markov Model (HMM) [53] has been widely used in analyzing and synthesizing human motion. Typically, the hidden states of the HMM refer to the distribution of body poses and the dynamics of the motions are represented by the transitions between the hidden states. The parameters of the HMM can be subsequently learned from training data by the Expectation-Maximization (EM) algorithm. Hara et al. [54] proposed to model daily activities using HMM in intelligent house. Françoise et al. [55] proposed to use HMM models for analyzing Tai Chi motion sequences. An early work proposed by Brand and Hertzmann [56] proposed to learn the dynamics of human motion using HMM in their motion style synthesis model. Tango and Hilton [57] proposed to learn a HMM model from captured human motion for synthesizing in-between frames in keyframe animation. Ren et al. [58] presented a data-driven approach for quantifying naturalness of human motion including those synthesized by HMM. While existing work focuses on finding statistical distributions of motions, our focus is on visualizing the motion richness and the transition dynamics for skill assessments.

2.1.4 Understanding of Boxing Motion Analysis

In my research topics on Chapter 3, the boxing motions serve as a demonstration. This is because boxing motions are complex and hard to analyze. Our system is generic and can be introduced to different sports. Here is an example of human

criteria to analyze the quality of the boxing motion [59]. Professional boxers are trained first on some basic postures e.g. defense, stepping and attack, threading through which are the transitions carried out by the boxer based on the strategy and the opponent in a match. A good boxer can carry out a variety of transitions at will to achieve the best outcome. Such information serves as indicators for assessing the skill level of a player and the same principle applies to many other sports e.g. tennis, fencing, etc. Unfortunately, there is a research gap in assessing the motions of the players from a higher-level point of view. Our graph-based representation on boxing motions helps us to capture such ambiguous information.

2.2 Interaction-based Motion Retrieval and Analysis

Synthesizing and retrieving multi-people interaction motions is a challenging task in computer graphics. In such scenarios, the ambiguous spatial relationships between interacting people are important to describe the motion semantics. Failure to capture such relationships makes motion synthesis to be difficult to generate nature movement, and retrieval system to get wrong results. In recent years, numerous researchers have started to explicitly model the spatial relationship between interacting people and introduced it to the motion synthesis system to simplify the process of generating complex interaction motions [8,42,43,60] , and motion retrieval system to improve the retrieving accuracy [61–64]. Here, the conventional human-centered representations for human motion are first reviewed, and their major weaknesses are discussed. Subsequently, the interaction-based representations are reviewed, and the difficulties of applying them in motion retrieval and analysis are highlighted.

2.2.1 Human-centered Representations

There is a large body of research about analyzing and identifying human motion using human-centered features of body movement. In the early research of human motion retrieval, traditional approaches utilize kinematic features such as joint an-

gles [65] and joints position-based distance [66] to evaluate different types of motion. Dynamic features such as forces produced by specific joints provide another mean to identify human movement [67]. Derived dynamic features such as the center of pressure can enhance body stability analysis [68].

Although it is possible to analyze individual kinematic and dynamic features, understanding the logical significance of a motion requires the meaningful combination of them. Logical rules based on combined kinematic features can be used as the motion features in motion retrieval [69]. By exploiting the body hierarchy, kinematic features concerning body parts can provide a higher level evaluation [70]. Movement notation language known as the Laban notation can abstract a short duration movement [71].

To better reflect the semantic meaning of human motion and minimize the tedious manual design, machine learning algorithms based on joint-pair relationship features are introduced to train classification systems that recognize different type of motion [72,73]. Learning a distance metric based on a set of single-character posture feature enhances motion recognition accuracy [74]. Deep learning algorithms such as the convolutional autoencoders can automatically learn a manifold to represent a motion from a large amount of data [75].

While these human-centred representations have been effective for interpreting basic movement, they fall short in representing scenarios involving multiple interacting characters, which is one of the key components in daily movements. For example, simply considering features from individual characters makes it difficult to distinguish waving from high-five between two characters.

2.2.2 Interaction-based Representations

Recently, there has been a significant increase in research to analyze the interaction between multiple characters. The relationships among characters are considered important features to recognize the semantic meaning of high-level interactions.

Relative kinematic features such as relative joint distance are proposed to represent the interaction between two characters [76]. The concept of kinematic-based logical rules can also be extended to represent inter-character kinematic features [69].

However, the feature dimension increases exponentially when considering multiple characters. While feature selection [72, 76] can be used to maintain a reasonable feature dimension, the optimal set of feature depends on the type of interactions. It is difficult to find a globally optimal set of low dimensional feature to represent all interaction types.

By considering the skeleton hierarchy of the interacting character as a number of strings, the *Gauss Linking Integral* is used to represent how these strings wrap around each other, thereby representing the interaction of the two characters [77]. Such a representation can be used to synthesize movement by considering close interaction [48], as well as motion indexing and retrieval [78]. However, the representation cannot effectively represent long-distance interaction such as one character avoiding an attack from another.

The *interaction mesh* has been proved to be a robust interaction representation [8]. It considers the joints of the interacting characters and applies Delaunay Tetrahedralization [79] to generate a mesh structure that indicates spatial proximity. Using the interaction mesh, interaction among characters can be adapted according to the user-defined criteria or environment changes [8, 80]. The structure is used in robotics to represent the interaction between a robot and the environment for movement adoption [68] and control [81].

There are some attempts to apply interaction mesh in interaction retrieval [62, 63]. However, the results are not satisfying. The major difficulties is that the topology and dimension of the interaction mesh changes over time depending on the posture of the interacting characters. In particular, a small change of body joints can result in a significant topological change of the mesh structure, making it difficult to compute the difference between two interaction meshes. Previous works attempt to solve the problem by breaking the distance function into two parts. For the edges that co-exist in two interaction meshes, a traditional geometry-based distance function is applied. For those edges that do not co-exist due to topological difference, the algorithm in [63] assumes zero distance, while that in [62] simply counts the total number. Since the two parts of the distance function have different nature, forcing them together creates inconsistent results.

In Chapter 4, a new unified framework is proposed for the retrieval and analysis of interaction-based motion. We adapt the interaction mesh structure due to its robustness and propose to compare two topologically different interaction meshes using the Earth Movers Distance [82]. Our method in Chapter 4 can discover the intrinsic similarity between interactions can discover the intrinsic similarity between interactions and produce superior results compared with existing work.

2.3 Crowd Motion Simulation

Crowd motion simulation has been widely studied under different methodologies including agent-based models [83–85], fluid-based models [86], optimization [12, 87, 88] and data-driven models [89,90]. Comparatively, the ease of control deserves more attention and starts to attract more research work. Crowd control has been widely used in many areas such as entertainment production and urban planning, where two central issues are control and simulation. Related to our research in Chapter 5, there are mainly three sub-fields where we draw our inspirations upon: gesturing on multi-touch devices, crowd motion control and formation control.

2.3.1 Gesture Recognition on Multi-touch Device

Since the invention of multi-touch devices, gestures have provided a rich capacity of control input design. Gestures can be sequenced to express complex control purposes and are typically represented by time series of positions and velocities. For any pre-designed stroke patterns, there are some user input variations. Spatially-based control design [91–93] mainly targets on recognizing stroke patterns out of variations to improve the expressibility, but with limited understanding on the time dependencies between strokes. To model temporal or semantic dependencies, rule-based systems such as gesture formalisation [94], grammar [95, 96], state machines [97] or syntax [98] are proposed. However, they either lack the accommodation of user input variations or do not generalise well.

Among the previous research works, Lü & Li's work [99] is most related to ours. They present a set of features based on translation, rotation, and scaling of a user's

finger configurations to encode strokes and recognise gestures. We use a similar stroke representation. However, while their work uses these features to form a state machine to recognize a predefined gesture, we use them to represent gesture data and design our recognition algorithm. For instance, our minimum oriented bounding box feature is effective at distinguishing control signals, which is previously not possible using only the set of features proposed in [99].

2.3.2 Crowd Motion Control

Crowd motion control has been studied extensively, including controlling the whole crowd [100, 101], subgroups [102, 103], sets of control points [104], and the style [11, 89].

Field-based control focuses on the design of guidance fields in the environment. Vector fields are typically used to guide each subgroup of the crowd [105]. In [106], the user can control crowd motion by adding anchor points to indicate their moving directions, with which a vector field is generated. In [107], the movement of the agents is generated by the guidance field that is sketched by the user or extracted from video. Such a field is used to construct a navigation field that refines the flow of the crowd by avoiding collisions in the environment. While these approaches enable the user to control the movement of a crowd easily, the main focus in their work is controlling the crowd by hand-crafted or extracted fields. In contrast, our method in Chapter 5 focuses on a data-driven approach that maps control gestures to basic crowd motions to enable an intuitive and interactive control scheme. This is facilitated by learning a statistical model from the crowd motions using GMM.

Mesh-based control is another control scheme that evaluates the crowd movement and formation using mesh deformation. Utilizing a single-pass algorithm, crowd movements can be evaluated based on a deformable mesh [108,109]. It is also possible to interactively edit large-scale crowd while maintaining the spatial relationship between individuals [12]. Voronoi diagram can be used to represent the spatial relationship between different agents and organize the crowd movement in constraint space using Torso Crowd Model [110].

One particular problem of exiting methods is the lack of high dimensional control

signals that can be used to define the movement details of a crowd that consists of multiple sub-groups. Existing methods typically employ multiple levels of control rules, such that the user can define the overall crowd movement first, and define the details of sub-group later. Instead, we decide to embed the control mechanism into our learned mapping between the control signal and the corresponding crowd motions. It solves the problem of potentially contradictory control objectives on different levels, such as different overall crowd and sub-group targets.

2.3.3 Formation Control

Formation control is a technique to control the movement of crowds while maintaining formations. A significant number of papers propose to represent the shape of the crowd by modelling the geometric relations between individual agents. Mesh-based methods are very popular because they can easily represent the formation and accommodate some randomness due to individual motions by controlled mesh deformation. Laplacian mesh editing [111] controls and combines existing crowd formations into larger scale crowd animation [104]. An intermediate 2D mesh between user input and crowd motion can be defined so that crowd formations are controlled by simple user gestures [108, 109]. Spectral analysis smoothly transforms the crowd from one formation to another which is represented by Delaunay tetrahedral meshes [112]. A local coordinate system called formation coordinates maintains the adjacent relationship between individuals in the crowd [101]. More variants of these methods can be found in [113–116].

The *Morphable Crowds* [11], which is based on data examples of different styles of crowd motion, is conceptually similar to our work in Chapter 5. While their method is based on modelling the positions of characters surrounding an individual in a crowd motion, our method models the full trajectories of characters in the crowd. Such a full modelling enables us to build up a precise control mapping from the input to crowd motions, which enhances the quality of controlling and synthesizing new crowd motions.

Path pattern that consists of flows of location-orientation pairs is also a good representation of crowd motions, which can be extracted from crowd video [117].

However, the representation is complicated and is too computationally expensive to be used for interactive control purpose.

2.4 Motion Capture Data Refinement

The major problem of using motion capture system is to deal with the noisy data obtained. Usually, the quality of the detected posture is of low resolution and suffers heavily from occlusion. It is likely to apply machine learning algorithms to enhance the quality of the data. The idea is to introduce a quality enhancement process that considers prior knowledge of the human body, which is typically a database of high-quality postures. In this section, we discuss how body information can be reconstructed from noisy data.

2.4.1 Posture Enhancement

The body tracked by motion capture system may contain inaccurate body parts due to different types of error. Simple sensor error can be caused by geometry shape of body parts and viewing angles. It is proposed to apply Butter-worth filter [118] or a simple low-pass filter [119] to smooth out the vibration effect of tracked positions due to this type of error. However, when occlusions occur, in which a particular body part is shield from the camera, the tracked body position would contain a large amount of error. Simple filter will not be sufficient to correct these postures.

As a solution, it is proposed to utilize accurately captured 3D human motion as prior knowledge and reconstruct the inaccurate postures from the motion capture system. In this method, a motion database is constructed using the carefully captured 3D motion. There are many open-source motion capture database such as *CMU Graphics Lab Motion Capture Database* [120]. Given a captured posture, one can search for a similar posture in the database. The missing or error body parts from the motion capture system can be replace by those in the corresponding database posture [121]. However, such a naive method cannot perform well for complex posture, as using only one posture from the database cannot always generalize the posture performed by the user, and therefore cannot effectively reconstruct the

posture.

More advanced posture reconstruction algorithms utilize machine learning to generalize posture information from the motion database [122–124]. In particular, the motion database is used to create a low dimensional latent space by dimensionality reduction techniques. Since the low dimensional space is generated using data from real human, each point in the space represents a valid natural posture. Given a partially mis-tracked posture from a camera, one can project the posture into the learned low dimensional space and apply numerical optimization to enhance the quality of the posture. The optimized result is finally back-projected into a full body posture. Since the optimization is performed in the low dimensional latent space, the solution found should also be a natural posture. In other words, the unnatural elements due to sensor error can be removed. The major problem of this method is that the system has no information about which part of the body posture is incorrect. Therefore, while one would expect the system to correct the error parts of the posture using information from the accurate parts, the actual system may perform vice versa. As a result, the optimized posture may no longer be similar to the original capture posture.

To solve the problem, optimization process that considers the reliability of individual body part was proposed [47]. The major different from this method compared with prior ones is that it divides the posture reconstruction process into two steps. In the first step, a procedural algorithm is used to evaluate the degree of reliability of individual body parts. This is by accessing the behaviour of a tracked body part to see if the position of the part is inconsistent, as well as accessing the part with respect to its neighbour body parts to see if it creates inconsistent bone length. In the second step, posture reconstruction is performed with reference to this reliability information, such that the system relies on the more parts with higher reliability. Essentially, the reliability information helps the system to explicitly use the correct body parts and reconstruct the incorrect ones. Such a system can be further improved by using Gaussian process to model the motion database, which helps to reduce the amount of motion data needed to reconstruct the posture [52, 125]. Better rules to estimate the reliability of the body parts can also enhance the system

performance [126].

2.4.2 Prior Knowledge

The major research focus of face and posture enhancement is to apply appropriate prior knowledge to improve data obtained in rum-time. In machine learning based algorithms, such prior knowledge is usually learned from a database, and represented in a format that can be efficiently used in run-time.

For motion enhancement, since human-motion is highly nonlinear with large variation, it is not effective to represent the database using a single model. Instead, many of the existing research apply multiple local models to represent the database, such as using a mixture of Gaussian model [52]. It is also proposed to apply deep learning to learn a set of manifolds that represents a motion database [127]. Pre-computing these models and manifolds are time-consuming, as it involves abstracting the whole database. Therefore, lazy learning algorithm is adapted, in which modelling of the database is not done as a pre-processs but as a run-time process using run-time information [47, 122]. During run-time, based on the user performed posture, the system retrieve a number of relevant postures from the database, and model such a subset of postures only. This method has two advantages. First, by modelling only a small number of postures that is relevant to the performed posture, one can reduce the computational cost of constructing a latent space. Second, since the subset of postures are relatively similar, one can assume that they all lay in a locally linear space and apply simpler linear dimensionality reduction to generate the latent space. This allows real-time generation of the latent space. With improved database organization, the database search time can be further reduced and the relevancy of the retrieved results can be enhanced [44, 128], such that real-time ergonomic and motion analysis applications can be preformed [128].

Figure 2.1 visualizes how prior knowledge can be estimated from the database. Each blue circle in the figure represents a database entry, and the filling color represents its value. The obtained prior knowledge from the scattered database entries is represented by the shaded area, which enables one to understand the change of value within the considered space. The left figure shows a traditional machine learn-

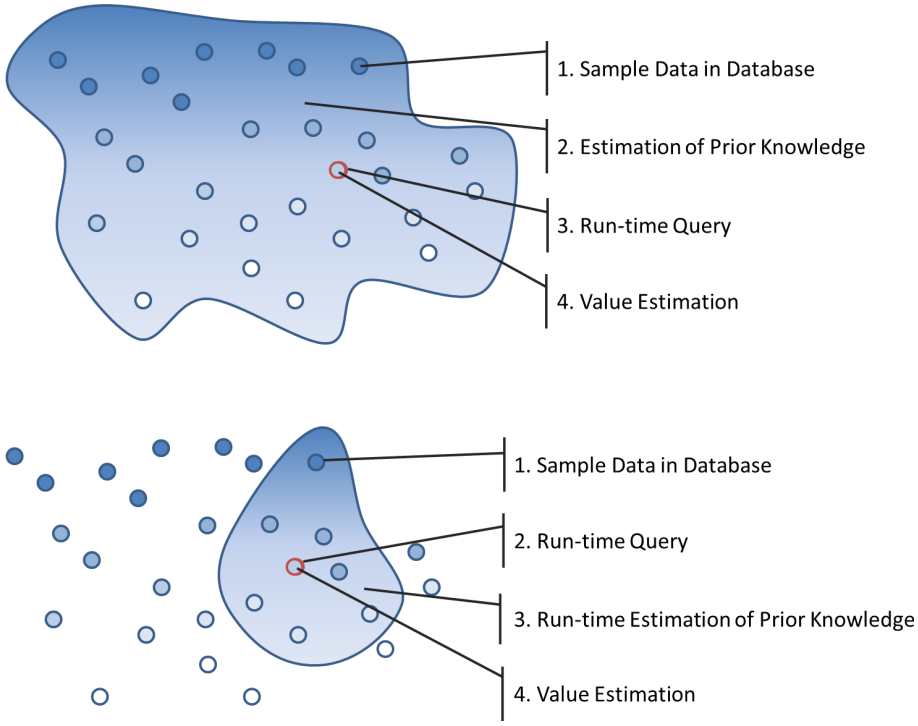


Figure 2.1: (Upper) Traditional machine learning that represents the prior from the whole database. (Lower) Lazy learning that represents the prior from a subset of the database based on the online query.

ing algorithm, in which prior knowledge is obtained as a pre-process, considering all database entries. During run-time, when a query arrives, the system uses the knowledge to estimate the corresponding value of the query. The right figure shows the case of lazy learning, in which prior knowledge is obtained during run-time. This allows the system to extract database entries that are more similar to the query, and estimate the prior knowledge with only such a subset of data.

2.5 Summary

The existing studies on motion analysis and synthesis techniques can accurately recognize and flexibly control the motion of a single character. In recent years, researchers focus on using such techniques in complex scenarios (e.g. multiple characters interactions and character-environment interactions). However, the results are still unsatisfactory. This thesis will focus on extracting high-level semantic meaning among the interaction and improve the results of motion analysis and synthesis on such complex scenes.

Chapter 3

Graph-based Human Motion Visualization and Analysis

Computer technologies [129] [130] [131] have taken on a crucial role in modern sports and health sciences, in revolutionizing the way to observe, analyze, and improve the performance of both amateur and professional athletes. Computer-managed weight lifting machines, treadmills and many other training equipments provide energy consumption or repetition and weight management in many sports clubs. Virtual reality technology has been applied in various training systems in baseball [132], handball [133] and tennis [134] to assist more professional sports activities. Nevertheless, these technologies are only able to analyze motions at a low level, i.e. recording the timing or repetitions of basic motions and comparing movement trajectories with those performed by better players. More advanced technologies are needed for personalized and higher-level analysis comparable to that from human experts.

In addition to the instantaneous movement features of the sports players, Experienced sports coaches consider high-level features such as the variety of actions and quality of transitions from one action to another. Taking boxing as an example, professional boxers have in basic actions such as defence, stepping and attack, threading through which the transitions are carried out based on the strategy and the opponent's reactions. The action transitions of a good boxer need to be flexible and contain great variety to achieve the optimal outcome. Such information often

serves as an important indicator in assessing the skill level of a player, and the same principle applies to many other sports such as basketball [41] and fencing [135]. Unfortunately, automatic systems for analyzing and evaluating sports motions at such a high level is very limited. Based on that observation, our research focuses on analyzing the higher level information of the sports motion, such as skillfulness, richness and unpredictability, which are consistent with human standards of evaluations. Then we visualize this high-level information in an intuitive way for people to understand. Our approach has several potentials, such as the motion evaluation system on sports training session, motion visualization and analysis system on evaluating the quality of motion capture data.

In this work, we propose a robust visualization system to address the above limitations, by representing motions as an interactive graph of high-level features, including the flexibility and richness of the actions as well as the transitions of actions. Although we use boxing as a demonstration in this work, our method is generic and can be applied to different sports. Our approach starts with capturing the *shadow boxing* training motion of a boxer, in which the boxer performs boxing with an imaginary opponent. An experienced coach can effectively assess the boxer's skill by watching the shadowing boxing motions. As a positive side effect, this method of motion analysis greatly reduces the complexity of motion capture due to occlusion and collision and has shown to be very effective in our system. The motion data is then processed and visualized in two different graphs: the posture-based graph and the action-based graph, for performance analysis.

In the posture-based graph, the semantic actions segmented from the captured motion are grouped into clusters based on a customized distance function that considers action specific features. Our system then automatically generates a motion graph structure known as *Fat Graph* [38], which uses nodes to represent groups of similar postures to start and end actions, and edges to represent groups of action. By applying dimensional reduction techniques, this graph can be visualized in a 3D space for performance analysis and evaluation. The transition capability of the boxer is visualized by the connectivity of the nodes, where the richness and preference of the actions are visualized by the edges in the graph. We further propose a

skill evaluation metric known as the *Connectivity Index* which evaluates the richness of actions and the flexibility of transitions according to the graph.

Whilst the posture-based graph focuses on the variety of basic postures and the transition flexibility between actions, the action-based graph mainly considers the richness of actions and the transition probability among them. The action-based graph is constructed as a customized Hidden Markov Model (HMM) [53], in which similar actions are grouped into clusters that formulate the nodes. The transition probability among actions is calculated and is expressed as edges between nodes. The graph is visualized in a 3D space, and the positions of the nodes and edges are optimized for better visualization. With such a graph, the pattern of action launching can be easily identified in order to assess the boxing strategy of the boxer. We further propose the *Action Strategy Index* to evaluate the unpredictability of action patterns according to the graph.

With the support of the proposed algorithm, the performance quality can be analysed as human experts usually do and the potential problems of a player can be readily identified for sport technical improvement. These edges are also colour and size coded, with colour indicating the sequence of continuous action pairs and size representing the probability of such transitions. Examples based on boxing, basketball and tennis motions are conducted to demonstrate the effectiveness of the proposed algorithm, and clear differences between the novice and experienced players are shown.

We conducted experiments on the motions captured from multiple boxers and evaluate their skills. The corresponding posture-based and action-based graphs were generated. As shown in Fig. 3.11, we can easily evaluate the skills of different boxers with our visualization system.

3.1 Contributions in This Chapter

There are three main contributions of this work:

- We propose a framework for high-level skill analysis through automatic motion analysis and visualization. Given a captured motion from a sports player,

our system automatically segments the motion into semantic action units and constructs two graph structures.

- We propose the posture-based graph, which is a variant of the Fat Graph, to visualize the skills according to different standard postures for launching and ending actions. It allows the user to identify the correctness of standard postures and the diversity of actions. We further propose the Connectivity Index that evaluates the richness of actions and the flexibility of transitions.

- We propose the action-based graph, which is a variant of the Hidden Markov Model (HMM), to visualize the skill according to different groups of action. It allows the user to identify the preference of actions and their transition probability. We further propose the Action Strategy Index to evaluate the unpredictability of action patterns.

3.2 Overview of Our Method

The overview of our method is shown in Fig.3.1 and the system is divided into two parts: the motion data pre-processing part and the Posture/Action Graph visualization part. The first part captures, analyze and organizes the long motion sequence of a character and represents it into semantically graph structure. The second part visualizes the graph representation by projecting entities to appropriate 2D space via dimension reduction techniques, and the resultant graph can be rendered. The quantitative results of different skill level motions can be achieved by analyzing corresponding graph topologies.

3.3 Motion Data Pre-processing

We first capture the motion required for analysis using motion capture systems. Then, we propose an automatic system to segment long sequences of captured motion into meaningful actions, which are used as building blocks of our posture-based and action-based graphs.

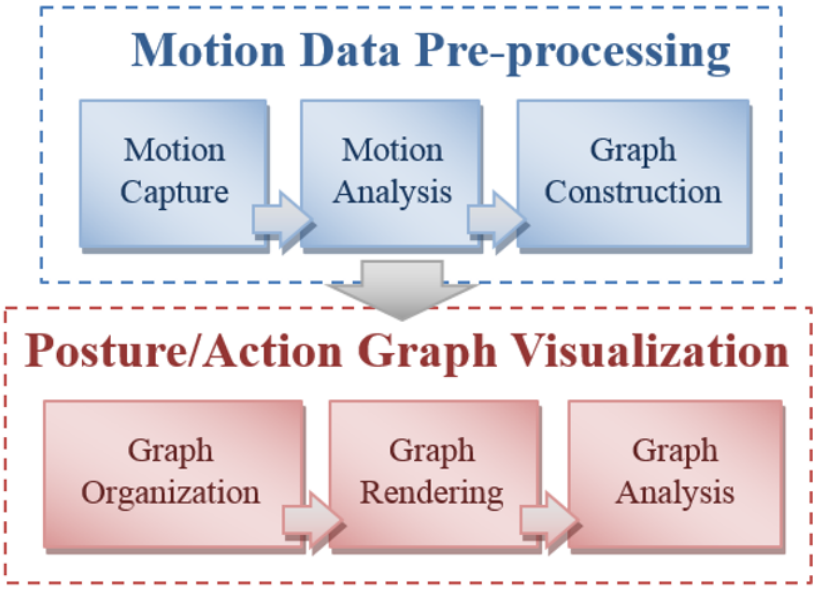


Figure 3.1: *Overview of graph-based human motion visualization and analysis system*

Here, we follow the definition from [40], in which a *motion* is considered to be a raw sequence of captured human movement, and an *action* is considered to be a short, meaningful segment of movement within a motion. In the field of boxing, an action can be an attack (such as a "left straight", "jab" or a "right kick"), a defence (such as "parries", "blocking" or "ducking") , a transition (such as "stepping to the left", "stepping forward" or "back step"), or any combination of them.

Postures and actions are good entities for skill visualization, as sports players typically plan their strategies and evaluate their performances with such terms. For example, a boxer typically thinks about what sort of attack/defence/transition should be launched during a match. A coach typically evaluates the overall strategy in the action level, as well as how well individual postures and actions are performed.

3.3.1 Motion Capture

Although it would be best to capture the motions of all players in multi-player sports because the data would reflect the features of the motions, capturing multiple players remains difficult due to the occlusions and collisions among players. Fortunately, it is

possible to only capture individual motions for our purposes without compromising the true motion characteristics. In boxing or any other martial arts, there is a training practice called "shadow boxing". The boxer imagines a boxing session with another boxer, and launches boxing actions to interact with such an imaginary opponent. The boxer launches not only offensive actions such as punching, but also defence, stepping, and the consecutive combination of all such actions. There are similar practice methods in basketball and soccer as well, in which players use the ball to conduct various techniques in the court, imagining that their opponents are trying to take the ball away from them. The players thus perform various actions to keep the ball and trick an imaginary opponent. This technique has also been used by coaches for skill assessment hence is suitable for our analysis. We employed an optical motion capture system to acquire the performed motion as shown in Fig. 3.2 as it was less intrusive and highly accurate. Also, we preferred to capture long and continuous clips of motion, such that the player could perform the motion in a natural manner.

To evaluate the performance of our system for real-people sports motions, we produce 4 motion sequences of shadow boxing motions performed by 4 boxers with different skill levels, which were captured at the University of Tokyo. We collect around 6 minutes of boxing from 4 sets of shadow boxing motions from novice to professional, as shown in Fig. 3.11. Averagely each motion sequence has around 1.5 minutes at speed of 30 frames per second. So each sequence has around 2700 frames.

3.3.2 Motion Analysis

After data capture, the system automatically segments meaningful actions from the raw captured motion and identifies the effective joints that contribute the most to the semantic meaning of the actions.

For boxing motions, we observed that actions normally start and end in a double supporting state (i.e. both feet touching the floor), as the state is usually dynamically stable. We detect such a state by monitoring the feet height and velocity and setting corresponding thresholds. This allows us to segment the raw captured

Figure 3.2: *Motion capture on a single human. The shadow boxing motions of several boxers were captured using an optical motion capture system.*

motion into a set of *movement segments*, which are the periods between every two successive double supporting states, as visualized in Fig. 3.3 Upper.

We also observed that actions normally require a relatively larger force to be performed, such as a punch or a step. We define periods with a high-level of force exertion as the *activity segments*. Since force is proportional to acceleration, these segments can be found when the sum of squares of acceleration of all joints is above a threshold, as visualized in Fig. 3.3 Middle. The threshold is statistically obtained from the acceleration profile of the motion.

Finally, the actions are composed by using the movement segments as the building blocks. The timing and the duration of the activity segments are used to determine if the movement segments should be merged together to form longer segments. Regarding the relationship of the movement segments and the activity segments, there could be three possible cases: (1) There is no activity segment inside a movement segment. In this case, the movement segment becomes a single action of pure body transition. (2) There is one activity segment inside a movement segment. In this case, this movement segment becomes an action with a special activity. (3) There are one or more activity segments lying across successive movement segments.

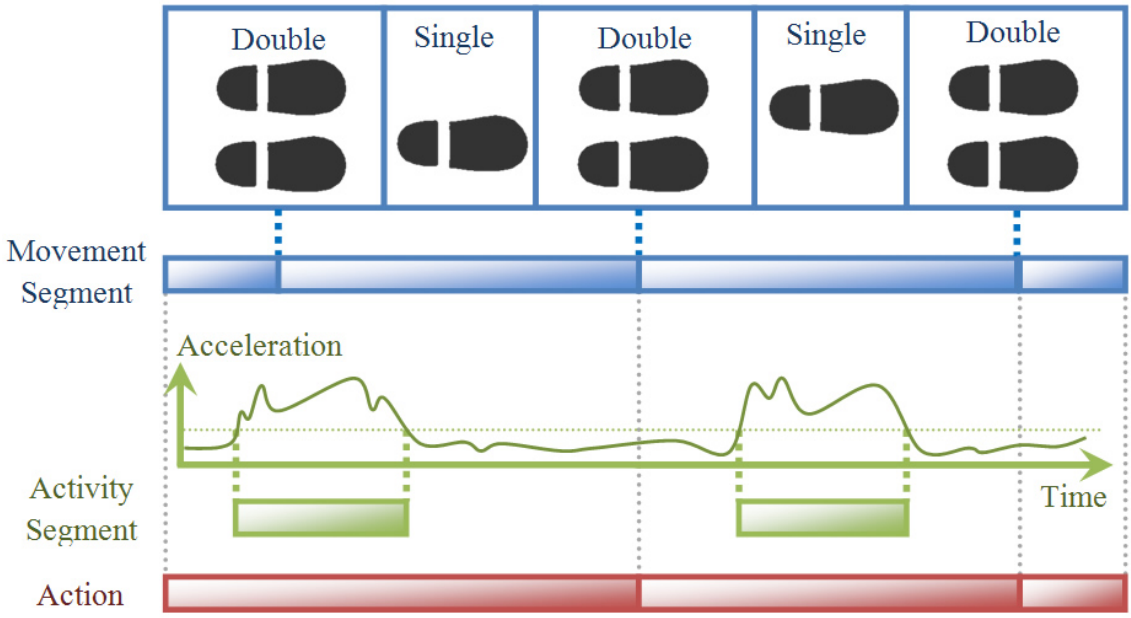


Figure 3.3: *The motion segmentation framework. Upper: The movement segment is defined as the period between two double support supporting phases. Middle: The activity segment is defined as the period with high acceleration. Lower: The action is the combination of movement segment and activity segment.*

In this case, the movement segments containing activity segments at the border are merged to form an action as visualized in Fig. 3.3 Lower. Note that due to this merging process, the resulting action may contain multiple activity segments. In our system, we implement an optional step to filter very short actions that are likely to be generated due to the noise of the supporting feet.

We define the *effective joints* to be the set of joints to represent an activity segment. In case (1) above, since the actions contain no special activities, the pelvis is considered to be the effective joint. In case (2) and (3), the effective joint is the joint that contributes the most to the sum of squares of the acceleration in the activity segment. In more complicated actions such as left-right combo punches, there may be multiple effective joints as there are multiple activity segments. Such joints are used in later processes to evaluate the similarity of actions.

3.4 Posture-based Graph

The posture-based graph focuses on evaluating the common postures that are used to start and end actions. In such a graph, the nodes represent similar postures and the edges represent similar actions. It allows us to evaluate the consistency of common postures and the diversity of actions.

3.4.1 Graph Construction

We adopt a Fat Graph structure [38] in the action level [40] to generate the posture-based graph, as it can effectively simplify the graph representation by grouping similar postures and actions together. The Fat Graph was originally proposed for motion synthesis, and thus it is not optimized for skill visualization. We redesign the algorithms to generate nodes and edges in the Fat Graph for our purpose.

3.4.1.1 Fat Nodes

In our system, the nodes of the Fat Graph, known as Fat Nodes, are the common starting or ending postures of the actions. We design an unsupervised clustering scheme for grouping all starting/ending postures into a finite set of posture groups, which avoids additional labour for posture labelling and grouping. Specifically, we used k-means to cluster postures. The distance between two postures P_0 and P_1 is defined as:

$$D(P_0, P_1) = \sum_{i=0}^{i=i_{total}} |\theta_0(i) - \theta_1(i)| \tag{3.4.1}$$

where $\theta_0(i)$ and $\theta_1(i)$ represent the 3D joint angle of the joint i in posture P_0 and P_1 respectively, and i_{total} is the total number of joints. Regarding the cluster number k, a large k would result in many clusters (Fat Nodes), which unnecessarily increases the complexity of the graph. A small k will cluster very different postures into the same node, defeating the purpose of the graph. Therefore, we set up a posture difference threshold empirically based on experts' suggestions. Based on lots of tests, we found a suitable Euclidean distance threshold to distinguish different types of postures in boxing motions. For different types of motions, this process should be repetitively done. Then, we iteratively search for a proper k by initially setting

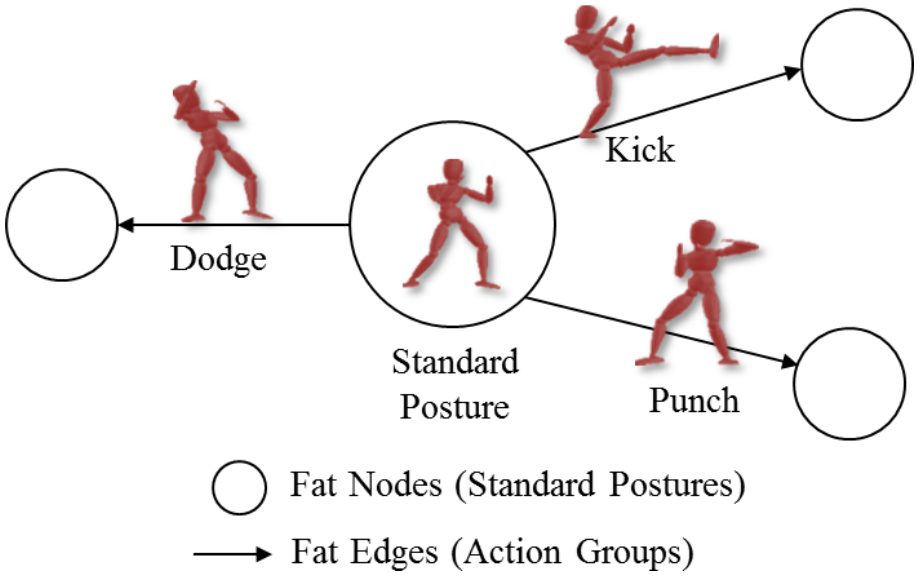


Figure 3.4: *Posture-based Fat Graph on single character's boxing motion. The Fat Node represents the standard fighting pose. The three outgoing Fat Edges represent different action groups.*

$k = 1$ and incriminating k by 1 until we find the first value of k that does not violate the distance threshold. After clustering, we use the mean posture of a group to represent the corresponding Fat Node. The nodes in the graph represent the set of standard postures which the player starts the various actions from. In the case of boxing, they are usually the fighting postures that the boxer uses to guard his/her face against the opponent, with both feet landing on the ground and keeping shoulder width apart.

By evaluating the Fat Nodes alone, we can already tell if a boxer has multiple unnecessary standard postures, or if any standard postures contain potential weakness. In general, experience players have fewer Fat Nodes, such that they can start actions in a standard posture effectively without the needs of shifting to other ones. Novice players sometimes may have a particular Fat Node for some particular actions. This is discouraged in boxing training as such postures hint the opponent as to what actions are going to be launched.

3.4.1.2 Fat Edges

We design the edges of a Fat Graph, known as Fat Edges, as directional edges that represent groups of similar actions. Each edge points from the Fat Node representing the starting posture to that representing the ending posture.

Similar to the Fat Nodes, we implement an unsupervised clustering algorithm to group similar actions into Fat Edges. We use k-means to cluster the actions and search for the smallest acceptable k for a given distance threshold. We define the actions distance according to the trajectory of the effective joints as explained in Section 3.3.2. This allows accurate clustering of actions and ensures that the effects of the effective joints are not smoothed out by other joints.

Formally, the distance between two actions A_0 and A_1 is defined as:

$$D(A_0, A_1) = \begin{cases} \infty \\ \text{if } A_0 \text{ and } A_1 \text{have different sequences} \\ \text{of effective joints} \\ \sum_{j=0}^{j_{total}} \sum_{f=f_{start}}^{f_{end}} [A_0(j)(f) - A_1(j)(f)] \\ \text{otherwise} \end{cases} \tag{3.4.2}$$

where $A_0(j)(f)$ and $A_1(j)(f)$ represent the 3D positions of effective joint j in frame f in the action $A - 0$ and A_1 respectively, j_{total} is the total number of effective joints in the actions, f_{start} and f_{end} are the starting frame and ending frame of the considering effective joint. In case two effective joints with different duration are to be compared, the shorter one is linearly scaled to the duration of the longer one.

In the field of boxing, a Fat Edge typically contains a set of actions with basic attacks or defences such as “straight punch”, “hook punch”, “parry”, or a set of complex actions combining several attacks and defences. Since member actions in a Fat Edges have to share the same starting and ending Fat Nodes, if an action group contains multiple starting or ending poses, it is sub-divided into multiple Fat Edges.

Again, by only looking at Fat Edges, one can tell the differences between experienced and novice players. Experienced players normally have Fat Edges with similar numbers of actions, as they have mastered a large variety of boxing actions

and can switch between them effectively using a small number of stable transition maneuvers. Novice boxers tend to have a larger number of Fat Edges but each with a small number of actions, due to the inability to reproduce boxing actions consistently. Fig. 3.4 shows the relationship of Fat Nodes and Fat Edges.

3.4.2 The Connectivity Index

It requires deep knowledge and years of experience to assess one's skills in sports. Here, we make use of the posture-based graph and define an index representing the skill level, allowing more objective and efficient skill assessment.

In many types of sports, there are two important skill indicators. The first one is the richness of the actions that indicate the resourcefulness of a player. The other is the flexibility of transitions between states so that the player can switch between different states at will. Our posture-based graph captures both of the indicators. The richness can be represented by the number of Fat Edges, indicating how many kinds of maneuvers the player has. The flexibility is indicated by the connectivity of the graph, which is inversely proportional to the number of Fat Nodes. A fully connected graph shows great flexibility because there are transitions between any two nodes.

Notice that these two factors are somehow contradicting. In general, the richer the actions are, the greater the number of different starting and ending poses is hence the poorer the connectivity of actions is. Independently considering either of them would not suffice. We therefore define a *Connectivity Index* that evaluates both the action richness and the action flexibility of a player:

$$CI = \frac{\text{Number of Fat Edges}}{\text{Number of Fat Nodes}} \tag{3.4.3}$$

To accurately reflect the skill level of a player, in our implementation, we do not consider Fat Nodes that are not intentionally created. For example, one of our boxers tripped over during a session. While it is good that our system can objectively pick up the posture generated by the accident, we do not include the corresponding Fat Nodes when calculating the Skill Index. Also, we only consider Fat Edges that are consistently performed, as those having only a small number

of member actions could be randomly performed actions. Empirically, we consider edges having more than 2 member actions.

3.4.3 Visualization System

Here, we describe the design of our visualization system to visualize the posture-based graph in an effective manner. We also introduce interactive features for the user to view the graph with different levels of details.

The posture-based graph consists of high dimensional Fat Nodes (groups of similar postures of many degrees of freedom) and Fat Edges (groups of similar actions in the spatial-temporal domain), which presents a challenge for visualization. To reduce the dimensionality for better visualization, we propose two different schemes for nodes and edges due to their different nature in this graph. Specifically, we project the Fat Nodes on a 2D space using Principal Component Analysis (PCA) as it creates a more consistent low dimensional space compared with other methods sush as Finite-State-Machine [42,43] or low-dimensional motion manifolds [127]. We represent Fat Edges with 2D curves and augment the curves with a combination of geometric primitives to visualize the action features.

3.4.3.1 Visualizing Fat Nodes

Although the degree of freedom (DOF) of human postures are high dimensional (45 DOF in our system), they are intrinsically dependent on each other [29]. In fact, the Fat Nodes can be represented effectively in a 2D space where nodes of similar postures are located together while those of different postures are located far apart. This allows viewers to easily understand the relationship between postures.

For each Fat Node, we obtain the mean posture as its representation. Given a set of postures, we apply principal component analysis (PCA) to reduce the dimensionality to 2. Essentially, we calculate the covariance matrix to evaluate the intrinsic dependency of the dimensions. We then calculate the eigenvectors from such a covariance matrix, and use the two eigenvectors with largest eigenvalues to form a feature vector.

PCA is used as it has shown to be effective on human postures [29]. However,

since we only have a small number of postures, we believe other dimensionality reduction techniques would also work.

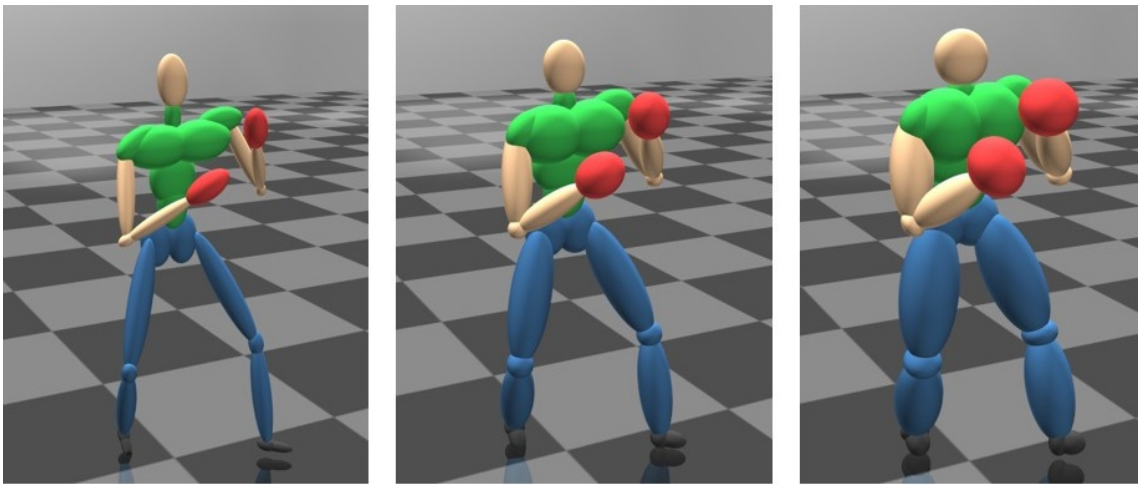


Figure 3.5: *The fatness represents the size of the node. From left to right, the character becomes larger as the size of the nodes increases.*

We render the mean posture of each Fat Node onto a 2D X-Z plan. This allows the user to identify inappropriately performed postures. In boxing, novice boxers sometimes lose track of their boxing rhythm, and hence start or end an action with an inappropriate posture. We use the fatness of the character to represent the number of member postures in the node, as shown in Fig. 3.5. This allows the user to easily observe the postures that the player usually uses to start actions.

3.4.3.2 Visualizing Fat Edges

Here, we explain how to visualize the Fat Edges, which contain information of groups of similar actions.

We do not apply dimensionality reduction techniques directly on the action data itself because the low dimensional projection would be very complex. Instead, we propose to visualize each Fat Edge by a 2D curve that represents its mean action on the X-Z plane. We optimize the angle and sign of these curves to minimize occlusion. For edges with a starting node different from the ending node, the edge angle is fixed. The only adjustable variable is the bending side of the curves, which is essentially the sign of the curves. For those with the same starting and ending

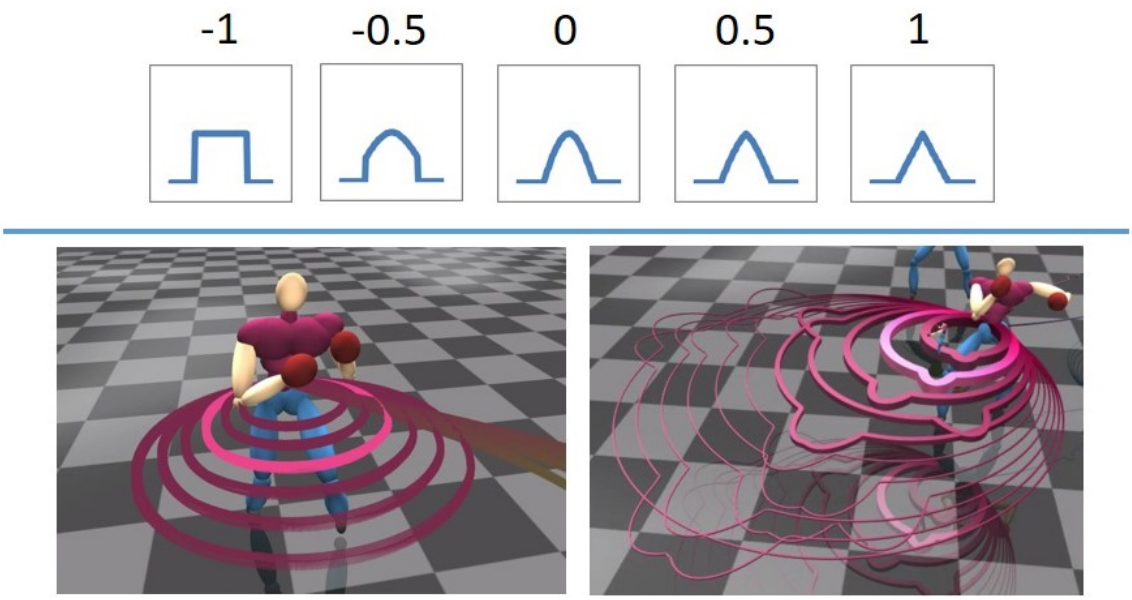


Figure 3.6: *Visualizations of Fat Edges in 1D space. The geometric patterns for landmark values between -1 and 1. Each pattern represents a landmark posture in an action. (Lower) Comparison of visualization without/with the patterns. Each curve represents a group of action. The right image shows the uses of landmark patterns to identify different types of action.*

node, both edge angle and bending side can be controlled. We optimize the signs and angles of the edges in a greedy manner such that they would blend towards a less dense region of the graph.

To visually distinguish between different Fat Edges, we add some geometric patterns to the 2D curves. We collect the high-energy frames of all actions and project them onto a 1D space using the PCA system explained in Section 3.4.3.1. Since the high-energy frames of different actions are typically distinguishing postures, the projection essentially maps all action features onto a normalized 1D space in the range of $[-1.0, 1.0]$. To visualize the value in this 1D space, we design some geometric patterns for landmark values -1.0, -0.5, 0.0, 0.5 and 1.0 as shown in Fig. 3.6 Upper. The patterns to represent values between two landmarks are obtained by linear interpolation between nearby landmarks.

We further represent the number of member actions in the edge by the thickness of the curve. This allows the user to identify the player's preferred actions. For

instance, if a boxer relies heavily on single straight punches, the Fat Edge for such action will be unreasonably thick, while edges for other attacks will be relatively thin, which demonstrates a potential lack of diversity attacking strategies.

Through the comparison between Fig. 3.6 Lower Left and Lower Right, it shows that adding the geometric patterns gives a better visualization of actions in the edges. This strategy presents an intuitive way to show the players preferences over actions of different complexity.

3.4.3.3 Interactive Features

We integrate some interactive features in our system to display relevant information based on user input. When the user selects any specific entities in the graph, related information will be shown.

When a Fat Node is selected, its corresponding Fat Edges will be highlighted for easier observation. Information about the number of members in that node, the number of outgoing edges, and a number of incoming edges are displayed in a sub window. When a Fat Edge is selected or highlighted (because of a Fat node selection), we render the member actions included, such that the user can understand the content of the edge.

As an example, in Fig. 3.7, there are three Fat Nodes indicated by red arrows and numbered as 1, 2 and 3, each visualized as a character with a mean posture in the node. The sizes of the nodes are indicated by the body fatness. Node 1 is represented by the most muscular character, which indicates the largest node size. Nodes 2 and 3 are far thinner. Fat Edges are rendered as curves between nodes such as the ones shown by 4 and 5. The thicknesses of the edges indicate the frequency of the actions taken. Edge 5 is thicker than edge 4, suggesting that this boxer takes action 5 more often. In addition, an edge can be smooth like a circle or bumpy with geometric patterns. A single pattern means one activity segment such as a single punch, while multiple patterns indicate a series of activities such as a combo attack. Our system also supports interactive features. Fig. 3.7 is the result when the user selects Node 1. All the edges starting from this node are highlighted, each with a small character performing the action on it.

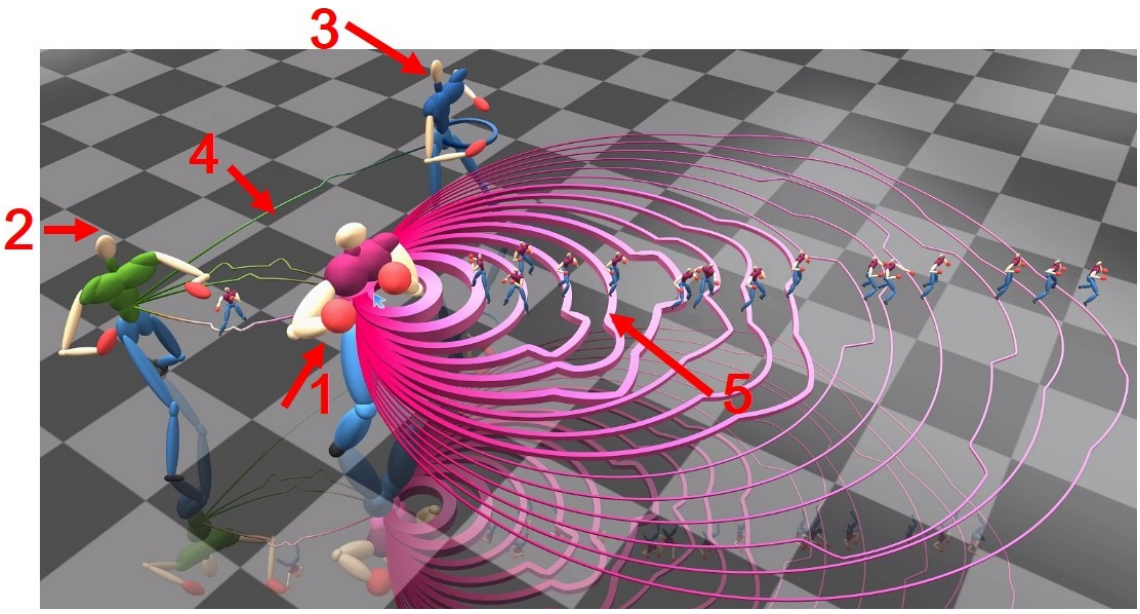


Figure 3.7: *The posture-based graph of the Boxer S. 1, 2 and 3 are Fat Nodes. 4 and 5 are two Fat Edges. 4 connects Node 2 and Node 3. 5 connects Node1 to itself.*

3.5 Action-based Graph

The action-based graph focuses on evaluating the transition probability from one action class to another. In such a graph, the nodes represent groups of action with similar activity segments. The edges represent the transition probability between two action groups. It allows us to evaluate the pattern of launching actions and extract the strategy of the boxer.

3.5.1 Graph Construction

We use the hidden Markov model (HMM) to organize the captured motion, as it has been shown effective in modelling human motion. In the domain of character animation, HMM has been mostly used in the posture level to create motion graphs [27]. We adapt the graph into the action level such that we can visualize the transition probability among actions.

The nodes of the graph represent different action groups. We apply Equation 3.4.2 to group the captured actions into a number of action groups with k-means

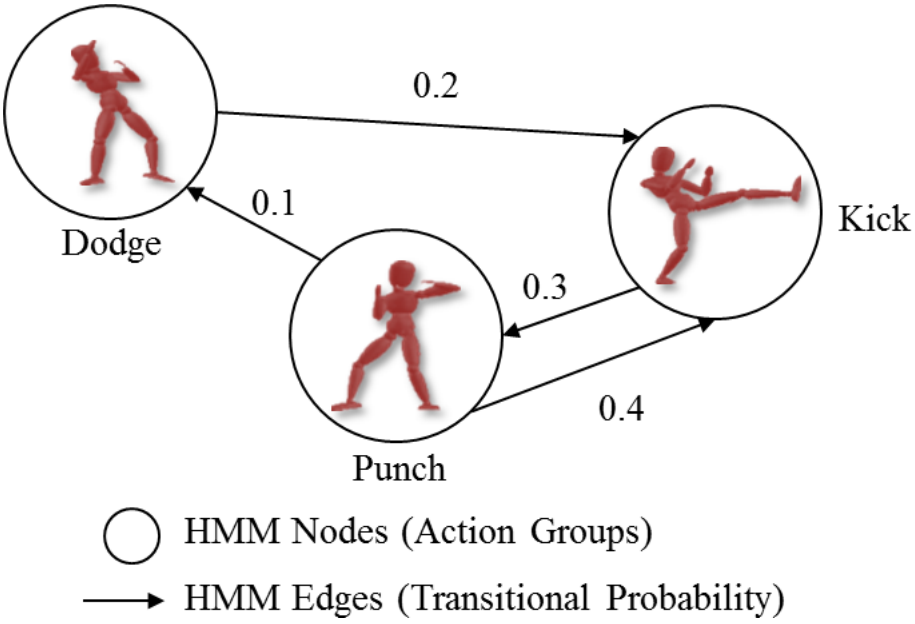


Figure 3.8: *Motion representation using HMM. The three HMM nodes represent action groups. The HMM edges represent transitional probability between them.*

clustering. The process is similar to that in Section 3.4.1, in which we define a threshold based on expert knowledge, and then incrementally increase the number of classes until the threshold is met. We denote k' as the total number of groups, $|G_i|$ as the number of actions in the i^{th} action group (which is used in the visualization system for visualizing the fatness and the placement of the node and will be described later).

The edges of the graph represent transitional probability from one action group to another. To obtain the transitional probability, we go through the sequence of actions in the captured motion and count the number of occurrences for an action belonging to group i to be followed by another belonging to group j, which is denoted as c_{ij}. The transition probability of action group i to action group j is defined as:

$$T_{ij} = \frac{c_{ij}}{\sum_{m=1}^{k'} \sum_{n=1}^{k'} c_{mn}} \tag{3.5.4}$$

where the denominator represents the total number of transition in the whole motion. Notice that i may be equal to j. In such a case, two actions of the same action group are launched successively.

The concept of the action-based graph is shown in Fig. 3.8. In general, experienced boxers tend to have a more evenly distributed transitional probability across

all actions, which means that there should be edges connecting all the nodes. This indicates that the boxer's pattern is dynamic and cannot be easily predicted by an opponent. Conversely, novice boxers may have limited edges and some thick edges connecting two nodes, which means a high probability to launch those two groups action consecutively. An opponent may discover such a pattern and counter-act in advance when the first action is observed.

3.5.2 The Action Strategy Index

In many sports, the unpredictability of action patterns is an important skill indicator. Experienced players would diversify their action patterns such that their opponents cannot predict the next action. However, novice players tend to perform actions based on predictable patterns (i.e. the sequence of actions to be launched continuously), which can be easily identified. For example, a novice boxer usually perform two straight punches successively. This is because the boxer is not able to link different types of punches fluently, and therefore would perform the simplest punches again and again. The proposed action-level graph allows easy observation of boxing patterns, as we can visualize the transitional probability among actions. We further propose the *Action Strategy Index*, which evaluates the unpredictability of action pattern. We obtain the number of outgoing HMM edges for each HMM node, forming a set that is denoted as $e = \{e_i\} \; \forall \; i \in [1, k']$, where k' is the total number of HMM nodes. Skillful players would have similar values in the e set, while novice players would have very different values. We therefore define the Action Strategy Index as the precision of e, that is, the reciprocal of its standard deviation:

$$ASI = \frac{1}{\sigma(e)} \tag{3.5.5}$$

where σ represents the standard deviation operator. A high ASI value indicate that the player's action patterns are more unpredictable, which indicates a higher skill level.

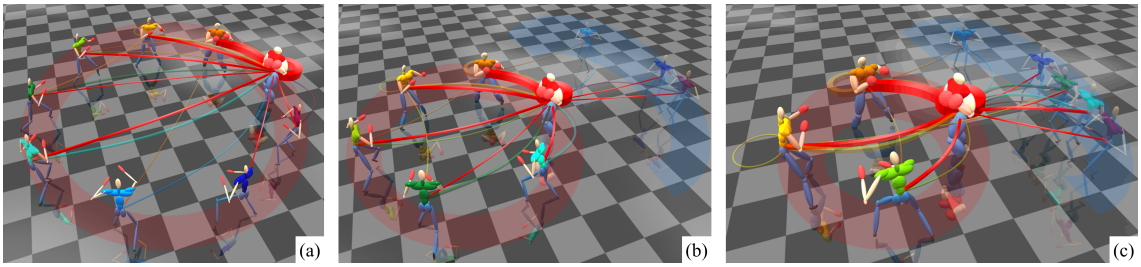


Figure 3.9: *Action-based graphs of the same boxer generated by setting the frequency threshold as (a) 0, (b) 1 and (c) 2. The red shade indicates the inner circle covering nodes of the frequent class, and the blue shade indicate the outer circle covering nodes of the rare class.*

3.5.3 Visualization System

Here, we explain the visualization system for the action-level graph. The system allows easy observation of the preference of action and the boxing pattern. Both are very important aspects to evaluate the high-level strategy of a boxer.

3.5.3.1 Visualizing HMM Nodes

Each action group is represented by its corresponding median action, which is the action that is the closest to the mean value of the action group during k-means clustering. We render the nodes using human characters with the starting posture of the median action. The number of actions in each action group is visualized using the fatness of the corresponding character. The color of the nodes are randomized.

As mentioned in Section 3.4.1.2, we observe that some boxers, especially novices, may produce random actions that are not repeatable. Such actions may generate a large number of thin nodes, which distract the user from evaluating the actions that are often launched. Therefore, we classify the action groups with $|G_i| > a$ into the *frequent class*, and groups with $|G_i| \leq a$ into the *rare class*, where G_i is the number of member actions in a node as defined in Section 3.5.1, a is a preset frequency threshold. Fig. 3.9 shows the result of setting different values of a. We find that setting $a = 2$ generates the best results.

We place the nodes belonging to the frequent class at an inner circle, and those belonging to the rare class at an outer circle, such that the user can identify them

easily and decide what to focus on. For the inner circle, nodes are ordered according to the corresponding value of $|G_i|$, and are placed evenly at a circle with a smaller radius. For the outer circle, to minimize edge crossing, we place the nodes at a position on a circle with a larger radius that is the closest to the nodes with incoming and outgoing edges. To implement this, we develop a simple optimization algorithm that optimizes the position of the nodes. During the optimization, we constrain the position to be at the circle and not overlapping with existing nodes. We then minimize the sum of distance with respect to the nodes connecting to the current one.

By default, we render the HMM node belonging to the frequent class with solid colors, and those belonging to the rare class in semi-transparent colors. This further avoids the user being distracted by the rarely performed actions.

3.5.3.2 Visualizing HMM Edges

We visualize the edges using 2D curves. While we can render the edges with straight lines, the resultant group would be difficult to observe as the lines overlap significantly. We augmented the edges with a small random curvature to solve the problem. We also render the edges as semi-transparent such that the users can see through partially overlapped edges. The thickness of the edge is proportional to T_{ij} calculated in Equation 3.5.4. As a result, a thicker edge connecting node i to node j indicates that the boxer often launches action group j after action group i. The color of the edges are decided based on that of the source node. This helps the user to identify which action groups the boxer may launch after a particular one.

3.5.3.3 Interactive Features

We also implement some interactive features such that the user can select what to view. The most important component of the action-based graph is the action itself. Therefore, we implement an interactive system such that when a user clicks on a particular HMM node, the median action of the corresponding action group is displayed. We also highlight the outgoing edges from such a node. This allows the user to examine individual action group together with the transition probability to

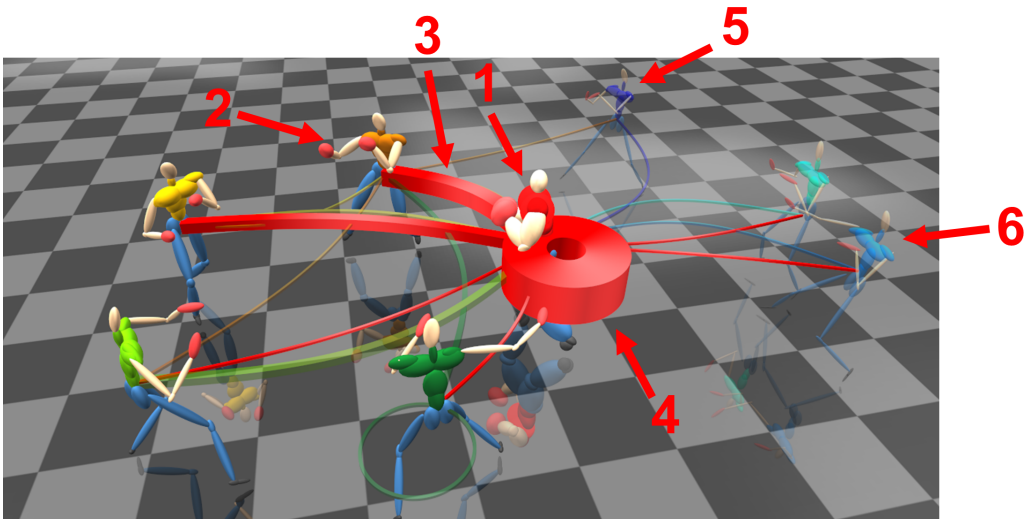


Figure 3.10: *The action-based graph of the Boxer S. 1, 2 are HMM nodes belonging to the frequent class. 3, 4 are outgoing HMM edges from the node 1. 5, 6 are HMM nodes belonging to the rare class.*

the next groups. The information of the node, such as the number of member actions and the number of out-going HMM edges, are displayed on a separate window.

As an example, in Fig. 3.10, there are 5 HMM nodes belonging to the frequent class including node 1 and 2. These nodes are visualized with more muscular characters, meaning that the boxer performs them more frequently. There are 3 HMM nodes belonging to the rare class including node 5 and 6, which are visualized with thinner characters. Node 1 has 5 outgoing HMM edges, in which edge 3 point towards another node, while edge 4 is a self-connecting edge. Edge 4 is thicker than the others, indicating that the boxer performs successive actions belonging to node 1 very frequently. The screen is captured when the user selects node 1, and as a result, all outgoing edges of node 1 are highlighted, and the character representing node 1 performs the corresponding median action.

3.6 Experimental Results

In this section, we present experimental results. We captured the motions of four boxers with varying skill levels. We first give detailed motion analysis and visualization of individual motions, and then compare them side by side using the proposed

indexes. This demonstrates that our system is an effective tool for motion analysis, skill assessment and comparisons. As it is difficult to show the motions in pictures, we refer the readers to the supplementary video for more details.

The four boxers chosen have different skill levels. As a ground truth, their skills were evaluated by a professional boxing coach as skilful, medium, medium and novice respectively, and were denoted as S, M1, M2 and N.

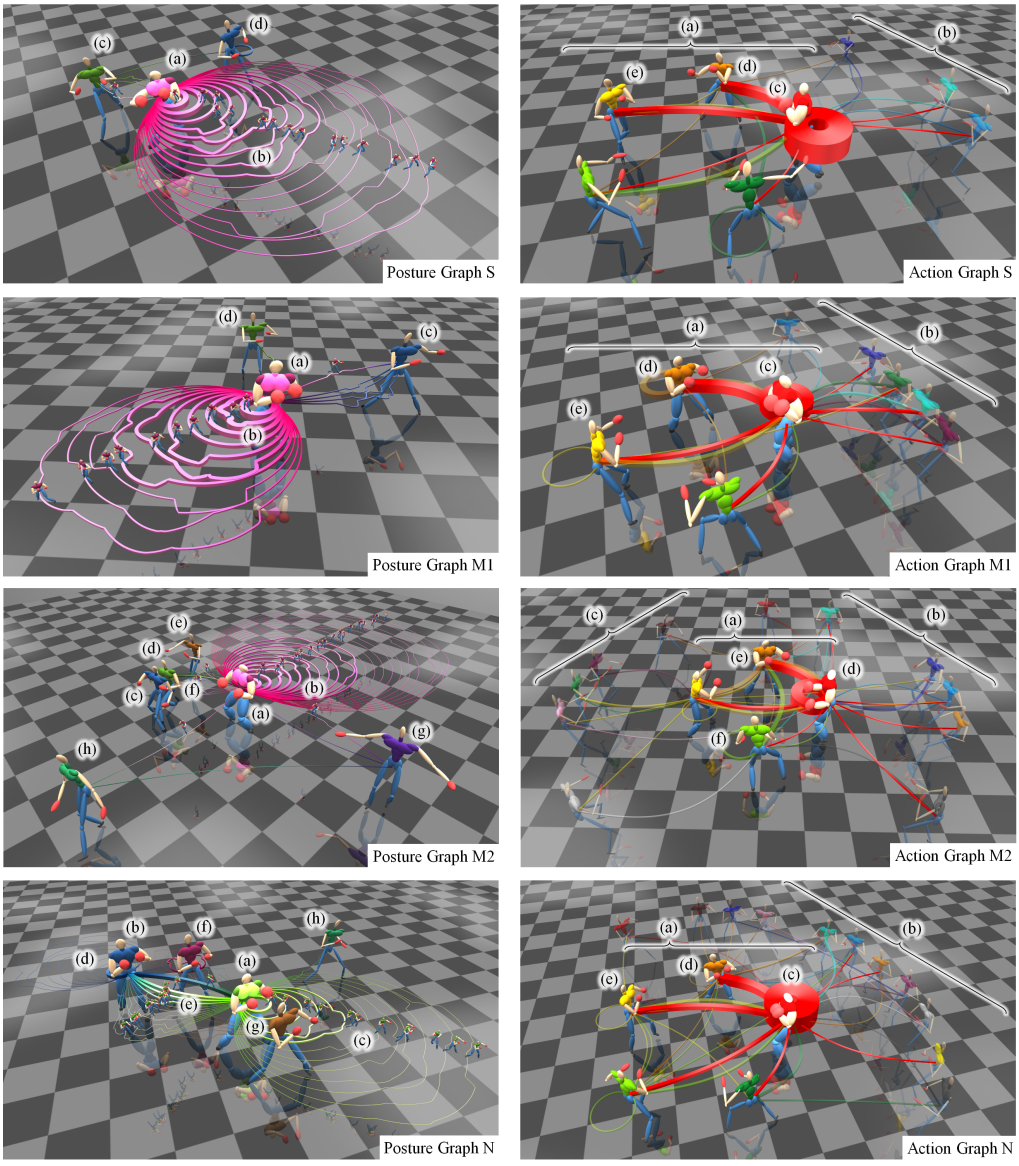


Figure 3.11: *Visualizations on different boxing skill level. (Left) The posture-based graphs and (Right) the action-based graphs for boxer N, M1, M2 and S (top to bottom) respectively.*

3.6.1 Boxer Evaluation

The boxers' posture-based and action-based graphs are shown in Fig. 3.11, in which letter annotations are given to help explain the graphs. These graphs allow users to assess boxing skills even if they are not familiar with boxing.

3.6.2 Boxer S

The first row of images in Fig. 3.11 shows the graphs of boxer S. The posture-based graphs shows a main standard posture (a) to start and end actions, which is good for boxing as it allows the boxer to transit from one action to another effectively through the standard posture. A large variety of actions (b) can be produced from such a posture. There is a second posture in which the arms are further apart (c). This should be avoided as such a posture is weak in blocking attacks. Posture (d) is generated because the boxer trips over during the training. Our system can pick up and visualize such a mistake accurately.

The action-based graph of boxer S shows there are many actions in the frequent group (a) and only a few in the rare group (b). This shows that the boxer is experienced and his actions are consistent. There is a major movement action (c) in the frequent group (a), and such an action has good connections to many of the others. This is good as experienced boxers typically use movement actions to adjust their position relative to their opponent, and launch attacks when the time is right. Other actions in the frequent group (a) are variations of attacks. For example, the more frequently used action (d) is a right-left combo and action (e) is a single right punch, which shows that the boxer tends to start an attack with the right punch. It is good to see that attacking actions may connect to each other, which enhance the unpredictability of the boxer.

3.6.3 Boxer M1

Next, we evaluate the posture-based graph of boxer M1. The boxer has a main standard posture (a) to launch most of the actions (b). However, he has a secondary posture (c) for launching some attacks and another (d) for launching a turning

action. In both postures, the arms are in a low position and cannot guard the boxer well from the opponent. More importantly, the relatively more frequently used secondary posture (c) is performed with the foot distance much wider than the shoulder width. This means the boxer has limited mobility in this posture, as the legs must move towards each other before another stepping action can be performed. These observations show that the boxer is not as experienced and consistent as boxer S.

The corresponding action-based graph shows that there are fewer frequent class actions (a) but more rare class ones (b) compared to boxer S. This means that that the boxing action of boxer M1 is less consistent. The boxer has a large number of movement actions (c) that are connected to all the rest of the action nodes. He also has a variety of attack actions as shown in other actions in the frequent class (a). In particular, action (d) is a left-right combo and action (e) is a left punch, showing that the boxer tends to start an attack with the left punch. Overall, there is an acceptable number of connections among attacks, demonstrating the acceptable unpredictability of the boxer.

3.6.4 Boxer M2

For boxer M2's posture-based graph, there is a main standard posture (a) launching the majority of actions (b). There are, however, a number of secondary postures (b), (c) and (d). These postures are all performed sub-optimally with his arms not guarding the head and should be avoided. Looking closely at the edges (f) going to posture (c), we can find that the posture is performed as a subtle movement to prepare various left punches. This should be avoided as the opponent can tell the moves whenever seeing such a posture. Postures (g) and (h) are very different from the rest and are geometrically far from the other postures. These two postures are performed because the boxer unintentionally raises the arms during the capture. Our system can pick up the mistake and visualize it in the graph.

From boxer M2's action-based graph, it can be observed that there are relatively fewer actions in the frequent class (a), but a large number of actions in the rare class combining (b) and (c). This shows that the boxer is quite inconsistent in the boxing

actions, and could be because of the lack of training and experience. Different from the boxers discussed, boxer M2 has the largest action node (d) of left punch. The second largest action node (e) is a double left punch. The movement action node (f) is relatively small. This shows that boxer M2 has a different boxing style to use left punch as a major action to connect to other actions and his left punch is dominant. Such a boxing style is not advised as a punching action, comparing to a movement one, consume more energy and expose a larger risk of being attacked.

3.6.5 Boxer N

In the posture-based graph of the novice boxer N, there are two major standard postures (a) and (b) instead of one. There are a large number of self-connecting actions (c) and (d) for both postures, as well as a lot of actions (e) connecting the two. This shows that the boxer is highly inconsistent in the boxing postures. Posture (a), the more relatively frequently used one, is inferior to posture (b), due to its wider foot distance. It does not allow the boxer to step freely. Posture (f), (g) and (h) are all secondary postures with different posture variations. They are all not well performed due to the low arm positions limiting blocking capability, and the wide foot width limiting movement capability.

The corresponding action-based graph shows some actions in the frequent class (a) but a large number of actions in the rare class (b). This means that the novice boxer cannot perform actions consistently. The action in the rare class (b) are mainly very long combo that is randomly combined and cannot be reproduced. The main action (c) is a movement action. Such an action cannot connect to a number of others in the rare class (b), and many actions in the rare class (b) are not well connected. This means that the boxer's action is more predictable, which is bad in a match as the opponent can guess what the boxer may launch next. The two more frequently used attack action (d) and (e) are left-right combo and left punch respectively, showing that the boxer tends to start an attack with a left punch. The relationship between boxers' skills and analyzed Connectivity Index results has been explained in Section 3.4.2.

3.6.6 Statistical Analysis

	Boxer S	Boxer M1	Boxer M2	Boxer N
SL	Skillful	Medium	Medium	Novice
PN	138	160	112	176
AN	69	80	56	88

Table 3.1: *Statistics of the boxing motions assessed by human experts. SL: Skill Level evaluated by a professional boxing coach. PN: Posture Number (for starting and ending actions). AN: Action Number.*

	Boxer S	Boxer M1	Boxer M2	Boxer N
FNN	3 (2)	6 (4)	3 (3)	5 (5)
FEN	20 (10)	36 (12)	16 (7)	57 (8)
CI	5.0	3.0	2.3	1.6

Table 3.2: *Statistics of the boxing motions in the Posture Graph. FNN: Fat Node Number (brackets show numbers after removing accidentally created nodes). FEN: Fat Edge Number (brackets show numbers of consistently performed edges). CI: Connectivity Index.*

Here, we give some statistics about the proposed system.

Table 3.1 shows the skill level assessed by a professional boxing coach, as well as the number of postures and actions, for each of the boxers considered. Table 3.2 shows the statistics related to the posture-based graph, including the number of fat nodes and fat edges, as well as the Connectivity Index calculated with Equation 3.4.3. The index evaluates the richness of actions and the flexibility of transitions. It aligns with the boxers' skill level and more skillful boxers have higher Connectivity Indexes. Table 3.3 shows the statistics related to the action-based graph, including the number of HMM nodes (which is further separated into the number for the frequent class and the rare class respectively) and HMM edges, as well as the Action Strategy Index calculated with Equation 3.5.5. It indicates the unpredictability of a boxer, and more skillful boxers are generally more unpredictable. Again, it aligns

	Boxer S	Boxer M1	Boxer M2	Boxer N
NN	7	11	9	16
NNFC	4	3	4	5
NNRC	3	8	5	11
EN	16	27	20	38
ASI	0.572	0.448	0.426	0.378

Table 3.3: *Statistics of the boxing motions in the Aciton Graphs. NN: Node Number. NNFC: Node Number for Frequent Class. NNRC: Node Number for Rare Class. EN: Edge Number. ASI: Action Strategy Index.*

with the boxer' skill level and more skillful boxers have higher Action Strategy Indexes. Although we have only used 4 types of shadow boxing motions in our experiment, we found that the diversity is enough for us to do evaluations and achieve the convincible results in such specific motions [59, 136].

In terms of the computational cost, we run the proposed system on a laptop computer with a Core i7-6820HQ CPU, 16GB of RAM and a NVIDIA Quadro M1000M graphic card. The computational time to analyze the captured motion (Section 3.3.2) and computing the graphs (Section 3.4 and Section 3.5) ranges from 6 to 9 seconds. Averagely, each sequence has around 2700 frames. The variation of computational time is mainly due to the iterative k-means clustering algorithm for both postures and actions, as a larger k requires longer computational time. The run-time cost is low and we achieve frame rate higher than real-time (i.e. 60Hz). The frame rate tends to be lower when there are more characters shown in the graphs.

3.7 Conclusion and Discussions

In this work, we proposed a method to visualize the skills level of boxers using an automatic motion analysis and visualization framework. The proposed posture-based graph is a customized Fat Graph that helps us to analyze the quality of standard postures for launching and finishing actions. The action-based graph is a

customized Hidden Markov Model that visualizes the transition probability among actions. We further introduce the Connectivity Index that is deduced from the posture-based graph and helps evaluation of the richness of actions and the flexibility of transitions, as well as the Action Strategy Index that is deduced from the action-based graph and achieves evaluation of the unpredictability of action patterns. The system is applied to the motion captured from 4 boxers with varying skill levels. The evaluations from our system align with that of a professional boxing coach.

Although we use boxing as our target sport in the experimentation section, the underpinning theoretical development can be applied to most sports that require swiftness, flexibility and creativity, such as tennis, fencing and basketball. The adaptation of the proposed system to these sports and the comparison of the system performances on different sports remain as future work.

We focus on analyzing the skill level of the boxers in terms of motion behaviour such as the richness of the action, the transition of action and the unpredictability of boxing patterns. We do not evaluate the lower-level parameters such as the speed of the punches, which has been explored in previous works. It is an interesting future direction to combine both high-level and low-level evaluation in order to have a full assessment of the boxers.

There are limitations to our method. First, our method is based on the assumption that sports skills mainly consist of a finite number of key postures and key actions. Admittedly, not all sports follow this pattern. Second, the visualization and skill assessment is based on an individual athlete, not considering skills related to collaborations such as those in group sports, in which the assessment might need to employ different criteria.

We argue that novice boxers tend to have different posture-based graphs, while experienced boxers tend to have graphs of a similar topology. This is because unlike experience boxers who have only 1 to 2 main postures nodes, novice boxers tend to have more nodes, resulting in a much larger variation on the graph topology. As a future work, we would like to utilize the system to evaluate a large number of boxers in different skill levels to verify this argument.

Chapter 4

Interaction-based Human Motion Retrieval and Analysis

Modelling and understanding human motion is a central problem for character animation, serious games and human-computer interaction. We observe that the semantic meaning of human movement is usually defined based on the interaction between multiple characters, instead of individual ones. For example, a punching movement that hits is semantically different from one that misses. Unfortunately, there is limited research in analyzing human motion in the sense of interaction, even less capable of comparing interaction of different classes.

Existing research of human motion modelling and retrieval algorithms mostly focus on features from individual characters, such as joint positions and joint angles. Such individual geometric features are limited in modelling the semantic meaning of complex movement involving two or more interacting characters, such as boxing and dancing. They cannot distinguish geometrically similar interactions that have different semantic meaning. For example, a high-five interaction between two characters is similar to a waving interaction if we look at the features of individual characters only. Similarly, they cannot identify the similar semantic meaning from geometrically different interactions, such as a right punch having some level of similarity to a left punch when they both hit the opponent.

Interaction-based features are introduced to solve the problem, but many of them suffer from various limitations. Relative kinematic features such as the joint

relative distance are used to model movement between characters [76]. However, the number of feature increases exponentially with the number of considered joints, and it becomes inefficient to use a high dimensional feature vector to represent the interaction involving multiple characters. A feature selection pre-process can be introduced, but there is a side effect that the optimally selected features depend on the types of interactions. Logical filters are efficient in indexing and modelling the motion of character using multiple manually defined logical rules [69]. However, for two or more characters, there will be an exponential number of possible logical rules, and manually defining the optimal rules requires domain experts' knowledge. The Gauss Linkage Integral (GLI) can represent the degree of rotation between two strings and is applied in analyzing two characters interactions [77]. The problem is that it models the human body as simplified strings, and cannot effectively represent long-distance interactions such as one character avoiding a punch from another. Overall, these interaction-based features either suffer from the problem of exponentially growing dimension size or perform optimally only for some types of interactions.

To solve these problems, the interaction mesh is proposed to robustly model interactions with a limited dimension of features [8]. It considers the set of joints from two or more characters as a point cloud and applies Delaunay Tetrahedralization [79] to connect nearby points, forming a three-dimensional mesh. Previous works have shown great success in using interaction mesh to retarget multi-character interaction [8]. However, the major difficulties are that the topology and dimension of the interaction mesh depend on the postures of the interacting characters, and therefore changes across different classes of interaction and across frames, making it difficult to compute the difference between two interaction meshes. Previous works attempt to solve the problem by dividing the distance function into two parts. For the edges that co-exist in two interaction meshes, a traditional geometry-based distance function is applied. For those that do not co-exist due to the topological difference, [63] assumes zero distance, while [62] simply counts the total number. Since the two parts of the distance function have different natures, forcing them together generates inconsistent results. [137] utilizes an affinity matrix calculated

based on a heuristic to extract the active joint pairs, but the heuristic requires domain knowledge and is likely dependent on the types of interaction.

In this work, we propose a new unified framework to analyze and retrieve human movement from the interaction point of view. We adapt a customized version of interaction mesh as the feature in our implementation as it is robust and has achieved promising results. The main contribution of our framework is a distance function that can compare two topologically different interaction meshes using the Earth Mover's Distance [82], which allows us to evaluate the intrinsic similarity between different classes of interactions robustly. For example, as shown in Fig. 4.1, "punch + being hit" and "kick + being hit" are usually considered to be different classes of interactions, but they are similar as they are both "attack + being hit" interactions. Our system can access the level of similarity between them, allowing us to retrieve interaction with intrinsic correspondence. Experimental results show that our motion retrieval system aligns much better with human perspective of interaction similarity comparing with existing algorithms.

Our system also enables a new way of human motion analysis, which can greatly improve existing motion-based training and monitoring applications. We compare two interactions and evaluate how individual body parts are similar or different from the interaction point of view. In particular, we extract a subset of the interaction mesh between a particular body part and the opponent for each interaction and compare such a sub-mesh using Earth Mover's Distance. Such a system works well for sport training such as boxing and social dancing. In these sports, the players have to understand how their movement differs from the expert ones in terms of how they interact with the opponent, instead of simply mimicking the joint angles of the experts.

4.1 Contributions in This Chapter

We have three major contributions in this work:

- We propose a new framework to evaluate the similarity between interactions by adapting Earth Mover's Distance onto a customized interaction mesh struc-

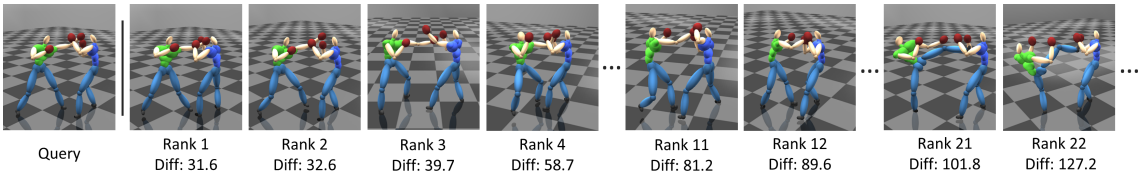


Figure 4.1: *An example of retrieving "left straight punch + being hit" using our system. Our unified framework can compare different classes of interactions and discover their intrinsic similarity.*

ture. This allows us to evaluate the intrinsic similarity between different classes of interactions.

- Utilizing the proposed framework, we implement interaction-based motion retrieval, which is to retrieve similar motions-based on the context of the corresponding interaction. We also implement interaction-based motion analysis, which is to analyze how individual body parts interact with the opponent.

- We produce an interaction database that is open to the research community for benchmarking. This is the first comprehensive database containing different classes of boxing interaction between two characters.

4.2 Overview of Our Method

The overview of our method is shown in Fig. 4.2 and it is composed of two stages: pre-processing and motion evaluation.

In the pre-processing stage, given an interaction motion, we first extract volumetric features between two characters at each frame by applying Delaunay Tetrahedralization process on the joint positions of both characters. Then we apply spatial alignment and keyframe extraction on the original features to encode the motion into a feature vector. This feature vector represents the temporal and spatial information of interaction motion.

In the motion evaluation stages, we calculate the similarity between two motions by evaluating the distance between two corresponding feature vectors. Each element of the feature vector is a volumetric mesh at a certain frame. We use mass transport

solver to evaluate the topology and geometric difference between two volumetric meshes. We adapt Dynamic Time Warping to evaluate two feature vectors with different length.

To evaluate the performance of our system, we build up an interaction motion database and apply leave-one-out approach: each sample in the database is considered as the query and the remaining become the data-set to be evaluated.

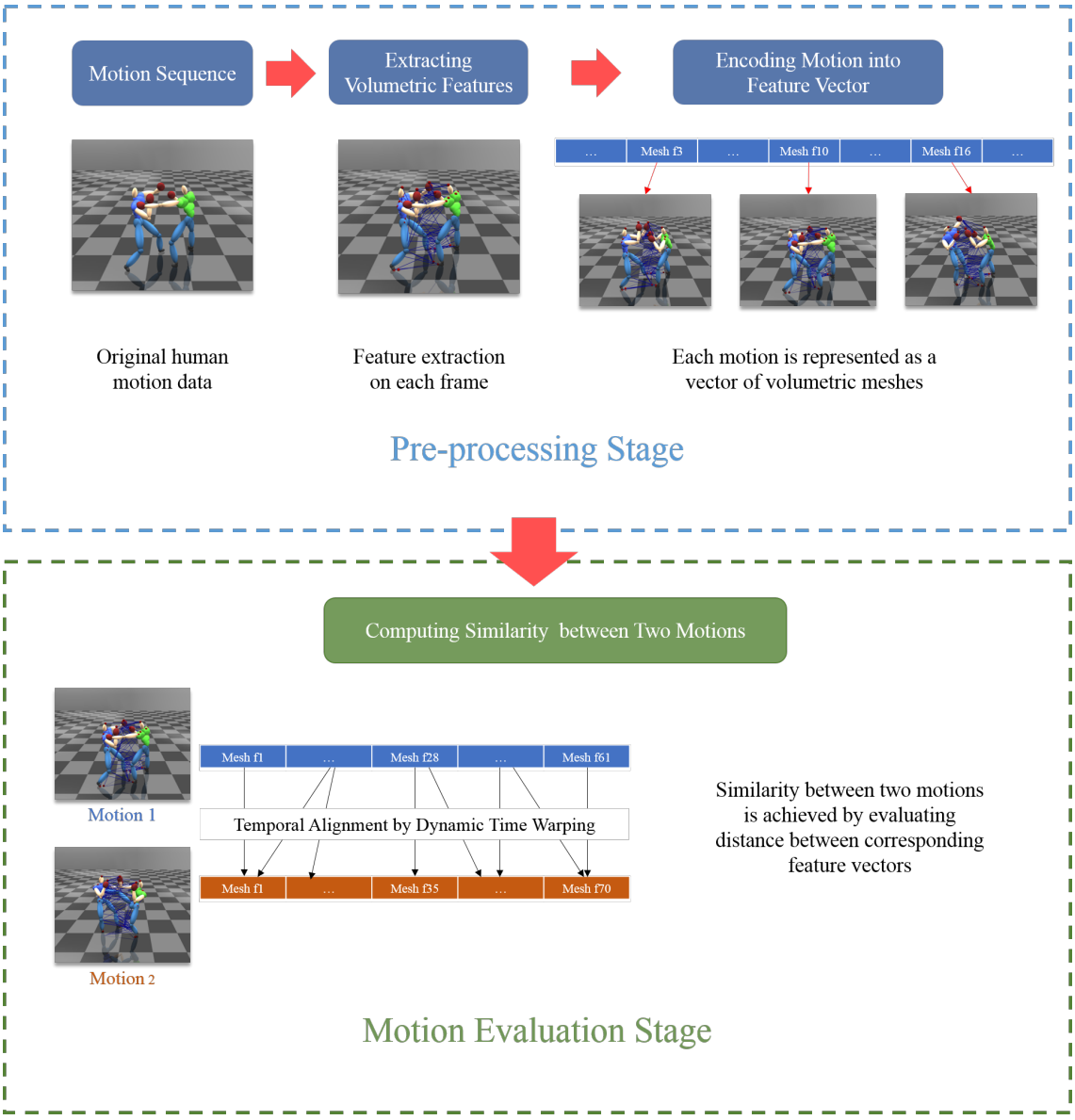


Figure 4.2: *Overview of interaction-based human motion evaluation system.*

4.3 Unified Interaction Comparison

In this section, we explain our unified framework for interaction comparison, which involves three major components. First, we elaborate the approach to represent an interaction sequence using a group of customized interaction meshes. Then, with the help of the Earth Mover's Distance, we evaluate the difference between two interaction meshes. Finally, with such a distance metric, we adapt Dynamic Time Warping to evaluate the difference between two interaction sequences.

4.3.1 Customized Interaction Mesh Structure

Here, we explain how we adapt the interaction mesh structure [8] to represent the interaction between characters. We explain our system using two characters interactions, but the framework can be extended to a scenario of three or more characters.

The interaction mesh structure is used in our system because it can quantitatively represent the implicit spatial relationship between body segments of two characters. Given one frame of an interaction, an interaction mesh is created by generating a volumetric mesh using Delaunay Tetrahedralization [79], considering the 3D Cartesian joint positions of the interacting characters as vertices. An interaction is therefore represented by a series of interaction meshes. The topology and the dimension of the meshes vary over time according to the changing poses of the characters.

We customize the process to generate interaction mesh such that the resultant mesh is more suitable for interaction analysis and retrieval. On top of the set of vertices generated by the joint positions of the characters as in [138], we also include a set of vertices by sampling the skeleton structure of the characters using a predefined length. This allows us to maintain a more uniform density for the mesh, such that motion retrieval and analysis based on the mesh are not biased to specific joints. In our implementation, a character consists of 25 joints, which are shown as the red circles in Fig. 4.3. We uniformly sample body segments using a sampling length of $15cm$. For segments that cannot be evenly divided, the remainder is distributed to vertices in the same body segment. This process creates another 13 vertices, which

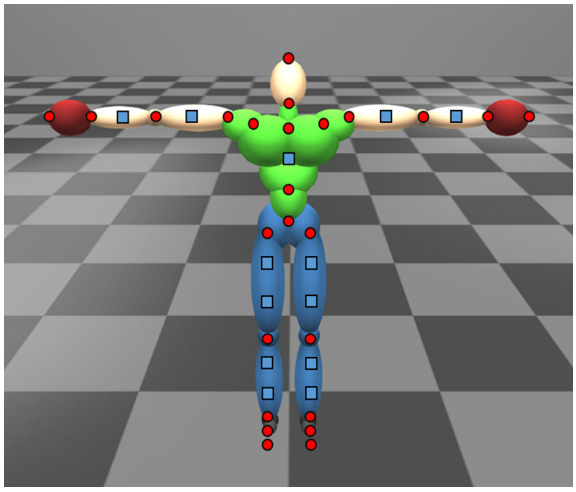


Figure 4.3: *Sampled vertices to create the interaction mesh.*

are shown as the blue squares in Fig. 4.3.

To create the interaction mesh, we consider frame t of an interaction between two characters, and denote $\mathbf{V}^t$ as the set of vertices of the characters:

$$\mathbf{E}^t_{DT} = DT(\mathbf{V}^t) \tag{4.3.1}$$

where DT is the Delaunay tetrahedralization process, and $\mathbf{E}^t_{DT}$ is the set of edges created. Different from [138] that considers all edges, we filter $\mathbf{E}^t_{DT}$ to remove all edges connecting to the same character, as those edges do not correspond to the interaction with the opponent. The resultant set of edges, $\mathbf{E}^t$, is known as interaction mesh of frame t. The blue lines in Fig. 4.4 show the edges before and after filtering.

The temporal sequence of an interaction is therefore represented as a series of interaction meshes, $\mathbf{E} \in \{\mathbf{E}^0, \mathbf{E}^1, \cdots, \mathbf{E}^{t_{total}}\}$, where t_{total} is the total number of frame in the interaction.

4.3.2 Distance between Interaction Meshes

One of the major features of our interaction representation structure is that it can represent semantically dissimilar interactions using the topologically and dimensionally varying interaction meshes, thanks to the use of Delaunay Tetrahedralization in evaluating geometry proximity. This allows us to effectively represent interactions of different semantic meaning (e.g. punching vs. kicking) using a consistent format.

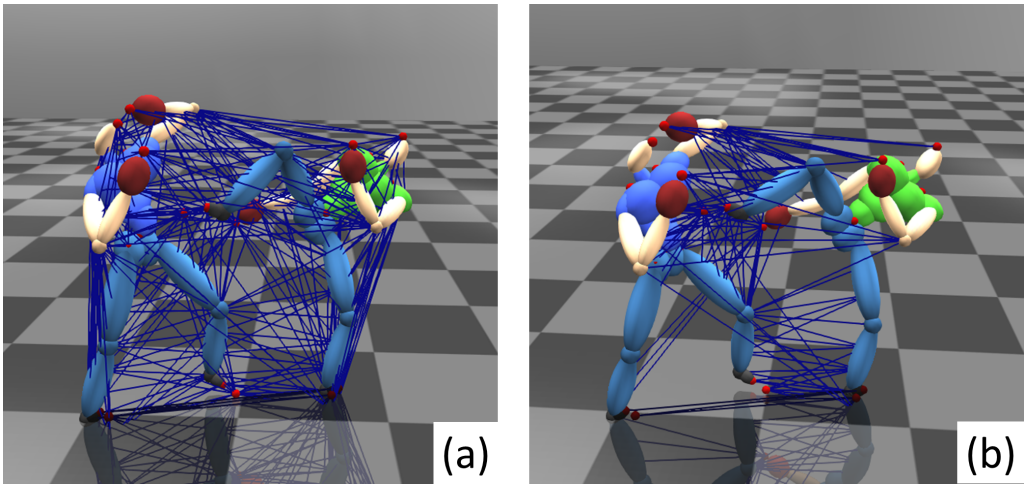


Figure 4.4: *Interaction mesh creation: (a) edges from Delaunay Tetrahedralization (b) edges after filtering.*

Therefore, unlike previous research, our algorithm allows the comparison of two interactions with different semantic meaning, and thereby find out if they have any intrinsic similarity. To achieve this, we propose a distance function that adapts the Earth Mover's Distance (EMD) [82] to find the best correspondence between the input interaction meshes. Such a distance function can effectively compare interaction mesh of different topologies and dimensions.

Here, we explain how to compute the distance between two interaction meshes of two-character interactions. The same distance function is used for human-object interaction, by considering the environment object as the second character.

Given edge e_i from interaction i and edge e_j from interaction j, we represent the difference between the two edges using a customized cosine distance function, which effectively combines the Euclidean distance and orientation distance between the two edges. It is defined as:

$$d(e_i, e_j) = (|e_{i1} - e_{j1}| + |e_{i2} - e_{j2}|) \times \frac{1}{2}(1 - \cos\theta), \tag{4.3.2}$$

where $|*|$ denotes Euclidean distance, e_{i1} and e_{i2} are the two endpoints of e_i connecting characters 1 and 2, e_{j1} and e_{j2} are that of e_j, θ is the angle between the two edges, and $\cos\theta$ is calculated by vector dot product. The idea of the equation is visualized in Fig. 4.5. The cosine term is multiplied by $\frac{1}{2}$ such that it has a range

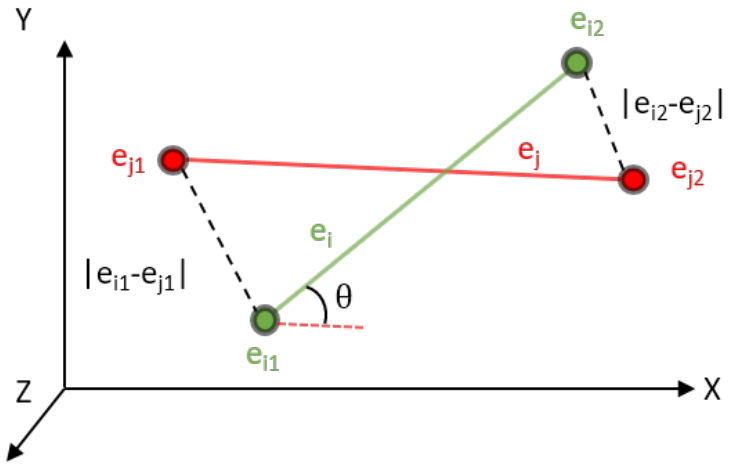


Figure 4.5: *The distance between two edges.*

of $[0.0, 1.0]$. Compared with other designs, such as the weighted sum of distances and cosine angles, ours does not require any parameter tuning. We then adapt a mass transport solver [82] to find the optimal edge-level correspondence between two interaction meshes. The idea is to match the edges by minimizing the overall sum of distance of all the edges. Given two sequences of interaction meshes $\mathbf{E_I}$ and $\mathbf{E_J}$, let us consider one interaction mesh $\mathbf{E}_I^{t_I} \in \mathbf{E}_I$ at frame t_I and one interaction mesh $\mathbf{E}_J^{t_J} \in \mathbf{E}_J$ at frame t_J. The mass transport solver optimizes a set of unidirectional flow to map the edges $e_i \in \mathbf{E}_I^{t_I}$ to $e_j \in \mathbf{E}_J^{t_J}$ with a minimized overall distance:

$$f_{i,j}^* = \arg\min_{f_{i,j}} \left(\sum_{i=1}^{m} \sum_{j=1}^{n} d(e_i, e_j) f_{i,j} \right) \tag{4.3.3}$$

subjected to:

$$\sum_{j=1} f_{i,j} = 1.0 \tag{4.3.4}$$

$$\sum_{i=1} f_{i,j} = \frac{n}{m} \tag{4.3.5}$$

where m and n are the total number of edges in the mesh $\mathbf{E}_I^{t_I}$ and $\mathbf{E}_J^{t_J}$ respectively, $d(e_i, e_j)$ is the distance between two edges calculated with Equation 4.3.2, $f_{i,j}$ is the set of flow values to be optimized. Equation 4.3.4 is a constraint such that in case an edge is mapped into multiple ones, the sum of all outgoing flows is always 1.0.

Equation 4.3.5 is another constraint such that the sum of all incoming flows to an edge is a constant. These two constraints ensure that all edges in $\mathbf{E}_I^{t_I}$ map to all edges in $\mathbf{E}_J^{t_J}$ evenly.

With the optimal set of flow values $f_{i,j}^*$, the minimum distance between two interaction meshes is calculated as:

$$D(\mathbf{E}_I^{t_I}, \mathbf{E}_J^{t_J}) = \sum_{i=1}^{m} \sum_{j=1}^{n} d(e_i, e_j) f_{i,j}^* \tag{4.3.6}$$

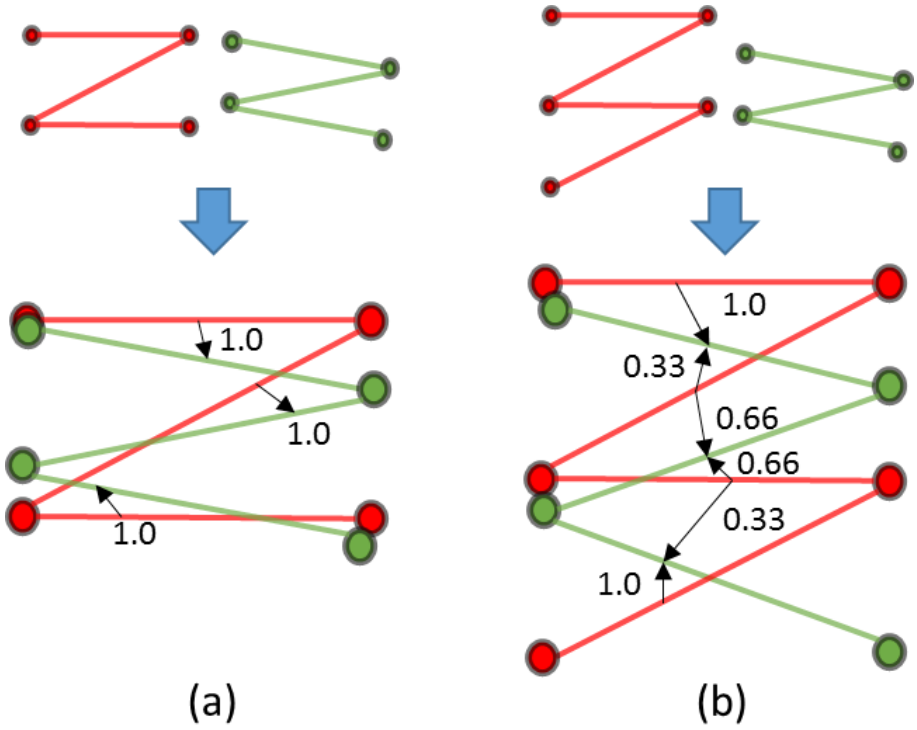


Figure 4.6: *The concept of mass transport solver in 2D.*

Fig. 4.6 visualizes the concept of the mass transport solver in two 2D scenarios, in which the red mesh is matched onto the green one. The flow to match the two meshes is represented by the black arrows, while the corresponding number is the magnitude of the flow. Fig. 4.6a is a simpler case in which both meshes have the same number of edges, and a solution of one-to-one mapping can be achieved. On the other hand, in Fig. 4.6b, since the red mesh has more edges than the green one, some of the edges in the red mesh match partially to those in the green mesh.

Finally, the EMD is calculated as the normalized minimal distance:

$$EMD(\mathbf{E}_I^{t_I}, \mathbf{E}_J^{t_J}) = \frac{D(\mathbf{E}_I^{t_I}, \mathbf{E}_J^{t_J})}{\sum_{i=1}^{m}\sum_{j=1}^{n} f_{i,j}^*} \tag{4.3.7}$$

With Equation 4.3.7, the distance between two meshes, which are usually topologically and dimensionally different, can be calculated.

4.3.3 Distance between Interaction Sequences

Here, we explain how to evaluate the distance between two sequences of interaction. Using Dynamic Time Warping (DTW) [139], we synchronize the temporal variations

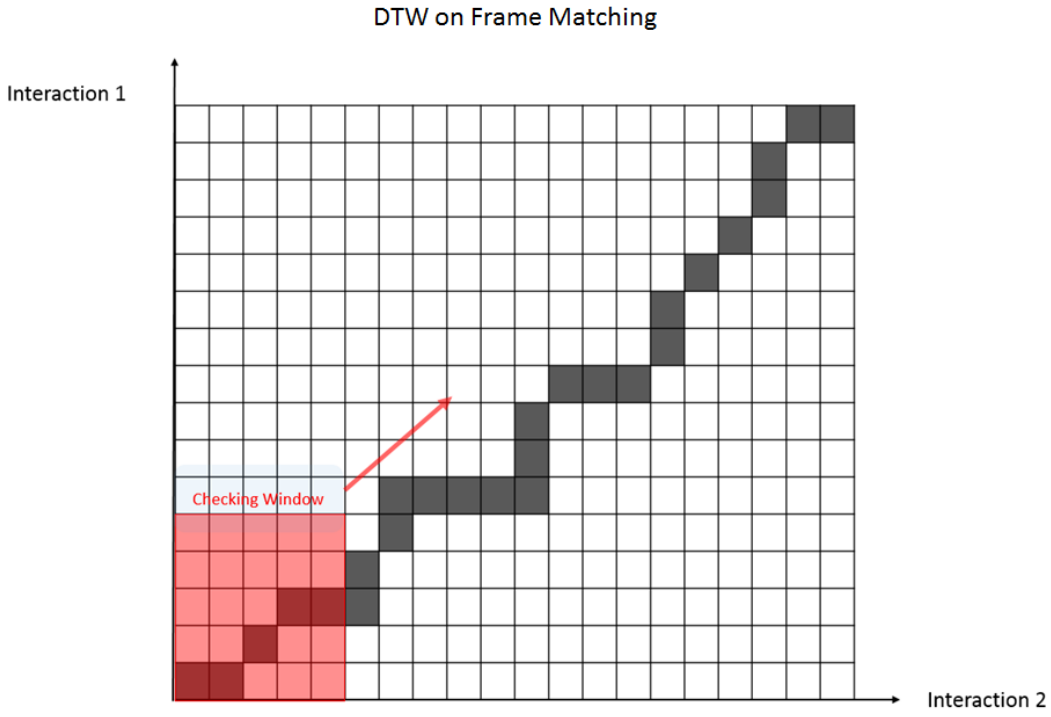


Figure 4.7: *An example of Dynamic Time Warping alignment between two interactions.*

over the time series of two contextually similar interactions. Figure 4.7 shows an example of aligned DTW path among two interactions. Given two sequences of interaction meshes, $\mathbf{E}_I$ and $\mathbf{E}_J$, we define a warping path with W pairs of integer values, $\mathbf{p} = [(p_{I1}, p_{J1}), (p_{I2}, p_{J2}), \cdots, (p_{IW}, p_{JW})]$, to align the two sequences. Using such a path, for each $w \in [0, W]$, the interaction mesh $\mathbf{E}_I^{p_{Iw}} \in \mathbf{E}_I$ is aligned with

$\mathbf{E}_J^{p_{Jw}} \in \mathbf{E}_J$. Therefore, the cost of the path is defined as:

$$c_p(\mathbf{E}_I, \mathbf{E}_J) = \sum_{w=1}^{W} EMD(\mathbf{E}_I^{p_{Iw}}, \mathbf{E}_J^{p_{Jw}}) \tag{4.3.8}$$

The DTW distance is defined as the distance using the optimal warping path $p*$ with minimal average cost:

$$DTW(\mathbf{E}_I, \mathbf{E}_J) = c_{p*}(\mathbf{E}_I, \mathbf{E}_J) \tag{4.3.9}$$

and the interaction mesh distance is defined as the average DTW distance:

$$\overline{DTW}(\mathbf{E}_I, \mathbf{E}_J) = \frac{1}{W} DTW(\mathbf{E}_I, \mathbf{E}_J) \tag{4.3.10}$$

We normalize interactions spatially to compare them with local coordinates, thereby eliminating the influence from different world coordinates. In general, there are two strategies to do so. The first strategy assumes that the interacting characters have unique identities. We therefore consistently use the same character in different interactions as a reference to normalize the whole time series of interactions, by removing its pelvis translation and its horizontal facing angle in the first frame. The second strategy assumes that the two characters are anonymous. We can then obtain two normalized results by considering either of them as the reference. In this case, when comparing interactions, we evaluate both normalized results and select the one that generates the smaller difference. We opt for the first strategy since characters in movies and games usually have unique identities. For example, "a hero kicking a monster" is considered to be different from "a monster kicking a hero".

In our implementation, we also speed up the DTW calculating by extracting key frames from the original motion. Here we adept the similar idea which has been represented in [140].

In particular, given an interaction motion, we first calculate a self similarity matrix (SSM) as shown in Fig. 4.8(b) which represents the distance between all frame-pairs during the whole motion. We make use of our interaction mesh representations and the distance between two frames are evaluated by Equation 4.3.7. This allow us to extract the key frames based on the change of spatial relationship between two characters.

After we achieve the *SSM* of the motion, we iteratively cluster the temporal adjacent frames, which are below the distance threshold, into the same group. Once the iteration is finished, the original motion are divided into several motion segments. Finally, we extract the central frames of all segments and consider them as the key frames of the motion.

According to different choices of distance threshold, different number of key frames are achieved. Based on our experimental results, we found that 9 key frames can speed up the calculation without introducing a significant error. Fig. 4.9 indicates the reconstruction errors between original motions and different number of key frames of that motions.

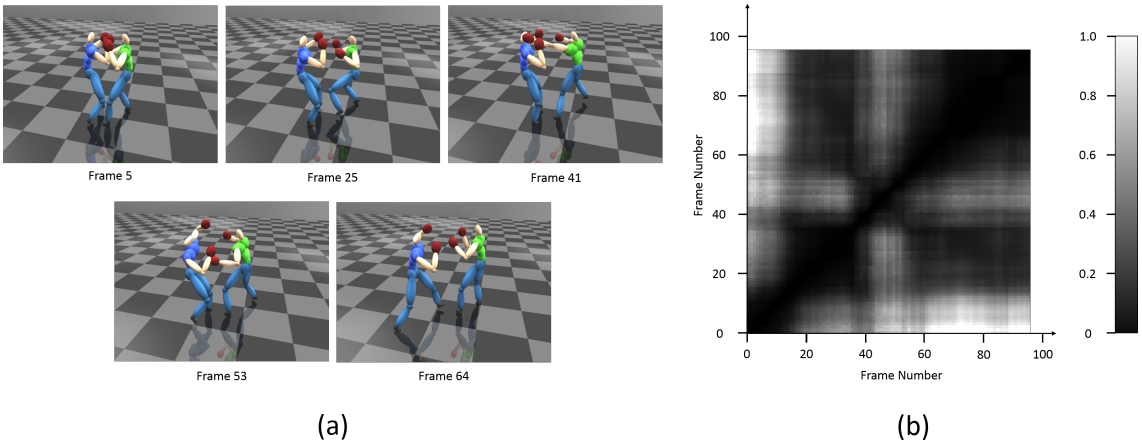


Figure 4.8: *An example of (a) the "left punch + being hit" motion and (b) the self similarity matrix of this motion. The higher similarity is indicated by lower brightness.*

4.4 Interaction-based Retrieval

In this section, we explain how we construct an interaction database, index and retrieve similar interactions using our algorithm proposed in Section 4.3. Interaction-based motion retrieval has the advantage of obtaining results that share similar high-level semantic meaning, which is useful for animation and gaming systems that involve heavy character interactions.

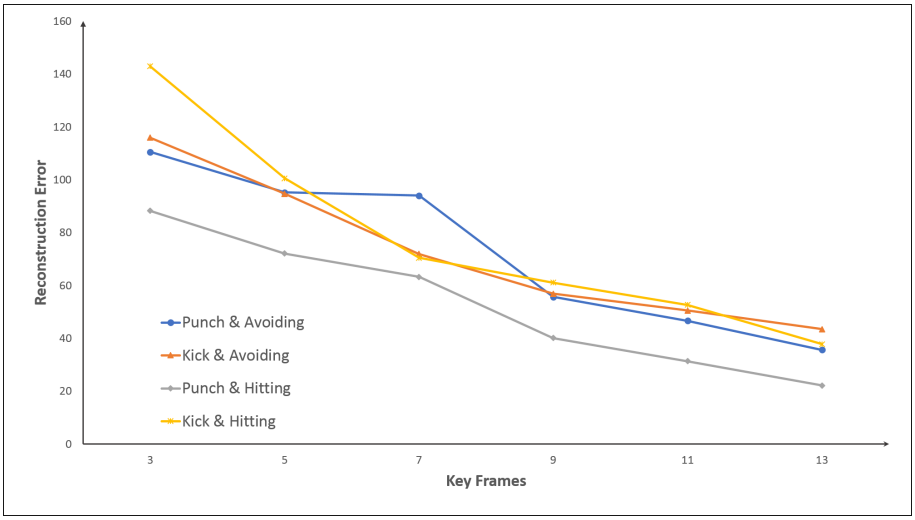


Figure 4.9: *Reconstruction errors at different number of key frames on: " Punch & Avoiding", "Kick & Avoiding", "Punch & Hitting" and "Kick & Hitting".*

4.4.1 Interaction Database Construction

We construct a comprehensive interaction database of kick-boxing between two characters. Kick-boxing was chosen as it involves a large variation of movement and is considered to be one of the most challenging domains of human motion research [40]. Since capturing two people boxing is costly and time-consuming due to the limitation of motion capture hardware, we adapt the interaction synthesis framework proposed in [40] to synthesize high-quality interaction. The major advantages of such an approach are that we can guarantee the availability of data for all classes of interactions, and we can categorize the data with synthesizing parameters, such as attacker's punching style and opponent's defending strategies.

We synthesize interactions as follows. First, we capture the shadow boxing of a single boxer and construct an *action level motion graph* [141]. In such a graph, actions are annotated, and those with similar starting and ending frames are connected. Second, we define a set of *semantic interaction classes*, in which each class defines the interaction pattern [39] to be performed by the characters. Third, we perform *temporal tree expansion* to synthesize interaction between two characters using a set of reward functions [40], and extract the interactions that fit into our pre-defined list of interaction classes. Finally, for each interaction, with the motions

Overall Interaction	Active Character's Motion	Passive Character's Motion
(1) A Attacks B	(1.1) A Single Punch	(1.1a) B Not Being Hit (1.1b) B Being Hit
	(1.2) A Multiple Punches	(1.2a) B Not Being Hit (1.2b) B Being Hit
	(1.3) A Single Punch + Single Kick	(1.3a) B Not Being Hit (1.3b) B Being Hit
	(1.4) A Multiple Punch + Single Kick	(1.4a) B Not Being Hit (1.4b) B Being Hit
	(1.5) A Single Kick	(1.5a) B Not Being Hit (1.5b) B Being Hit
	(1.6) A Single Kick + Single Punch	(1.6a) B Not Being Hit (1.6b) B Being Hit
(2) B Attacks A	(2.1) B Single Punch	(2.1a) A Not Being Hit (2.1b) A Being Hit
	(2.2) B Multiple Punches	(2.2a) A Not Being Hit (2.2b) A Being Hit
	(2.3) B Single Punch + Single Kick	(2.3a) A Not Being Hit (2.3b) A Being Hit
	(2.4) B Multiple Punch + Single Kick	(2.4a) A Not Being Hit (2.4b) A Being Hit
	(2.5) B Single Kick	(2.5a) A Not Being Hit (2.5b) A Being Hit
	(2.6) B Single Kick + Single Punch	(2.6a) A Not Being Hit (2.6b) A Being Hit
(3) Both A & B Attack	(3.1) A Multiple Punches & B Multiple Punches	
	(3.2) A Multiple Punches + Single Kick & B Multiple Punches + Single Kick	

Table 4.1: *The semantic interaction classes. "A" and "B" represent the interacting characters.*

of the two characters, we generate the corresponding interaction mesh as explained in Section 4.3.1.

The complete list of interaction classes used to synthesize interaction is shown in Table 4.1. Designing such a list requires domain knowledge, and is more of an art than a science. Our strategy is to enumerate different combinations of boxing interaction by first deciding which (or both) of the characters attacks, as shown in the left column of the table. Then, we list the commonly seen attacking patterns for the attacking character, which refer to the middle column of the table. Notice that since it is relatively uncommon for a real boxer to kick continuously, we do not include such a class. For the passive character, the two common motions are being hit and not being hit. Class 3 in the table refers to the case in which both characters attack and none of them gets hit. By enumerating the combinations, we obtain 26 interaction classes. With these classes, we synthesize 315 interactions.

4.4.2 Interaction Retrieval

Here, we explain our interaction-based retrieval system and our interactive retrieval system with user-given constraints.

We implement an interaction-based retrieval system. Given one interaction, we apply Equation 4.3.10 to evaluate its difference with respect to all motion in the database, and retrieve the most similar ones. Fig. 4.1 shows the retrieved results of using a left straight punch and hit interaction as the query (i.e. class 1.1a in Table 4.1), annotated with the corresponding ranks and differences. The advantage of our system is that /colorredit compares different types of interactions and discover their intrinsic semantic similarity. This allows the system to retrieve results that align with human perception of interaction.

We also implement an interactive retrieval system based on user-provided constraints. In particular, a distance constraint and an object collision constraint are designed. Our system first precomputes the distance between all pairs of interactions in the database using Equation 4.3.10. During run-time, the user provides a query interaction with constraints. Our system then retrieves the most similar interaction that satisfy the constraints in real-time. These constraints demonstrate the potential of applying our system in interactive animation production, in which the required interaction that fits with the environment and storyboard can be found automatically. More complex constraints can be designed according to the user's need.

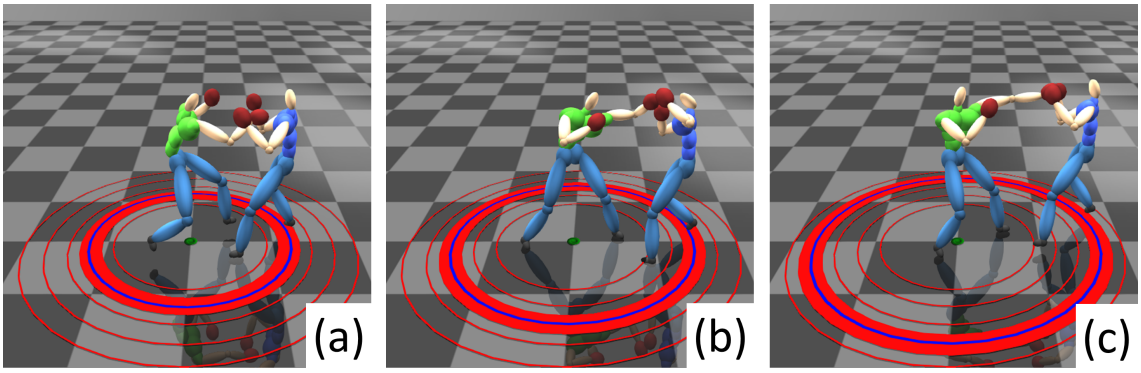


Figure 4.10: *Interactive retrieval by adjusting the distance between two characters.*

We have designed the distance constraint as follow:

$$d_{min} < \left|\mathbf{V}^0_{hips\_A} - \mathbf{V}^0_{hips\_B}\right| < d_{max} \tag{4.4.11}$$

where d_{min} and d_{max} are the lower bound and the upper bound distance given by the user, $\mathbf{V}^0_{hips\_A}$ and $\mathbf{V}^0_{hips\_B}$ are the 3D hips positions of the two interacting characters at frame 0 respectively. This constraint therefore enforces the distance between the characters during the first frame of an interaction. Fig. 4.10 shows an example of applying the distant constraint, in which the inner and outer radii of the red torus represent d_{min} and d_{max} respectively, and the blue circle represents the initial distance between the two characters. Fig. 4.10a shows the initial interaction. When the preferred distance between the characters increases in Figs. 4.10b-c, similar interaction that fits the constraints are retrieved.

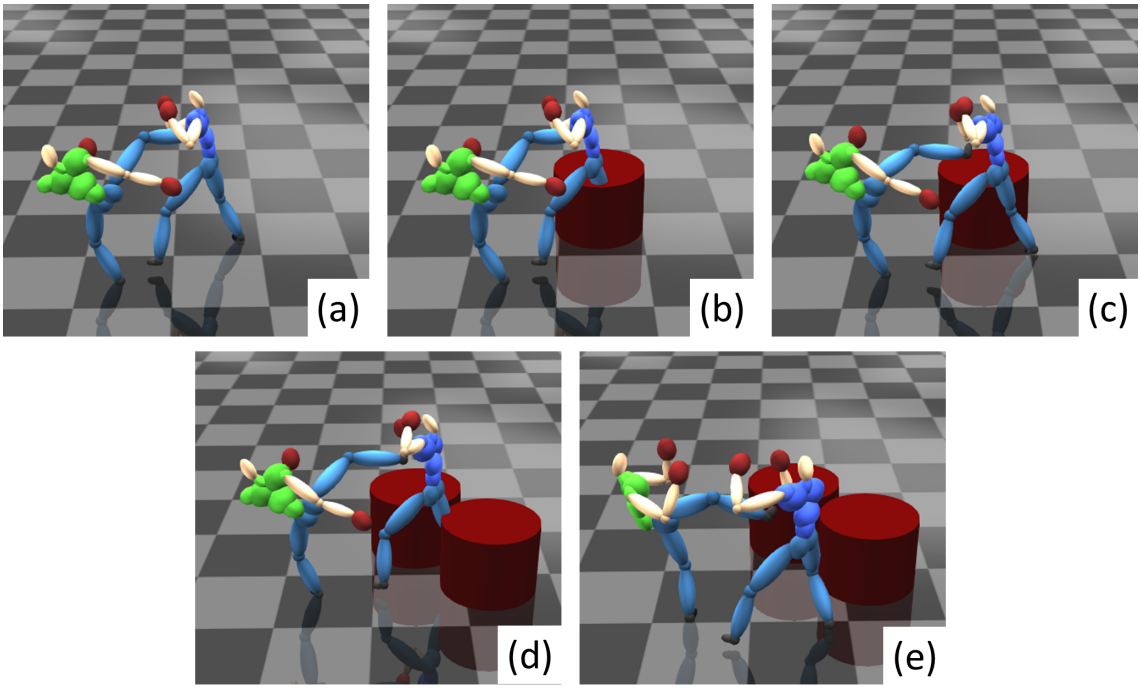


Figure 4.11: *Interactive retrieval by introducing objects.*

We have also designed the object collision constraint as follow:

$$\left|\mathbf{V}^t_j - \mathbf{V}_{obj}\right| > d_{obj} \quad \forall t, j \tag{4.4.12}$$

where $\mathbf{V}_{obj}$ is the 3D position of an object, d_{obj} is the distance to avoid colliding with it, $\mathbf{V}^t_j$ represents the position of joint j at frame t. We consider all the joints for

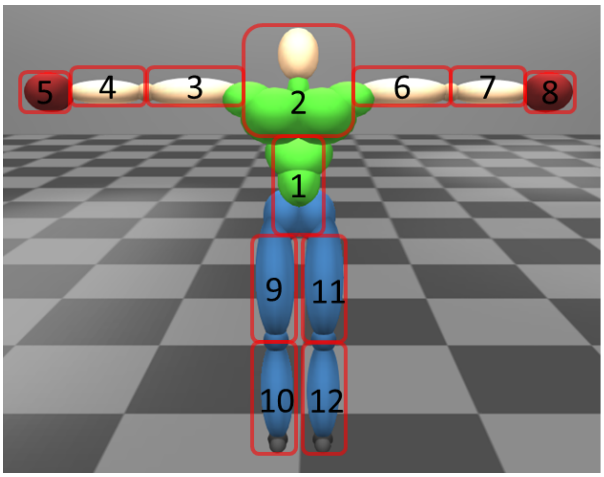
both characters in all the frames for this constraint. This ensures that the characters do not collide with the objects during the whole interaction. We can apply the constraint to multiple objects as well. Fig. 4.11 shows an example of applying the collision constraint. Fig. 4.11a is the initial interaction. In Figs. 4.11b-e, the user introduces objects that lead to collision. The system then retrieves the most similar interaction that satisfies the object constraint.

4.5 Interaction-based Motion Analysis

In this section, we explain how we apply our algorithm proposed in Section 4.3 to analyze motions. In particular, we would like to compare two interactions and evaluate how individual body parts are similar or different in terms of interacting with the opponent. This technique can enhance sport training such as boxing, dancing and fencing, in which novice players are usually required to mimic how experienced players interact with the opponent.

4.5.1 Interaction Sub-mesh Analysis

Here, we compare individual body parts in two interactions in terms of how the parts interact with the opponent. This involves segmenting the character into multiple body parts, generating interaction sub-mesh, and evaluating difference.



Index		Body Part
1		Torso
2		Chest & Head
3	6	Left/Right Upper Arm
4	7	Left/Right Lower Arm
5	8	Left/Right Hand
9	11	Left/Right Upper Leg
10	12	Left/Right Lower Leg

Figure 4.12: *Body parts definition for motion analysis.*

While it is possible to analyze the interaction based on individual vertices as shown in Fig. 4.3, such a representation is not trivial to human understanding due to the small size of a vertex. Therefore, we segment the character into 12 body parts with easily understandable part names, as shown in Fig. 4.12. Each part consists of a subset of vertices defined in Fig. 4.3. Some vertices belong to multiple body parts depending on the body hierarchy, such as the elbow belonging to both upper and lower arms.

Considering one body part of one character, we extract a corresponding sub-mesh from the interaction mesh, which represents how the part contributes to the interaction with the opponent. In particular, given an interaction mesh, we extract the set of edges connecting the vertices of the considered body part to any vertices of the opponent. Notice that only the vertices with spatial proximity are connected in the interaction mesh. As we only consider a subset of vertices from the two characters, we name the generated mesh as interaction sub-mesh.

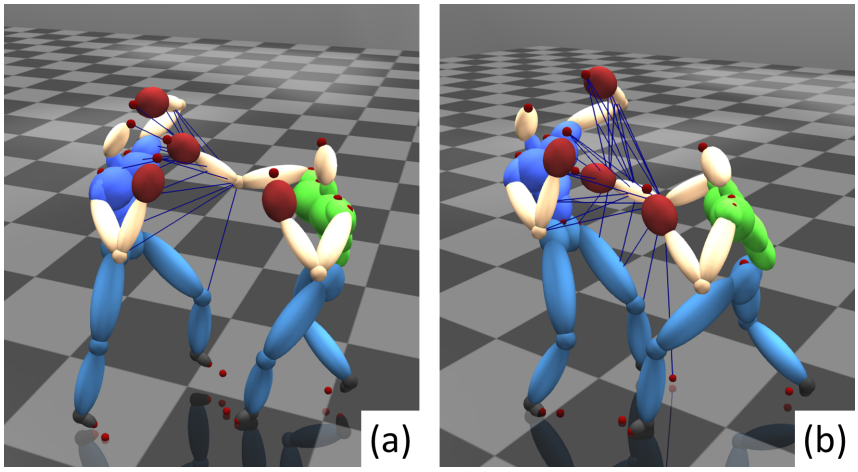


Figure 4.13: *The sub-meshes of two interactions extracted by the right lower arm of the green character.*

Fig. 4.13 shows the interaction sub-mesh generated for the right lower arm of the green character in two interactions. While both interactions are "right straight punch + being hit", the hitting target in Fig. 4.13a is the head, while that in Fig. 4.13b is the torso. As a result, the sub-mesh generated for the former mainly concerns the upper body of the opponent, while that for the latter concerns the middle part of the opponent. By considering the interaction sub-mesh, we can identify

such a semantic difference. However, using traditional methods that evaluate each character independently, it is difficult to achieve similar results.

Given the extracted interaction sub-meshes for the considered body part in all frames of the two interactions, we first apply Equation 4.3.9 to find out the optimal temporal alignment $\mathbf{p}*$ that consists of W warping pairs, i.e. $(p_{Iw}, p_{Jw})\forall w \in [1, W]]$. Then, instead of finding the average distance across all frames, we use such an alignment to calculate a temporal series of frame-based EMD to understand the temporal aspect of the distance:

$$EMD(\mathbf{E}_{Ib}^{p_{Iw}}, \mathbf{E}_{Jb}^{p_{Jw}}) \quad \forall w \in [1, W] \tag{4.5.13}$$

in which $\mathbf{E}_{Ib}^{p_{Iw}}$ and $\mathbf{E}_{Jb}^{p_{Jw}}$ are the interaction sub-mesh created by body part b for interactions I and J at frames pIw and pJw respectively. Such a sub-mesh distance represents how different the considered body part interacts with the opponent in the two interactions.

4.5.2 Interaction Difference Visualization

Here, we explain our system to visualize the difference for the body parts between two interactions.

The inputs of our visualization system are a reference interaction and an analyzing interaction. In typical application such as sport training, the former can be an interaction performed by experts, while the latter can be that performed by novices. The target is to evaluate how the analyzing interaction differs from the reference one.

We first evaluate the interaction sub-mesh distances between the two interactions for all body parts in all frames according to Equation 4.5.13. We then normalize the distance values into the range of $[-1, 1]$ as:

$$\widehat{EMD}(\mathbf{E}_{Ib}^{p_{Iw}}, \mathbf{E}_{Jb}^{p_{Jw}}) = \frac{EMD(\mathbf{E}_{Ib}^{p_{Iw}}, \mathbf{E}_{Jb}^{p_{Jw}}) - \mu}{3\sigma} \quad \forall w, b \tag{4.5.14}$$

where μ and σ are the computed mean and standard derivation of all distance values obtained. We use 3σ as the denominator as it covers 99.7% of the data under a normal distribution. Values beyond 3σ are capped. We finally map the normalized

distance values onto a color scale from green to red, such that green color represents similar body parts, and red color represents difference ones. To enhance visual quality, we apply a Gaussian filter for filtering the color of each body part.

Fig. 4.14 shows an example of our visualization system, in which the reference interaction is a punch blocked by the opponent, while the analyzing interaction is a punch hitting the opponent. The visualization shows that the punch of the attacker, as well as the head and body of the defender, are the body parts that contributes to the difference of the interaction the most. Such a different is important in defining the semantic meaning of the interaction. It is not easy to detect using traditional distance metric such as joint angle distance.

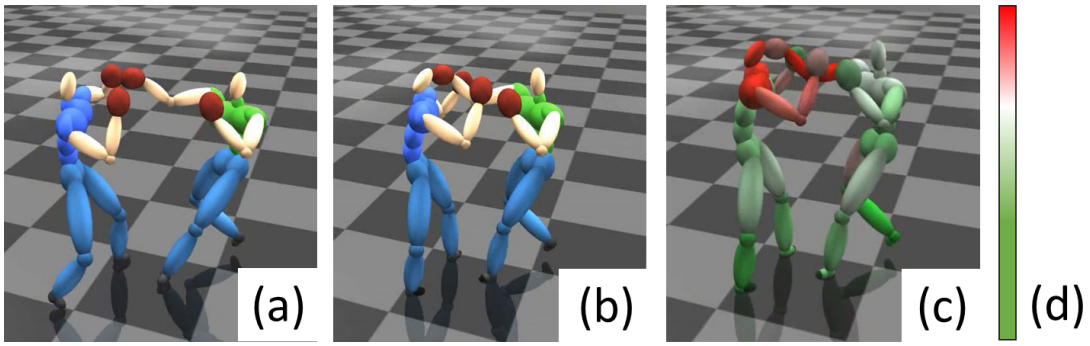


Figure 4.14: *Visualizations of different interactions: (a) Reference interaction (b) Analyzing interaction (c) Interaction difference visualization (d) Color scale used.*

4.6 Experimental Results

In this section, we evaluate the performance of our interaction-based motion retrieval and analysis system. We compare our method with an interaction-based feature known as *space-time proximity graphs* [62], as well as traditional human-centered features including joint positions [66] and joint angles [65].

The experiments were performed on a computer with dual Intel Xeon E5-2687W CPUs, a NVIDIA Quadro K4000 display card and 64GB RAM. Using our proposed method, computing the distance between two interactions took 0.17 second on average. Precomputing the distances of all combinations of interactions in our database with 314 interactions took around 3 hours. A single interactive retrieval with user

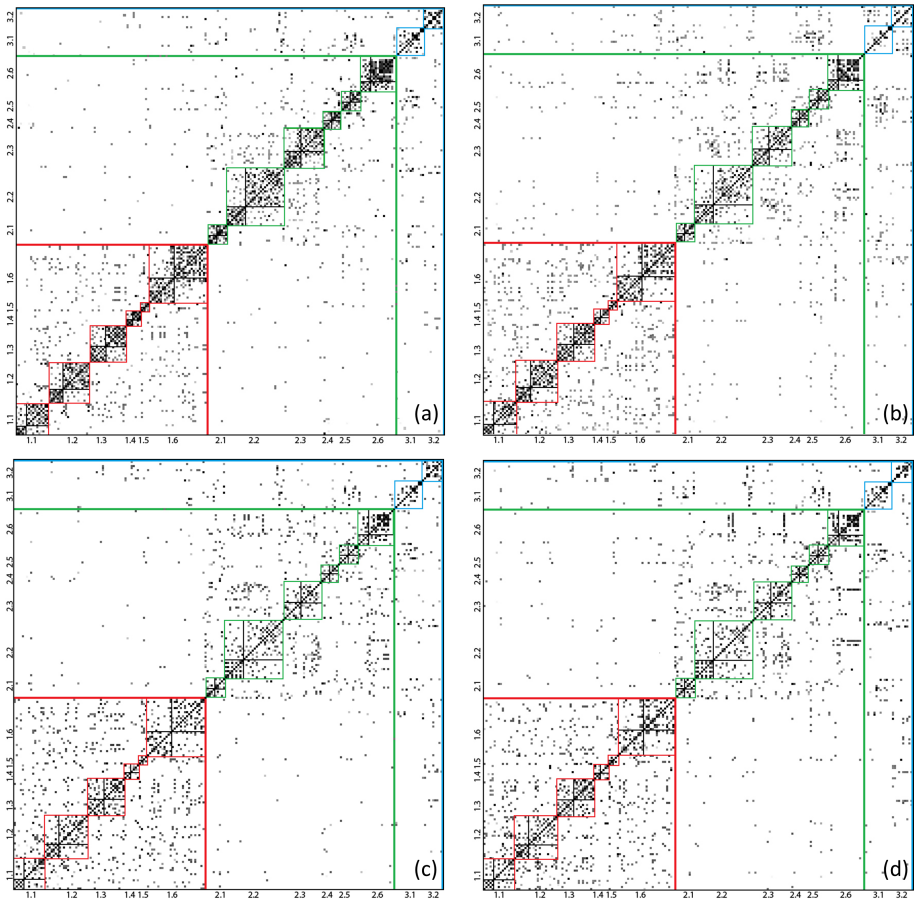


Figure 4.15: *Similarity matrices evaluated by (a) our method, (b) space-time proximity graphs [62], (c) joint position distance [66], (d) joint angle distance [65].*

constraints took 0.02 second on average. Computing the body part difference visualization between two interactions took 0.26 second on average.

Our interaction database is open for public usage. Please find it in our project website.

4.6.1 Retrieval Performance Analysis

Here, we access the performance of our interaction-based retrieval by similarity matrices, as well as the precision and recall graphs.

Fig. 4.15 shows the similarity matrices our method comparing with the others, in which each pixel represent the similarity between two interactions. Pixel color is calculated by the normalized distance between the two interaction, and the value 3σ (i.e. standard derivation) of each method is used as the normalizer of the respective

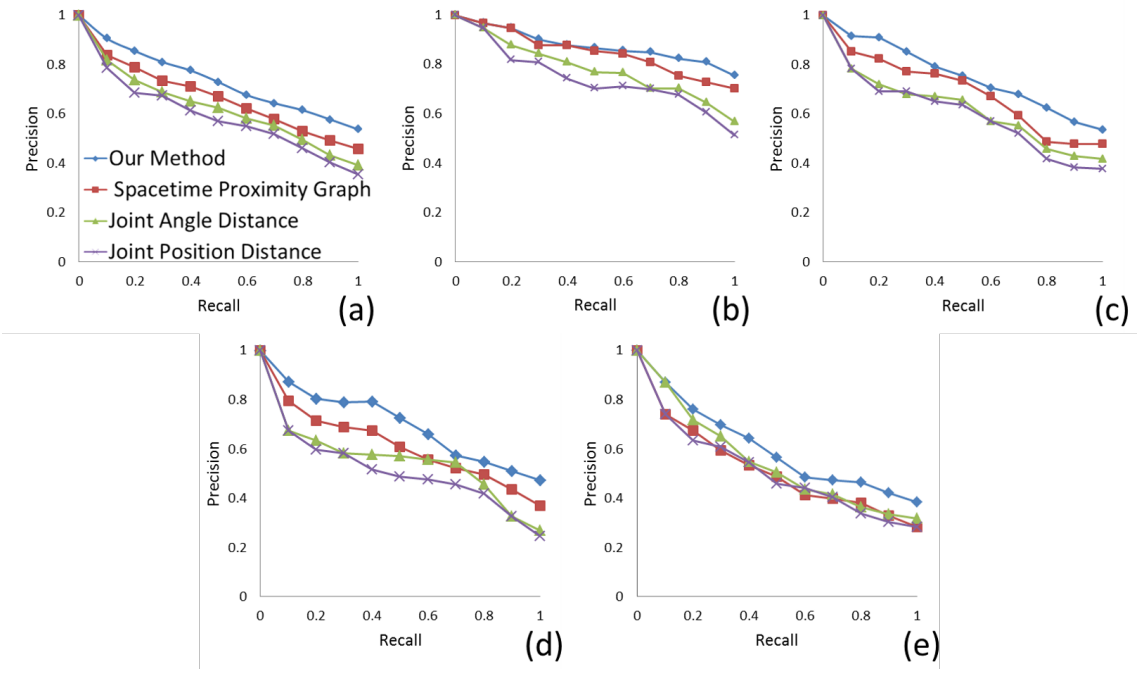


Figure 4.16: *Precision and recall graph for (a) the overall system, (b-e) the complexity level 0-3 respectively.*

matrix. Darker pixels represent higher similarity. We arrange the motion according to semantic interaction classes defined in Table 4.1, and mark the X and Y axes using the class labels. We also highlight the square areas in the matrix that belongs to the same semantic class with red (class 1), green (class 2) and blue (class 3). Within each square in class 1 and 2, we further highlight the sub-classes a and b with black squares.

From the Figure, we can observe that our method has high intra-class similarity and high inter-class difference.In our method, the retrieval results are more aligned with diagnosing and more concentrated in each box (class). In particular, comparing to [62], our method performs better in more complex interactions, such as classes 3.1 and 3.2 in which both characters attacks, as well as classes 1.2, 1.4, 2.2, 2.4 in which the attacking patterns are more complex. This is mainly because the distance function in [62] involves a topology distance term, which performs inconsistently during complex movement. Human-centered features [65, 66] cannot classify semantic meaning well, resulting in a high inter-class similarity. In particular, they fail to accurately classify sub-classes that indicate if a character is hit or not.

Complexity Level	Included Semantic Classes
Level 0	1.1, 1.5, 2.1, 2.5
Level 1	1.3, 1.6, 2.3, 2.6
Level 2	1.2, 1.4, 2.2, 2.4
Level 3	3.1, 3.2

Table 4.2: *Grouping of semantic interaction classes into different levels of complexity.*

We also compare the methods using precision and recall as shown in Fig. 4.16. Given a query interaction, only retrieved results within the same semantic subclass as defined in Table 4.1 (e.g. class 1.1a) are considered as relevant results. As shown in Fig. 4.16a, our method consistently outperforms the others. To evaluate how our system performs in interaction of different levels of complexity, we group the interaction classes as shown in Table 4.2. Fig. 4.16b-e shows that our system outperforms the others in all complexity groups. In particular, comparing with [62], it performs better in more complex interactions.

4.6.2 Alignment with Human Perceived Similarity

Here, we access how our retrieval system aligns with human perception of interaction similarity.

We conducted a survey of 35 participants (age between 18 and 36) from different backgrounds. We created a smaller interaction database by randomly selecting 20 interactions from the full database. It generated 210 distinct pairs of interactions ($^{20}C_2 + 20$) for similarity evaluation. Each participant conducted a questionnaire in which 30 pairs of interactions were drawn in random. The participant graded the similarity of each pair in a 9 point scale, which 1 representing "totally different" and 9 representing "exactly the same". We then calculated an average similarity value for each pair of interactions, which was used as the ground truth.

We compare the normalized distance obtained from different methods with the ground truth. Fig. 4.17 shows the root mean square error, in which the axes

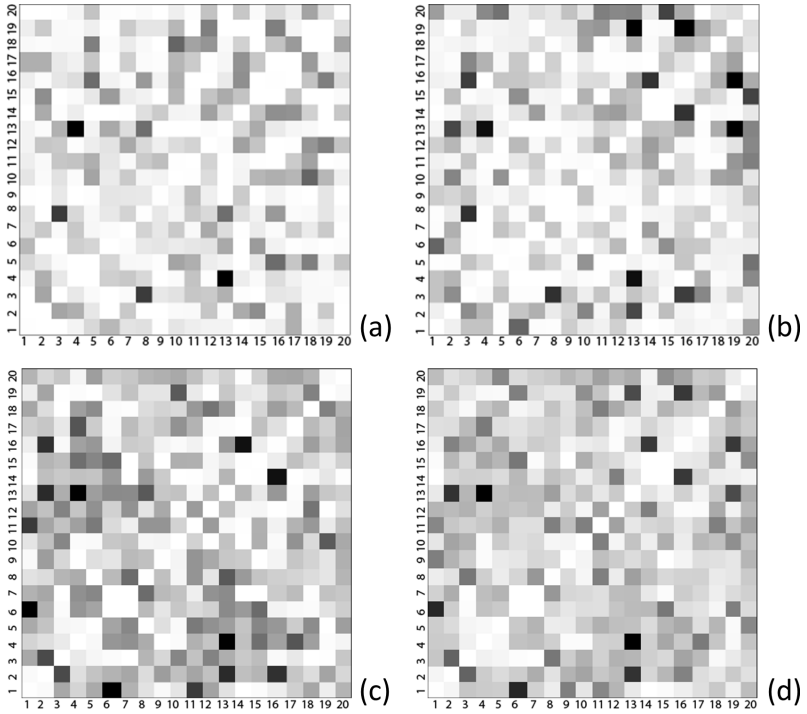


Figure 4.17: *Root mean square error using human perceived similarity as ground truth on (a) our system, (b) space-time proximity graphs [62], (c) joint position distance [66], (d) joint angle distance [65].*

list interaction identifications and each pixel represents the error value, with darker pixels representing the higher error. It is shown that our method aligns the most with human perceived similarity. The space-time proximity graph performs better than joint position and angle distances, but has a high error in some specific interactions.

To further evaluate the performance, we group the interactions according to Table 4.2. Fig. 4.18 shows the root mean square error obtained by all considered methods in the overall database and individual groups of interactions. It is shown that our method performs the best in all groups, and the space-time proximity graph does not work as well on complex interactions.

4.6.3 Interaction Comparison and Visualization

Here, we visualize how individual body parts are similar or different between two interactions in terms of how they interact with the opponent. More experiments can be found in the attached video.

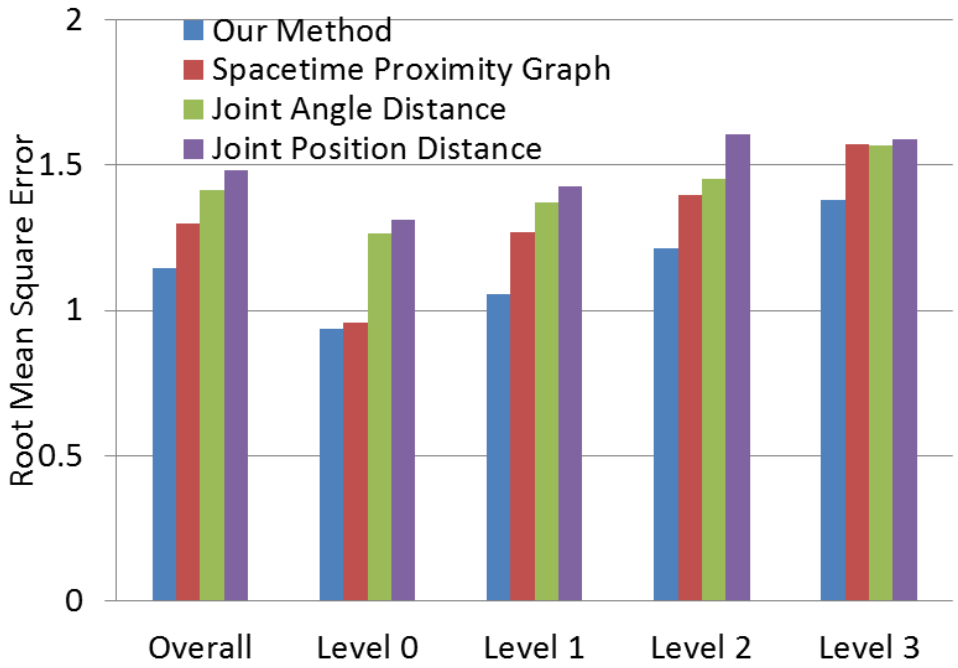


Figure 4.18: *Root mean square error for interaction groups of different levels of complexity.*

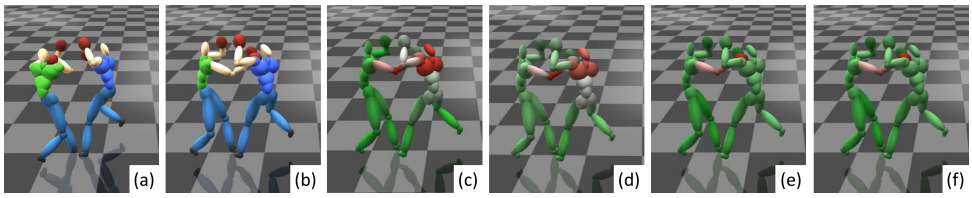


Figure 4.19: *Visualization of difference in interaction: (a) reference interaction (b) analyzing interaction (c) our method, (d) space-time proximity graphs [62], (e) joint position distance [66], (f) joint angle distance [65].*

Fig. 4.19 shows the case in which the reference interaction consists of an elbow punch, while analyzing one consists of a hook punch. While all methods can visualize the difference from the attacker point of view, only our method and the space-time proximity graph successfully identify the difference from the defender point of view. That is, the defender is being interacted by a different body part from the attacker.

Fig. 4.20 shows that during complex movement, our system outperforms the space-time proximity graph. Both interactions are about a kick that hits the opponent on the chest. However, the space-time proximity graph generates a false negative at the lower body. This is because during a close interaction, there is a lot

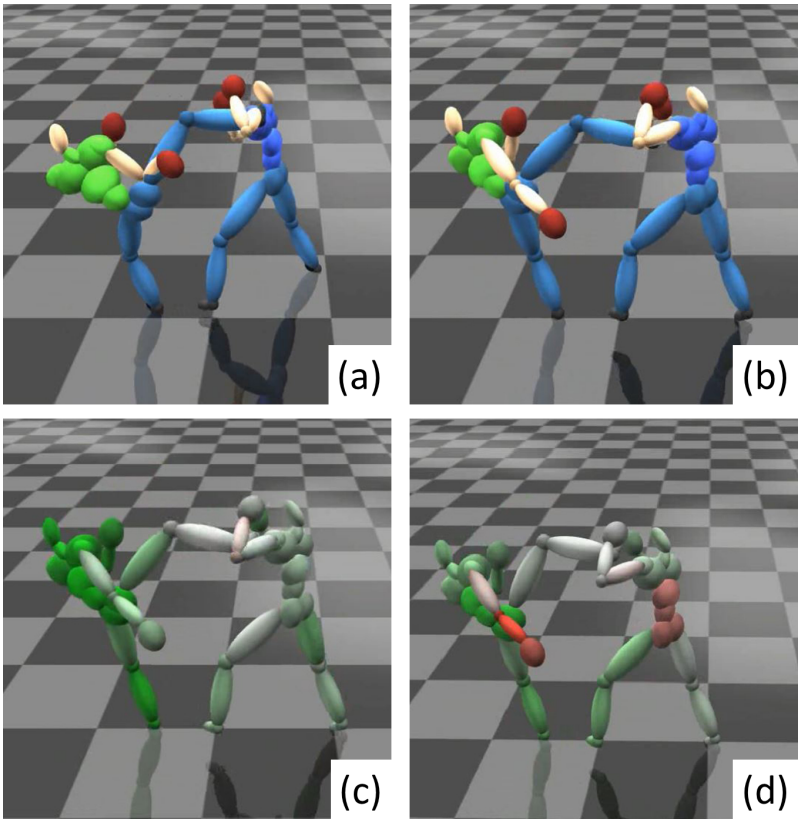


Figure 4.20: *Visualization of difference in interaction: (a) reference interaction (b) analyzing interaction (c) our method, (d) space-time proximity graphs [62].*

of topology change in the graph structure, and the topological distance term in that method performs inconsistently.

4.7 Conclusion and Discussions

In this work, we propose a new method for motion retrieval and motion analysis from the interaction point of view. This allows us to evaluate movement based on the high-level semantic meaning of the interaction. Our method can compare the interaction of different topology and discover their intrinsic semantic similarity. Experimental results show that it aligns with the human perception of interaction similarity. We demonstrate how our system can be used to retrieve semantically similar interactions, and suggest suitable interactions based on a set of user-defined constraints. We also demonstrate our system on visualizing how individual body parts are similar or difference between two interactions from the interaction point

of view.

Our system adapts Earth Mover's Distance to compare interaction graphs of different topology. Theoretically speaking, such a design can be applied to other features such as joint relative distance as well. However, we prefer the interaction graph structure as it can be used robustly in different kinds of interactions. It can also discover spatial proximity, which is one important aspect in defining interactions.

We apply Dynamic Time Warping (DTW) to align interactions temporally. While this allows us to effectively compare interactions that are performed at different speed and order, DTW has a side-effect of reducing the importance of temporal features.

Chapter 5

Data Driven Crowd Motion Control

In Chapter 3 and 4, the research progress of our efficient modelling method on the single person motion and multiple people interactions have been introduced, along with their potential applications in computer graphics. In this Chapter, I introduce our research idea in a more complex scenario, crowd motion. Controlling and simulating crowd motions are also very important in the research area of computer graphics. The ways of modelling crowd motions are the key components to facilitate such tasks. Quite different from the idea of modelling a single person motion / multiple people interactions, crowd motion is always modelled as an entity and the control schemes are designed on top of it. In this Chapter, the method of efficient crowd motion modelling and intuitive control are introduced.

Controlling a crowd using hand gestures captured by multi-touch devices appeals to the computer games and animation industries. First, multi-touch systems are getting more and more popular nowadays due to the advancement of hardware. Second, a crowd has a large degree of freedom, which is difficult to be controlled using traditional controllers with lower dimensional control signals, such as mice and keyboards. Multi-touch devices register several simultaneous control inputs, such that the user can control the complex formation of a crowd intuitively.

The hand gestures captured by multi-touch devices are typically sets of time series of finger positions. Many existing works show that it is possible to map such

control signals to a crowd motion using predefined control schemes [108, 109]. This allows the user to control the formation and movement of the crowd by performing specific gestures. While these manually designed control schemes are efficient in crowd control, different systems usually employ different control schemes. This is because there is an infinite number of possible mappings between the gesture and the crowd space. Rules need to be explicitly defined to fulfil the control needs optimally. As a result, the users have to learn the schemes in advance before using the systems. Unlikely previous work, we learn a mapping that focuses on both user-friendliness and control expressibility in this work to shorten the learning curve.

To this end, we present different crowd motions to a group of users and ask them to give their desirable control gestures, which allows us to generalize the preferred gestures and implement an intuitive control scheme. For every crowd motion in our training dataset, we ask the users to perform a control gesture that they think to be the best to create such a motion. It results in a database with pairwise samples of gestures and crowd motions. During run-time, we obtain a gesture from the user and find the K nearest gestures from the database. We then interpolate the corresponding K crowd motions in order to generate the run-time control. Since the control scheme is learned from different users without prior constraints, our system is intuitive to use.

One important component of our research is the gesture space representation. As we do not impose any constraints when collecting the control gestures, a representation invariant to individual gesture variations is needed, such as the number of fingers used, different speed, etc. Users often articulate gestures with one or both hands, using multiple fingers when performing similar tasks [142]. At the same time, they show similar variations in their gestures when asked to provide control for the movement of robot groups [143]. We propose a set of gesture features that effectively represents a wide variety of gestures while independent to inter-user style differences. This includes the centroid feature, the distance to centroid feature, the rotation feature and the minimum oriented bounding box features. We further propose a distance function to evaluate the distance between two gestures in such a space in order to obtain the K nearest neighbours of a run-time gesture.

Similarity, crowds under different scenarios contain variations such as the numbers of characters within the crowd, and therefore crowd motions also require a general representation. Such a representation should ideally parameterize the whole crowd motion space based on the crowd data. We propose a crowd motion feature space that models a crowd motion with a Gaussian mixture model (GMM), in which the trajectory of each character is modelled by the distribution of the Gaussian components. The major advantage of GMM is that we can set up multiple Gaussian components to accurately model the movement of small groups of characters within the crowd. We further propose a scheme to interpolate multiple crowd motion in the feature space in order to generate the run-time control signal.

We demonstrate that our system can appropriately output the crowd motion based on a given gesture. Users can effectively control a crowd of arbitrary size with intuitive gestures and guide the crowd to navigate through a given virtual environment. Our system is best to be applied in computer games like the crowd control systems in real-time strategy games, and in interactive character animation designs.

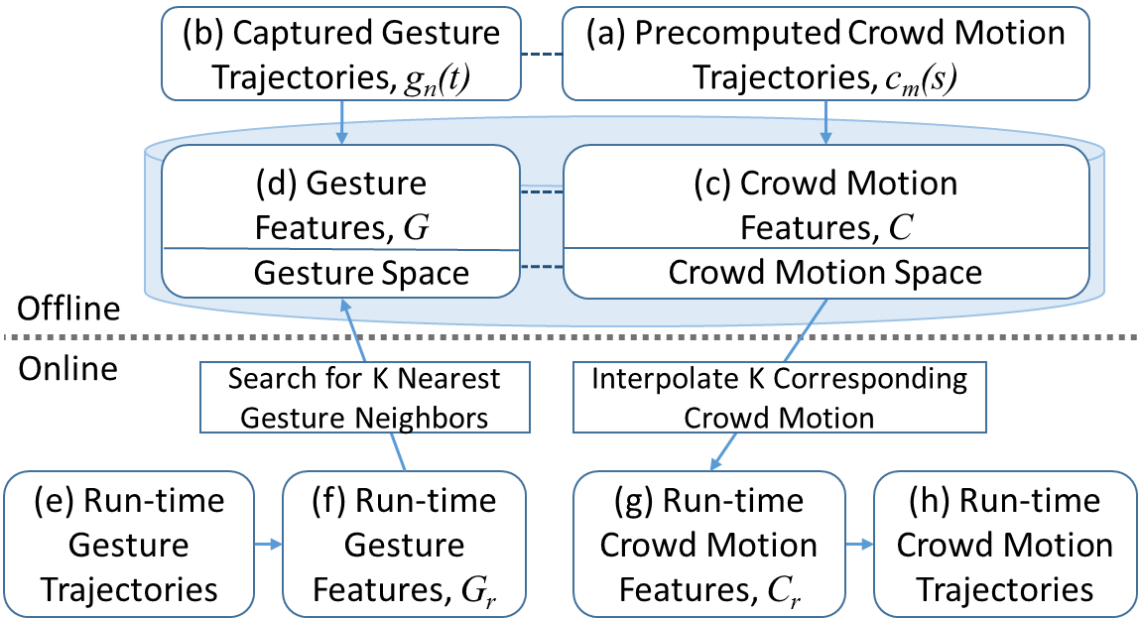


Figure 5.1: *The overview of our crowd control system.*

5.1 Contributions in This Chapter

This work presents the following contributions:

- We propose a data-driven method for inferring an appropriate crowd motion based on the gesture input obtained from a touch device. Our approach is not restricted by a pre-specified control scheme. Instead, the control scheme is learned as a mapping between user-preferred gestures and corresponding crowd motions, which encodes both user-friendliness and control expressibility.
- We propose a set of gesture features that are invariant to the variations of the user's preferred touch input style such as the number of fingers used. These features are used for recognising different properties of a user's multi-touch input, allowing the system to distinguish between a variety of control signals.
- We propose to represent crowd movement with a set of crowd motion features that are obtained from GMM. This representation allows modelling different sub-groups of the crowd and is independent of the number of characters. We further propose a method to interpolate crowd motion features in order to generate a new crowd motion that matches the user input the best.

5.2 Overview of Our Method

The overview of our system is shown in Fig. 5.1. In the offline stage, we collect user data that describe the mappings between gesture inputs and given crowd motions and create a database. We prepare a number of precomputed crowd motion trajectories (Fig. 5.1a) and obtain the corresponding gesture trajectories (Fig. 5.1b) from the users. As trajectory information has inconsistent dimensions and inefficient representation, we propose to map gesture and motion trajectories into their respective high-level feature space (Fig. 5.1c and d). The correlated gesture and crowd motion feature spaces generalize and unify the representation of the gesture and crowd data respectively. In the online stage, our system receives run-time user gestures and evaluate the corresponding crowd motion. Given the run-time gesture trajectories (Fig. 5.1e), we calculate its gesture features (Fig. 5.1f) and conduct

a K nearest neighbours (KNN) search in the database. This allows us to obtain K similar gestures and their corresponding K crowd motion. We interpolate the K crowd motion and generate the resultant run-time crowd motion features (Fig. 5.1g), which is finally converted into crowd motion trajectories (Fig. 5.1h) that can control the run-time crowd. Since the gesture data in the database are obtained from real user inputs with different variations, our system allows intuitive control of a crowd in real-time.

5.3 Data Collection

In this section, we explain how we collect user gesture data based on a set of pre-generated crowd motion data.

We first generate a set of crowd motions with the crowd simulation system presented in [108,109]. We created 150 motions under 10 different motion classes, which are shown in Fig. 5.2. Such a set of crowd motions consists of 6 classes of typical crowd motions, including *translate* (i.e. characters all moving in a direction), *twist* (i.e. characters moving in a circular direction around the centre of the scene), *contract* (i.e. characters moving towards the centre of the scene), *expand* (i.e. characters moving away from the centre of the scene), *join* (i.e. two groups of characters moving towards one another), and *split* (i.e. two groups of characters moving away from one another). It also consists of 4 classes of more complicated crowd motions, including *split then translate*, *translate then join*, *twist while expanding*, *twist while contracting*. The motion set is designed to demonstrate that our system can handle typical crowd motions seen in computer games and movies, as well as complicated motions that consist of combinations of typical motions. Our proposed framework is easily extensible. Developers can add or remove classes of crowd motions based on the requirement of the target application.

Ten participants, aged between 20 and 50, were presented with a subset of the crowd motions using a touch screen. They were asked to provide a corresponding hand gesture on the screen as if they were controlling each of such motions with two or more fingers. The participants were not given any instruction about what the

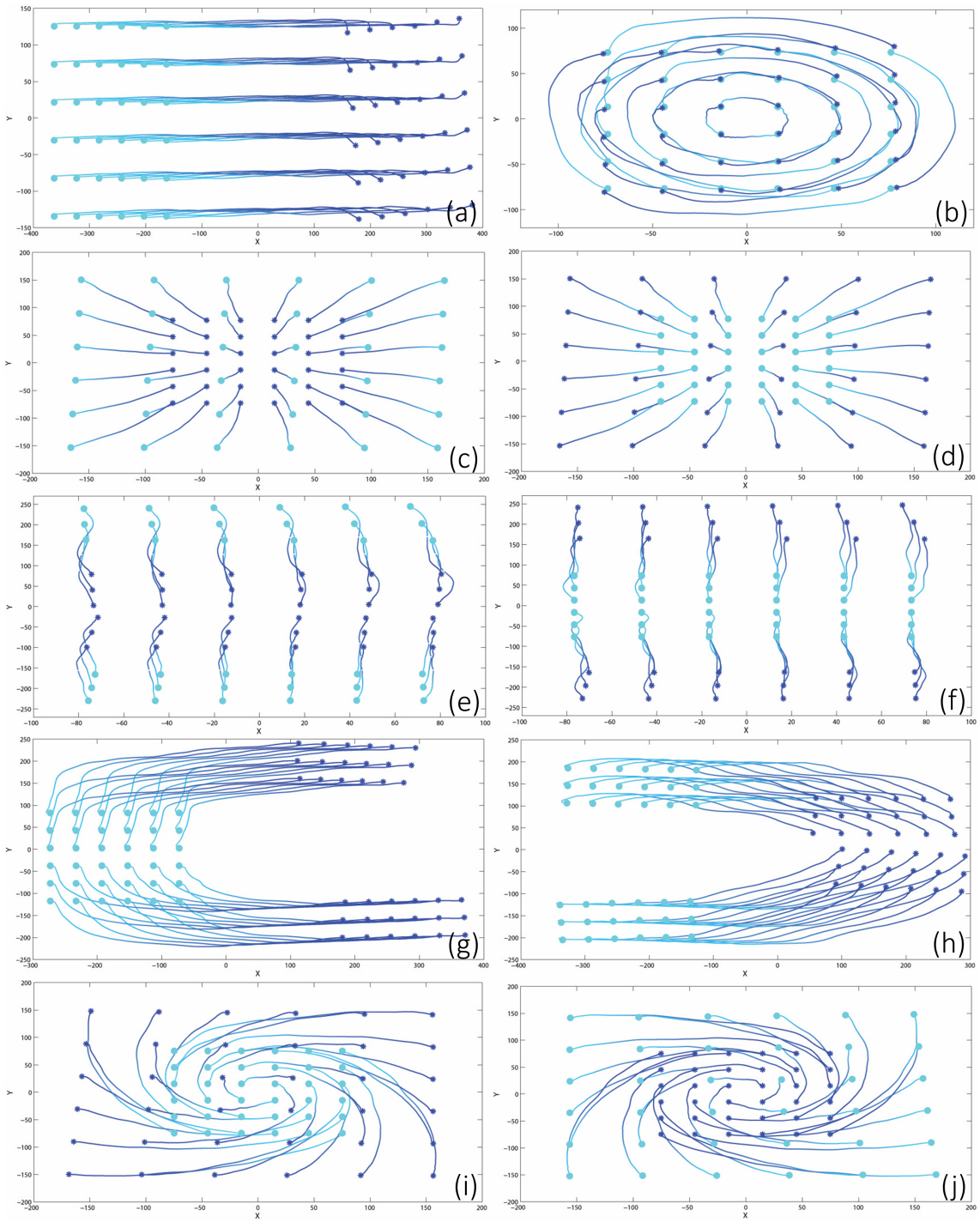


Figure 5.2: *Examples of crowd motion shown to users to collect their control gestures, with the light blue colour indicating the start of the motion and the dark blue indicating the end: (a) translate, (b) twist, (c) contract, (d) expand, (e) join, (f) split, (g) split then translate, (h) translate then join, (i) twist while expanding, and (j) twist while contracting.*

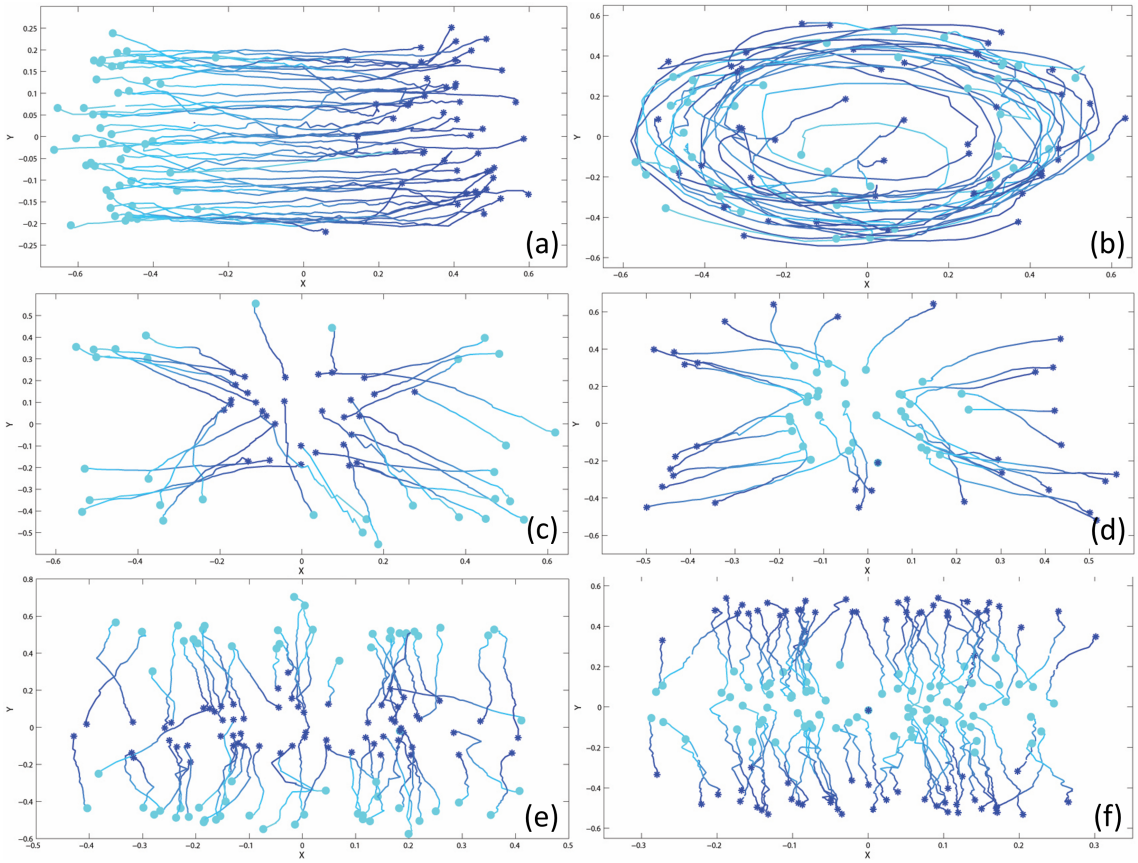


Figure 5.3: *Examples of collected user gesture for different crowd motions, with the light blue colour indicating the start of the motion and the dark blue indicating the end: (a) translate, (b) twist, (c) contract, (d) expand, (e) join, (f) split.*

gesture should be. They decided the number of fingers to be used, as well as the durations and the trajectories of the gestures. The orientation of the crowd motion on the screen was varied to prevent any bias in terms of the positioning of the hands when recording the gesture. This process results in 150 gestures, corresponding to the 150 motions in the crowd motion set. Fig. 5.3 shows some example input for the typical crowd motions.

5.4 Gesture Space

The success of finding a good mapping between the gesture and crowd motion space lies in their corresponding parametrization, such that the variation of the data can be captured. For gestures, we find that the combination of multiple features provides

a powerful representation. In this section, we propose a set of gesture features that can be extracted from raw gesture trajectories. Such features form the *gesture space*, which is a low dimensional, continues space in which each point represents one gesture. It allows us to compare and distinguish gesture effectively.

Our concept of a gesture space is similar to the idea of *Motion Fields* [144], in which the authors propose a high-dimensional continuous space that incorporates the set of all possible motion states in character motion. However, unlike character motion, the way a user performs a particular gesture can vary significantly from person to person. For example, users can use a different number of fingers to perform the same intended gesture. Our proposed gesture space consists of a set of features that are independent of such inter-user variations, while capable of capturing the intent of the input. This allows us to robustly distinguish between different types of user gesture.

5.4.1 Gesture Trajectories

Here, we define the representation of raw gesture trajectories and explain the process to resample the gesture with a spline function.

A raw gesture is described by the set of trajectories corresponding to the finger inputs provided by a user. The touch screen records the position of each touch points in discrete time intervals. As a result, a gesture G_{raw} is defined as a set of trajectories:

$$g_n(t) \quad \forall \quad n \in [1, N], \quad t \in [1, T_n], \tag{5.4.1}$$

where N is the total number of trajectories, T_n is the total number of time intervals (i.e. points) in the trajectory n, the representation $g_{n'}(t')$ indicates the 2D location of a specific trajectory n' at a specific time t'. Similar to existing research [92,93,145], we normalize the gesture by translating and uniformly scaling the whole gesture. In particular, the whole gesture is translated such that the minimum x and y position in the gesture is at the origin. We calculate a scaling factor λ to scale the gesture uniformly:

$$\lambda = 1/max(d_v, d_h), \tag{5.4.2}$$

where d_v and d_h are the maximum vertical and horizontal distance among all points respectively. After normalization, all trajectory points are within the range $[0, 1] \times [0, 1]$.

We assume all touch trajectories have a similar number of time intervals, as a gesture usually starts and ends with all fingers touching and leaving the screen at a similar time. This allows us to utilize spline functions for approximating and resampling touch trajectories to the same length. In our implementation, we apply the Hermite spline [146] to approximate each of the n touch trajectories. Then, we uniformly resample each trajectory from T_n points to T_H points. The choice of value for T_H is important since undersampling would remove too much information from the original gesture, but oversampling would add unnecessary details and increase computational overhead in later stages. We follow the suggestion in [145] and set $T_H = 64$, which works effectively in all of our experiments. As a result, we define a gesture G as:

$$g_n(t) \quad \forall \quad n \in [1, N], \quad t \in [1, T_H], \tag{5.4.3}$$

where T is the pre-defined sample number.

There are multiple advantages for approximating and resampling the gesture with a spline function. First, different touchscreens have a slightly different sample rate. Resampling the trajectories allows the system to work robustly with different hardware. Second, it unifies the density of points in a trajectory, which helps us to more accurately identify a gesture using a gesture database. Third, from our discussion with practitioners, crowd control is usually based on the geometry of the trajectories instead of the speed of performing them, as the movement speeds of a crowd are usually constrained in graphics systems. Resampling the trajectories allows us to remove the speed factor from the trajectories. If the gesture speed is needed, it can be calculated before the resampling stage and stored as an extra feature.

5.4.2 Gesture Features

Here, we define a set of high-level gesture features extracted from the gesture trajectories. Such features are designed to represent the essential components of a gesture

in low dimension, making them effective in identifying gestures. Also, they are independent of the number of touch points. As a result, with gesture features, gestures of different touch points can be directly comparable.

The **centroid feature** represents the average position of the user's touch inputs over time. It captures the general shape of the gesture and is independent of the number of touch points. It is defined as a column vector:

$$C(G) = [c_G(1), c_G(2), ..., c_G(T_H)]^T, \tag{5.4.4}$$

where $c_G(t)$ is the centroid at time t:

$$c_G(t) = \frac{1}{N}\sum_{n=1}^{N} g_n(t). \tag{5.4.5}$$

The **distance to centroid feature** represents the distance of each touch point relative to the centroid over time. It allows us to capture the spread of the gesture. It is defined as:

$$S(G) = [s_G(1), s_G(2), ..., s_G(T_H)]^T, \tag{5.4.6}$$

where $s_G(t)$ is the spread at time t:

$$s_G(t) = \frac{1}{N}\sum_{n=1}^{N} |g_n(t) - c_G(t)|, \tag{5.4.7}$$

where $|*|$ represents the Euclidean norm, $c_G(t)$ is calculated in Eq. 5.4.5.

The **rotation feature** represents the average cumulative change in rotation over time of the touch inputs around the centroid. Such a feature allows us to capture the overall rotation in the gesture. It is defined as:

$$R(G) = [\sum_{t=0}^{0} r_G(t), \sum_{t=0}^{1} r_G(t), ..., \sum_{t=0}^{T_H} r_G(t)]^T, \tag{5.4.8}$$

where $r_G(t)$ is the average change in rotation at time t:

$$r_G(t) = \begin{cases} 0, \text{ if } t = 0 \\ \frac{1}{N}\sum_{n=1}^{N} \arctan \frac{(g_n(t)-c_G(t))\times(g_n(t-1)-c_G(t-1))}{((g_n(t)-c_G(t))\cdot(g_n(t-1)-c_G(t-1))}, \\ \text{otherwise} \end{cases} \tag{5.4.9}$$

The **minimum oriented bounding box feature** represents the minimum and maximum dimension of the minimum oriented bounding box (MOBB) of the touch

inputs at each time step. It allows us to represent the movement variation of the gesture over time, which can approximate the area of the gesture. Given a set of touch points at time t, $g_n(t)$, we apply the rotating calipers method [147] to calculate the minimum rectangle bounding the points. We extract the width, $b_w(t)$, and the height, $b_h(t)$ of the rectangle, and define the feature as:

$$\begin{aligned} B(G) = [&(b_w(0), b_h(0)), (b_w(1), b_h(1)), \\ &..., (b_w(T_H), b_h(T_H))]^T, \end{aligned} \tag{5.4.10}$$

Finally, the gesture space is formed by concatenation of the four gesture features. As a result, a gesture G can be represented by a point in the space with the feature vector:

$$G = [G(G), S(G), R(G), B(G)]^T. \tag{5.4.11}$$

5.4.3 Distance between Two Gestures

Here, we explain how we compare gestures using gesture features in the gesture space.

Given two gestures G_0 and G_1, we define the distance as

$$\begin{aligned} D(G_0, G_1) \ = &\alpha \, \mathrm{DTW}(C(G_0), C(G_1)) + \\ &\beta \, \mathrm{DTW}(S(G_0), S(G_1)) + \\ &\gamma \, \mathrm{DTW}(R(G_0), R(G_1)) + \\ &\delta \, \mathrm{DTW}(B(G_0), B(G_1)), \end{aligned} \tag{5.4.12}$$

where DTW provides a distance between two vectors using dynamic time warping [148], and α, β, γ, and δ, are weights for each feature. We empirically found that $\alpha = 0.04$, $\beta = 0.36$, $\gamma = 0.36$, and $\delta = 0.24$ work well in our dataset.

It is possible to combine all 4 features into one vector and then do DTW based on it. However, for more complex features such as spreading fingers while moving all touch points to the right, people may perform different features in different speed. For example, one may spread very quickly while moving slowly, while another may spread very slowly while moving quickly. So we need to align individual features independently.

The feature set and distance metrics together determine the well-represented gesture space where algebraic operations can be sensibly defined. It forms the basis of the control scheme learning in later sections.

5.5 Crowd Motion Space

In this section, we present our formulation of a *crowd motion space*, which is conceptually similar to a *gesture space*. We consider the set of movement trajectories from the characters of the crowd and represent the overall crowd movement with a set of features modelled by a mixture of Gaussian processes.

5.5.1 Crowd Motion Trajectories

Here, we represent the motion of a crowd using the trajectories of the characters in the crowd.

A crowd motion C is defined as a set of trajectories:

$$c_m(s) \quad \forall \quad m \in [1, M], \quad t \in [1, S], \tag{5.5.13}$$

where M is the total number of character in the crowd, S is the duration of the crowd motion, the representation $c'_m(s')$ indicates the 2D location $(c'_m(s').x, c'_m(s').y)$ of a character m' at time s'. Similar to the gesture trajectories, we normalize the crowd motion trajectories by translating the whole motion such that the starting point is at the origin.

We also resample the crowd motion trajectories from S points to S_H points using the Hermite spline [146] and set $S_H = 64$ [145], as we do for the gesture trajectories in Sec. 5.4.1. As a result, a crowd motion C is defined as:

$$c_m(s) \quad \forall \quad m \in [1, M], \quad t \in [1, S_H]. \tag{5.5.14}$$

For the sake of calculation simplicity, we express the trajectory of the m'^{th} character, $c_{m'}(s)$, as a vector of serialized X and Y positions:

$$c_{m'}(s) = [c_{m'}(1).x, c_1(1).y, c_{m'}(2).x, c_1(2).y, \cdots, c_{m'}(S_H).x, c_{m'}(S_H).y.]^T. \tag{5.5.15}$$

5.5.2 Crowd Motion Features

Next, we present our crowd motion features that describe the high-level features of a moving crowd. Such features are independent of the number of characters in the crowd. They can also be used to interpolate two crowd motions in order to generate new ones.

Since the character trajectories in a crowd are controlled by one input gesture, we assume that there exists a linear low dimensional space that can represent the trajectories of all characters. Trajectories can be treated as functions. Essentially, each crowd motion is a series of 2D functions that define the trajectories of all characters. This allows us to construct a low-dimensional space and represent the motion trajectories of all characters using Functional Principle Component Analysis (FPCA). FPCA projects a group of functions linearly into space where a mean function and functional variations serve as the basis of function representation, similar to PCA but on a function level. The mean function, $\bar{c_s}$ where $s \in [1, S_H]$, can be computed by averaging the motion trajectories of all characters. Then, a set of eigenfunctions, E_V^C, and a set of eigenscores, E_S^C can be computed. The eigenfunctions describe the principle movement over time of all characters in the crowd, and the eigenscores represents how the movement of a character can be projected into the low dimensional space. The trajectory of the m'^{th} character can be recovered as:

$$c_{m'}s = \bar{c_s} + E_V^C E_{S_{m'}}^C, \tag{5.5.16}$$

where $E_{S_{m'}}^C$ is the Eigenscore of the m'^{th} character.

Figure 5.4 gives an intuitive illustration on how FPCA is used to map the original crowd motion, "translate" in this case, into the low dimensional space. On the right side of the figure, the red arrow represents the mean movement of the whole crowd and green arrows (Eigenfunctions) represent the major variation components among the mean movement. Eigensocre is a mapping function to map the original variation of the crowd movement into eigenfunctions.

Although FPCA gives a compact representation, it does not generalize enough to take all the input variations into account such as different numbers of trajectories or style variations of the same motion. This motivates us to further generalize the

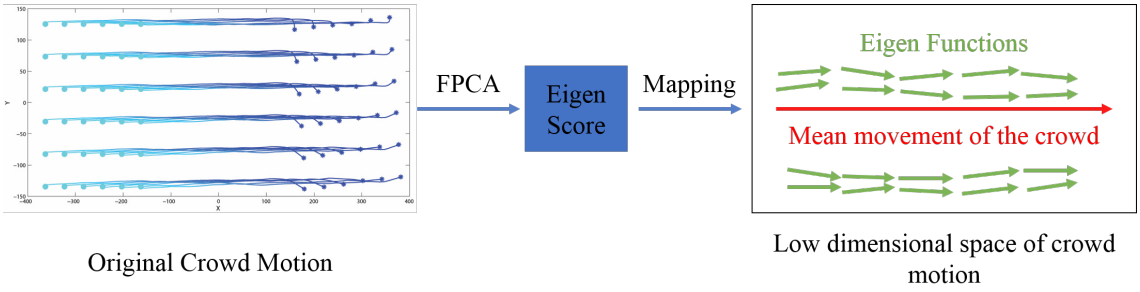


Figure 5.4: *The illustration of using FPCA to map original crowd motion into low dimensional space.*

representation. We discover that the high-level visual observation of the general motions can be described by the eigenscore distributions. As a result, modelling the crowd motion trajectories of the whole crowd can be considered as modelling the distribution of the eigenscores of the characters. This high-level model allows us to interpolate the distribution of eigenscores, instead of the actual trajectories, between two crowd motions effectively. In addition, such a distribution-based representation does not depend on the number of characters and does not explicitly map the trajectories of the characters from one crowd to another.

Since the eigenscores of a group of similar motions usually exhibit multi-modality, we propose to use GMM to model the distribution of the eigenscores. There are three main advantages. First, the non-linearity of GMM fits the trajectory data well. Second, the multi-modality nature of GMM captures semantic-level meanings such as the crowd being split into multiple groups, which cannot be modelled easily with a single model. This is particularly relevant to crowd motion such as splitting and joining. Finally, multiple GMMs can be easily interpolated and the interpolation has visual as well as semantic meanings, which is important to generate new crowd motions.

There are two import issues in applying GMMs to model the data, which are the optimal parameters and the number of components of the model. We apply the Expectation-Maximization algorithm [149] to optimize the parameters for the distribution of eigenscores, $\phi(E_S^C)$. The component number essentially allows the system to accurately model multiple sub-groups in the crowd motions. In theory,

it is possible to automatically determine that by iterating from one and choose the smallest value that reaches the required data likelihood. In practice, we found that users rarely split a crowd into more than two subgroups, even with two hands controlling the crowd. As a result, two Gaussian components are enough to model our database. For simpler motion with only one sub-group, the two components in the GMM blends together to represent the distribution of character trajectories. If more complicated crowd motions with multiple sub-groups of characters are involved, an analysis of data likelihood should be performed and more components can be used.

Therefore, the crowd motion features of a crowd C is defined by a vector:

$$C = [\bar{c_s}, E_V^C, \phi(E_S^C)]^T, \tag{5.5.17}$$

where $\bar{c_s}$ is the mean trajectory, E_V^C is the set of eigenvectors, and $\phi(E_S^C)^T$ is the distribution of the eigenscores modelled by GMM. Conceptually, our crowd motion feature is similar to the *morphable motion primitives* [150, 151]. The difference is that it is applied on a crowd instead of an individual motion.

Here, we include an optional step to improve the performance of our system. We observe that there is an intrinsic redundancy in the crowd motion trajectories as the characters' trajectories are not arbitrary. Therefore, it is not necessary to use all the eigenvectors E_V^C as the features. In fact, we only use the first 15 principal components returned by FPCA and the recovered trajectories from Eq. 5.5.16 achieves $< 1\%$ error for all the crowd motion in our database. This not only reduces computational cost but also removes noises that may exist in the motion data.

5.6 Crowd Motion Control

In this section, we explain how a run-time gesture can be identified based on the set of gestures in the database. Then, we explain how such a gesture generates the corresponding crowd motion.

5.6.1 Run-time Gesture Representation

Here, we explain how we represent a gesture using neighbour ones in the database, which allows us to understand the crowd motion the user intended to perform.

We have collected a set of gestures with the corresponding crowd motions as explained in Sec. 5.3. The gesture space is non-linear due to the complex nature of hand gesture. Modelling the whole space with high degree non-linear functions would require a large amount of gesture data, which is labour intensive to obtain and would limit the feasibility to increase the gesture types. Instead, we propose to model a local area of the gesture space that is relevant to the run-time gesture using a linear function. Such a method works robustly even with smaller database and generates reliable results.

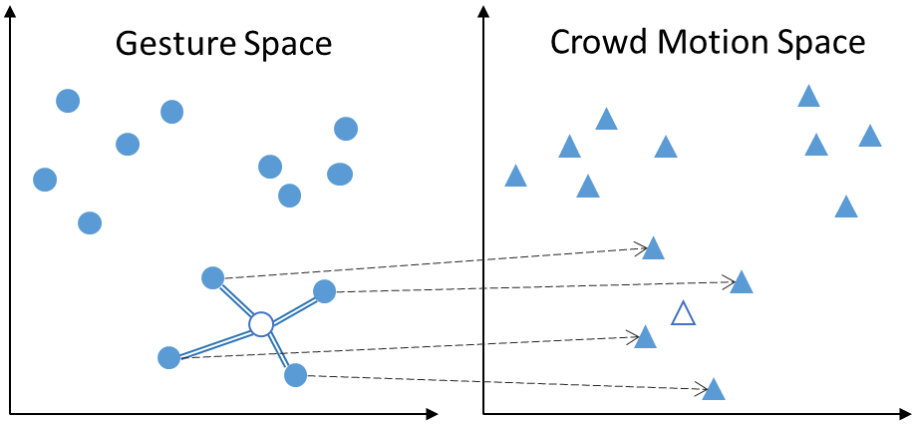


Figure 5.5: *Generating crowd motion with the run-time gesture. The circles represent gestures in the gesture space, with the hollow one representing the gesture obtained in run-time. Based on the run-time input, we obtained the K nearest gestures, visualized by the double lines. The triangles represent crowd motion in the crowd motion space. We find the crowd motions corresponding to the K nearest gestures, pointed out by the black arrows. We finally interpolate these crowd motions to create the run-time crowd motion represented by the hollow triangle.*

In particular, given a run-time gesture, G_r, we utilize Eq. 5.4.12 to evaluate its distance with the stored gestures in the database. We represent G_r using a set of K nearest neighbours, $G_k \ \forall \ k = [1, K]$. The neighbours are associated with the corresponding weights, $w_k \ \forall \ k = [1, K]$, which are inversely proportional to

the distance with respect to the runtime gesture. The particular weight $w_{k'}$ that is corresponding to the gesture $G_{k'}$ is defined as:

$$w_{k'} = \frac{1}{D(G_r, G_{k'})} / \sum_{k=1}^{K} \frac{1}{D(G_r, G_k)}, \tag{5.6.18}$$

where the summation term acts as a normalization factor to ensure that all the weights sum up to 1.0. In our experiment, we found that $K = 10$ produces good results. This process is visualized in the left part of Fig. 5.5.

Since our gesture database is relatively compact, a brute force search is quick enough to find the K nearest neighbours in real-time. For a larger database, we may organize the database with data structures such as the k-d tree to speed up searching.

While it is possible to apply methods such as regression to evaluate the run-time gesture, we find that KNN is the most reliable way, mainly because our gesture database contains a variety of gestures, where the sample size is big enough to locally approximate the gesture manifold as hyper-planes. In theory, if the database is dense enough, it could be possible to use the most similar gesture only. However, KNN is more robust against outliers, and constructing a dense database is labour intensive.

5.6.2 Run-time Crowd Motion Creation

Here, given the K nearest neighbours of the run-time gesture, we interpolate the corresponding K crowd motions in the database in order to generate the run-time crowd motion.

Given a run-time gesture, the obtained K nearest gestures, $G_k \ \forall \ k = [1, K]$, are corresponded with K crowd motions, $C_k \ \forall \ k = [1, K]$, according to the database. The run-time crowd motion, $C_r = [\bar{c_s^r}, E_V^{C^r}, \phi(E_S^{C^r})]^T$, is evaluated as the weighted sum of the K crowd motions. This process is visualized in the right part of Fig. 5.5. Such an interpolation involves interpolating the crowd motion features individually as follows.

The run-time mean crowd trajectory can be obtained by vector sum, as all mean

trajectories in the database has the same size S_H:

$$\bar{c_s^r} = \sum_{k=1}^{K} w_k \bar{c_s^k}. \tag{5.6.19}$$

Similarly, we interpolate and create a new set of eigenvectors:

$$E_V^{C^r} = \sum_{k=1}^{K} w_k E_V^{C^k}. \tag{5.6.20}$$

To ensure orthonormalization of the new eigenvectors, we apply the modified Gramm-Schmidt method presented in [152].

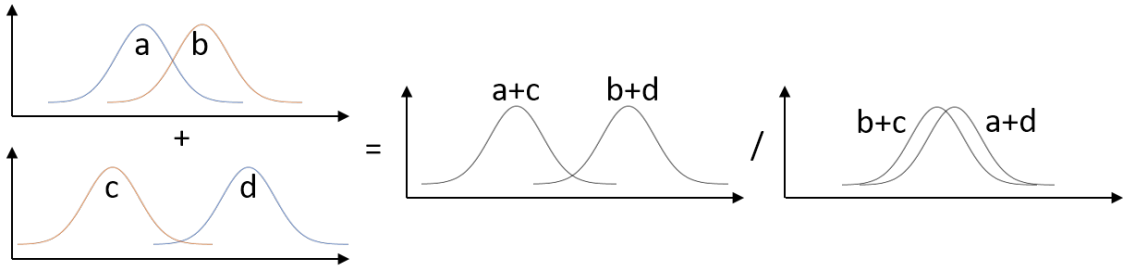


Figure 5.6: *Cross-fading problems of interpolating multiple Gaussians: (Left) Mixing two GMM (each with two components) can generate different results depending on how the Gaussian components are matched. (Middle) The desired result that retains the features of the source GMMs. (Right) The undesired result of cross fading.*

The blend weights w_k is important to ensure the quality of the resultant GMM, which account for the naturalness of the generated crowd motion. Considering that our gesture-motion pairs in the database are very distinctive and that both the gesture space and crowd motion space can be modelled by the local hyperplane, we use $w_{k'}$ in Eq. 5.6.18 as the blend weights for the crowd motion w_k. The underlying assumption here is that similar gesture in the gesture space would indicate similar crowd motion in the crowd motion space.

Finally, we propose a mass transport solver based method to combine multiple distributions of eigenscores and generate $\phi(E_S^{C^r})$. A naive one-to-one combination of the Gaussian components of two GMMs does not work well. As shown in Fig. 5.6, assuming each GMM has two components, depending on how we match the

components, blending two GMMs has two possible outputs. One of them retains the features from the source GMM, while the other does not as Gaussian components of very different parameters are blended, resulting in a scenario known as cross fading.

We follow the displacement interpolation method presented by [153] here. First, given two GMMs, we establish the correspondence of their Gaussian components. Each Gaussian component is defined by a mean value and a covariance value. We evaluate the correspondence using the mass transport solver [108], in which the source and target are set as the Gaussian means of the Gaussian components. Second, we produce a weighted sum of the Gaussian mean and covariance of each matching Gaussian component, in which the weights are obtained by Eq. 5.6.18, in order to generate a combined GMM. We iteratively combine all the GMMs in the K nearest crowd motions and obtain $\phi(E_S^{C^r})$.

5.6.3 Crowd Motion Synthesis

Here, we explain how we apply the crowd motion created in the last section to control a run-time crowd.

Assume that the user is controlling a group of M characters. Given a user gesture, we obtain the corresponding crowd motion $C_r = [\bar{c_s^r}, E_V^{C^r}, \phi(E_S^{C^r})]^T$ as explained in Sec. 5.6.2. We first utilize the distribution of the eigenscores, $\phi(E_S^{C^r})$ to sample M eigenscores. Second, we apply the eigenscores with the mean trajectory $\bar{c_s^r}$ and eigenvectors $E_V^{C^r}$ to generate M crowd motion trajectories using Eq. 5.5.16. Third, we apply a mass transport solver [108] to find out the optimal matching between the controlling characters and the calculated crowd motion trajectories. This is done by setting the positions of the characters as the source and the starting points of the trajectories as the target. By using the mass transport solver to evaluate the matching, we avoid visual artifact in which characters have to move a long distance before reaching the starting point of their corresponding trajectories. Finally, the characters move to the starting point of their respective trajectories, and then follow the trajectories, in order to produce the overall motion.

For handling collision detection and avoidance, we implemented the high-level crowd motion synthesis and the low-level character collision avoidance as two sep-

arate levels. The high-level system provides the desired position of all characters in the crowd, while the low-level system resolves their positions locally. In our experiments, the low-level system considers the potential penetration among characters and resolves the penetration by estimating a new position with a spring-mass model based on the penetrated depth and direction. Other more advanced collision avoidance systems can be directly employed in our framework.

We apply the full body motion synthesis method in [40] as an offline process to generate full-body running motion based on the point-based movement trajectories. This involves creating a motion graph that consists of different running actions and evaluating the optimal action to perform in order to follow the trajectories. We also apply the physical modelling method in [121] to create physically plausible movements. This allows us to resolve body part level collisions and penetrations effectively.

5.7 Experimental Results

In this section, we provide both qualitative and quantitative evaluations of our proposed system.

All the experiments are run with one core of a Core i7 2.67GHz CPU with 1GB of memory. For the multi-touch input, we used a G4 multitouch overlay from PQ labs, attached to a 24" Acer S240HL LCD monitor. In general, the system runs in 40 frames per second, which is higher than the real-time requirement of 30 frames per second. However, there is a slow-down when computing a new crowd motion from a hand gesture, which takes 330 ms, including 300 ms for the KNN searches, 12 ms for generating the crowd motion features, 4 ms for generating and assigning trajectories to characters. We believe that adapting a multi-thread implementation framework can create a more consistent frame rate. Also, more efficient search algorithms such as k-d tree search can further reduce the computational time.

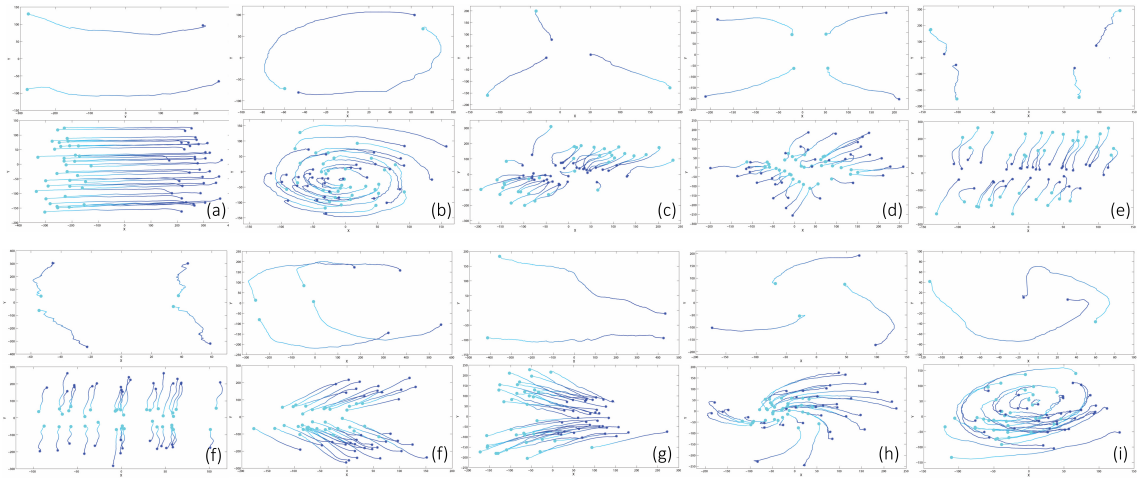


Figure 5.7: *Examples of user input (upper of each sub-image) and the corresponding crowd simulation (lower of each sub-image) for (a) translate, (b) twist, (c) contract, (d) expand, (e) join, (f) split, (g) split then translate, (h) translate then join, (i) twist while expanding, and (j) twist while contracting. The light blue color indicates the start of the input/crowd motion and the dark blue color indicates the end.*

5.7.1 Qualitative Evaluations

Here, we evaluate our system qualitatively with different experiments. The readers are referred to our supplementary video for more results.

First, we evaluate the effectiveness of our method by producing a set of crowd motions from a number of user inputs. Fig. 5.7 shows some user gestures and their corresponding simulated crowd motion. Our system generates crowd motions that accurately reflect the different user gesture types. It also works well under different initial positions of the characters. The number of touch points provided for the gestures does not affect our system's ability to produce the appropriate crowd motion.

We setup some virtual environments and ask a user to use our system to control the navigation of the crowd. Fig. 5.8 (upper) shows a corridor environment. The initial crowd cannot fit through the narrow corridor. The user, therefore, applies a *contract* gesture to reduce the size of the crowd, and two *translate* gestures to move the crowd through the corridor. Finally, the user applies an *expand* to expand the

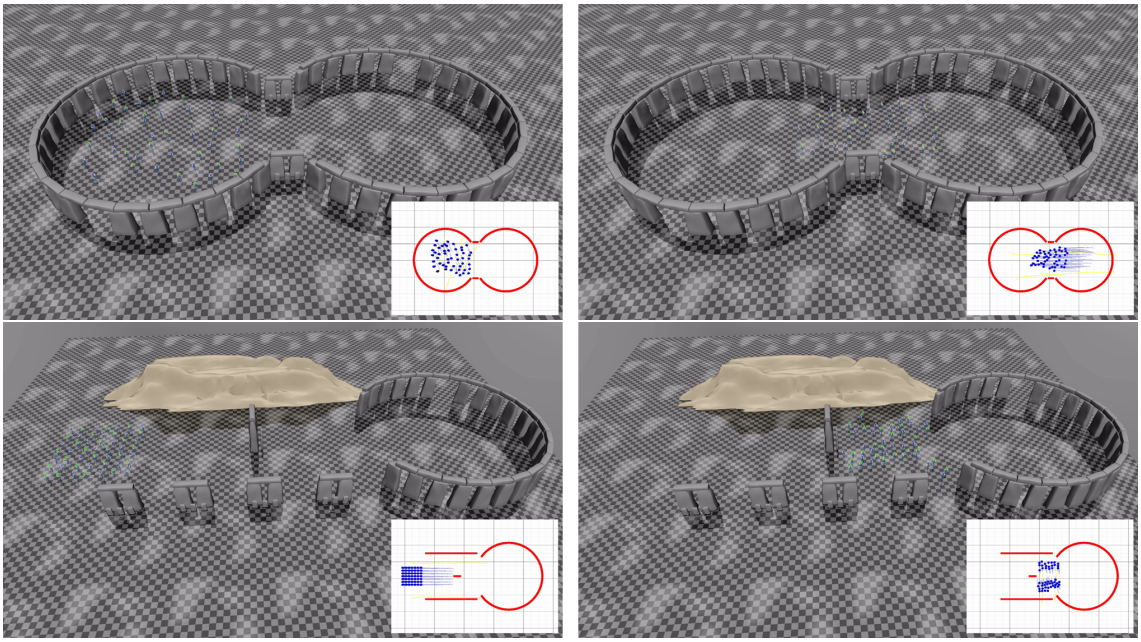


Figure 5.8: *Screenshots of controlling the crowd in complex environments: a crowd to navigate through (upper) a corridor environment, and (lower) a more complicated environment with an obstacle.*

crowd to its original size. Fig. 5.8 (lower) shows a more complicated environment, in which there is an obstacle in the middle of a corridor. The user successively applies the gestures *translate*, *split*, *translate*, *join* and *translate* such that the crowd can avoid the obstacle and reach the other side of the environment. The user finally applies a *twist* gesture such that the crowd can rotate inside the circular environment. These experiments show that our system can potentially be applied to console games that require crowd control, such as real-time strategy games.

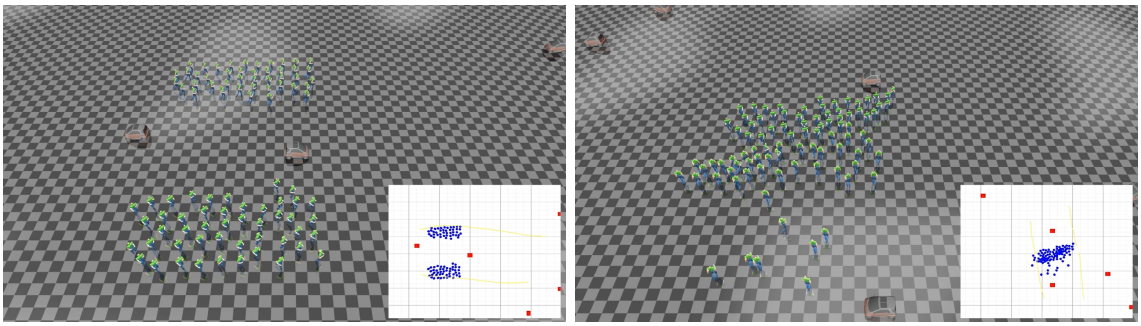


Figure 5.9: *Screenshots of a user controlling a crowd in a complicated scenario with dynamic obstacles.*

We generate a high-quality, complicated scenario in which 100 characters avoid a number of dynamic moving cars, as shown in Fig. 5.9. The user controls the crowd movement with our touch-based system that offers intuitive control of the timing for the change of formation. Multiple gestures are required to steer the crowd. This kind of real-time, precise, interactive control is difficult to be achieved with existing systems. As this demo focuses on demonstrating the animating power of the system for generating realistic scenes, we implement a Gaussian filter to smooth the crowd motion transitions. Gaussian filter has been widely used in motion stitching to smooth out glitches between different motion clips [154–157]. Motion Stitching means combining multiple short motion clips to generate a long motion sequence. It is a very popular techniques in computer graphics to create animations. Here we also use Gaussian filter to smooth out the glitches between different crow motion segments to create a high quality crowd motion sequence.

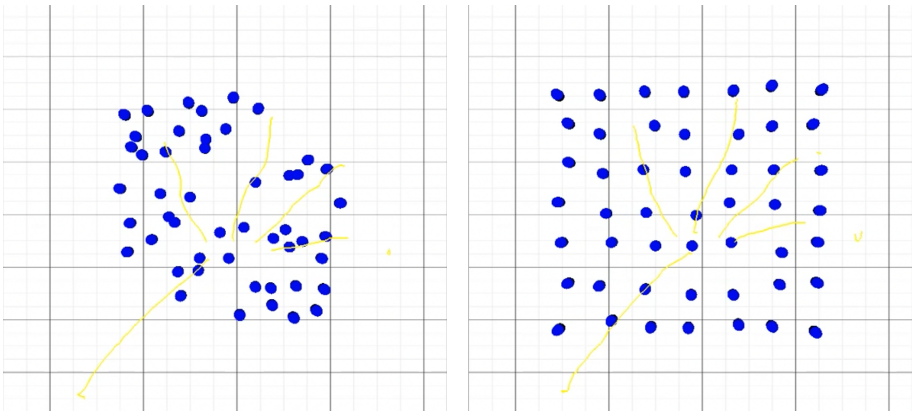


Figure 5.10: *The synthesized **expand** crowd motion using the database built with (left) Henry et al. and (right) RVO2.*

While we propose to utilize [108, 109] to generate examples for constructing the crowd motions in the database, the overall framework is independent of the underlying method to generate the crowd motions. Basic crowd simulation systems that control characters by setting the starting and goal positions can effectively generate the database and produce comparable results. To demonstrate this, we perform an experiment to utilize the Reciprocal Velocity Obstacle (RVO) 2 system [158] to generate the crowd motion database and synthesize new crowd motions based on the user inputs. We compare the newly created results with those generated

by our existing database, as shown in Fig. 5.10. We find that while the two databases result in crowds of different behaviour due to the different training data, the resultant crowd motion quality is comparable. This demonstrates the generalizability of our control system.

5.7.2 Quantitative Evaluations

Here, we give an illustration on how our proposed gesture features are discriminative on different types of users' input gestures. Fig. 5.11 respectively shows the proposed features (Section 5.4.2) on different types of users' input. In Fig. 5.11(a), The user gestures for controlling the "translate" crowd motion show the most significant change in the centroid of the users touch inputs. The "expand" and "contract" motions both indicate a large variation in centroid position along the y axis. In Fig. 5.11(b), this feature clearly shows that it can distinguish between three significant subsets of gesture styles, coupling the join & contract, split & expand, and the translate & twist types of gesture. It is obvious that this feature is capable of distinguishing those types of users' input into 3 different subsets. As a consequence, this feature is unable to separate the inputs inside this subset but it is considered by using other proposed features. In Fig. 5.11(c), most of the gesture types show an insignificant rotation of user inputs. But there are large rotations which are shown by "twist" and some of " translate" motions. This feature indicates the presence of a twist style in user input and for distinguishing this from other styles of gestures. In Fig. 5.11(d), the Minimum Oriented Bounding Box (MOBB) feature is able to clearly separate between the contract & expand gestures from the split & join gestures, something that is not seen in the basic Distance to Centroid feature shown in Fig. 5.11(b).

According to the above quantitative results, each proposed feature has a strong capability to distinguishing some specific types of users' gestures. We linearly combine these features so that they are complementary to be discriminative on majority types of users' input gestures.

In order to test if our proposed gesture features are discriminative, we conduct leave-one-out cross validation using the gestures for the 6 types of typical crowd

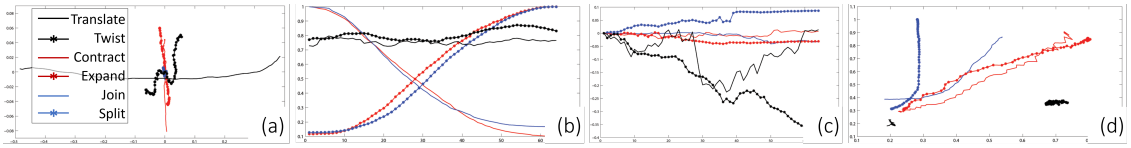


Figure 5.11: *Proposed features in Section 5.4.2 on different types of users' input. (a) Centroid feature, (b) Distance to Centroid feature, (c) Rotation feature and (d) Minimum oriented bounding box feature.*

motions (i.e. translate, twist, contract, expand, join, split).

We first use the gesture features of one gesture as testing data for classification, and that of all other gesture as training data. We then obtain the K nearest gestures. Within them, we conduct a weighted nearest neighbour voting with the weight obtained from Eq. 5.6.18, where the gesture type with the highest total weight is considered to be the recognized type. We finally check if such a type is the same as the real gesture type of the testing data. We iteratively evaluate all gestures and calculate the average accuracy.

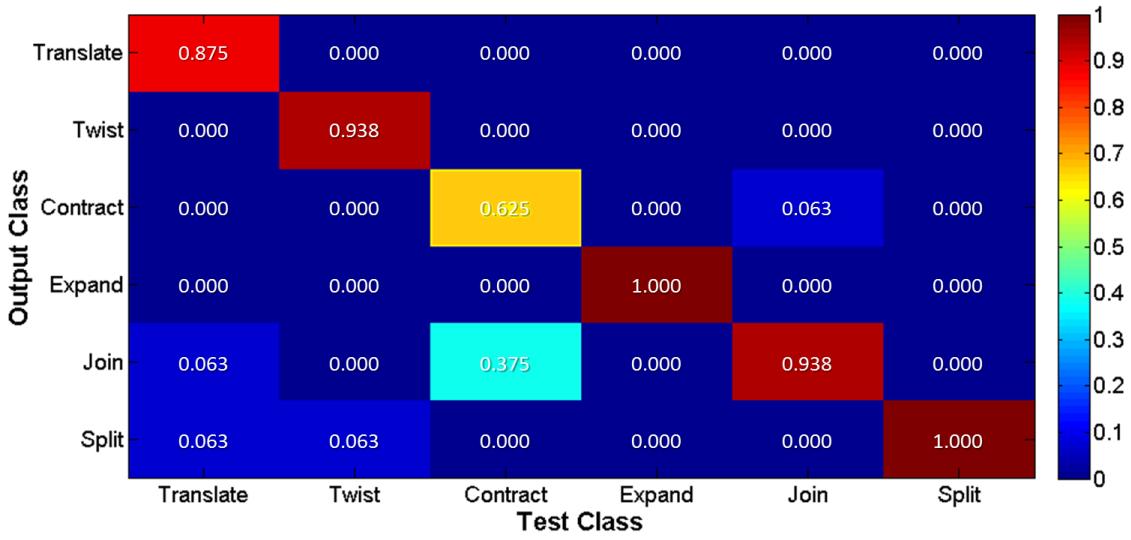


Figure 5.12: *Confusion matrix of 6 typical motion types. The cell in column i, row j indicates the proportion of all i^{th} test gestures recognized as the j^{th} output gesture.*

Fig. 5.12 shows the confusion matrix of this analysis. It shows that the proposed gesture features are discriminative in order to accurately identify an unknown gesture. The average classification accuracy is 87.5%. For all gesture types except

contract, the accuracy is over 89.6%. The contract type has a lower accuracy of 62.5%, as some of the gesture samples are very similar to those in the join type.

5.8 Limitations

Our system does not consider the mapping between the gestures and the full-body motions of the characters. Although this is an interesting idea, such a mapping will suffer from the ambiguity such as the walking phase as presented in [159], and extra parameter inputs will be required. Instead, the detailed movements (e.g. walking or jumping motions) are modeled by another sub-system given the computed trajectories. Splitting the mapping into two sub-systems leaves the degrees of freedom to the animators for designing their preferred movements. This idea is similar to the framework proposed by [109].

Data collection could be non-intuitive. The final motion quality depends on the representativeness of the database. Currently, the crowd motions can be easily generated because most of the trajectories can be represented by some close-form function plus noises. However, to collect more complex gestures, more work needs to be done by generating corresponding crowd motions. The situation could be mitigated by two facts. First, the system once trained can be used everywhere so no re-training is needed. Second, for complex motions, they could be decomposed into simpler ones which we already have in the database.

We only consider zero-order information (i.e. positions). Higher-order information such as motion and gesture velocities are not incorporated, which may limit the representational power of the features. However, mapping motion information on multiple orders at the same time can be tricky because it can sabotage the representational power of the features and the KNN mapping. The usage of higher-order features will be experimented in the future.

5.9 Conclusion and Discussions

In this work, we propose a data-driven approach for crowd control using a multi-touch device. Our method learns from a set of user-performed gestures and allows a user to intuitively control a crowd. To achieve this, we propose a set of gesture features that represent high-level information of the user-performed gestures. We also propose a method to extract crowd motion features using a mixture of Gaussian processes. Given a run-time gesture, we perform a KNN search in our gesture database and find the K corresponding crowd motions. We then combine the K crowd motions to control the run-time crowd. Our system runs in real-time and has high control accuracy.

Like many existing systems, the simulation time increases with the number of characters. However, our system is relatively computationally efficient with a large number of characters. This is because the majority part of the system is based on the extracted motion features and gesture features, which are independent of the number of characters. The only step that is computationally proportional to the character number is the synthesis of the final crowd motions.

A run-time crowd motion is generated by interpolating multiple instances in the database. Theoretically speaking, given the right gesture, it is possible to interpolate two classes of crowd motion (e.g. *translate* and *join*) to generate a new run-time motion. However, it rarely happens in practice due to the relatively wide range of gestures we collected to cover the possible variation within the same class. As a result, the interpolation performed is mostly intra-class.

Theoretically, the mapping could be contaminated if the gesture-motion pairs are not generated well, such as two similar gestures generating very different motions or vice versa. In practice, we find that KNN helps to reduce the effect of outlier mappings, as multiple motions/gestures are combined, and less similar ones are given smaller weights. Also, the mapping in our database is very descriptive thanks to the distinctiveness among the types of basic motions, which results in a set of distinctive corresponding gestures. More complex motions can be decomposed into the combinations of basic ones to avoid over-complicated motion-gesture mappings.

Chapter 6

Conclusions and Discussions

In the present thesis, the studies on modelling the human motions in motion analysis, motion retrieval/recognition and crowd motion control, namely the hot topics in the field of computer graphics, are introduced. Human motion modelling is an essential part of the advanced approaches of these topics. Due to the lack of information in underlying human motion data, achieving such tasks is still a big challenge. In the research here, the latent information behind human motion was found and modelled meaningfully. Besides, the motion analysis, motion retrieval and crowd motion control systems were designed on the top of the proposed motion representations. The proposed systems, capable of analyzing and retrieving human motion in semantic level in Chapter 3 and 4), and intuitively controlling crowd motion in Chapter 5, are demonstrated. According to experimental results in each chapter, the research here also can be summarized as *"handling human motions from single person movement to multi-people interaction, then to the crowd movement"* in computer graphics. The summary of contributions in this thesis is presented in the following section Section 6.1.

6.1 Summary of Contributions

This section summarizes the research contributions of this thesis. According to the contributions, the significance of human motion modelling to different tasks of computer graphics research is demonstrated.

- In Chapter 3, an automatic analysis and visualization system is proposed to assessing high-level skills of sports motions on a single person. In our experiment, the boxing motion was only tested, whereas our method is generic and can be applied to different sports (e.g. basketball and fencing). In sports training sessions, we found that human experts assess the athletes skills based on such features as the diversity of actions, transitions flexibility and unpredictability of action pattern. Based on the above observation, we first proposed two types of motion graphs on the athletes motion capture data. The Posture-based Graph is a variation of *Fat Graphs* [160] and it indicates the quality of static postures for initiating and completing actions. The Action-based Graph is formulated by *Hidden Markov Model* and it indicates the flexibility of transitions between different actions. We further analyzed topology structures of proposed graphs and raised two numerical assessment mechanisms which are Connectivity Index and Action Strategy Index. The Connectivity Index is calculated from Posture-based Graph, suggesting the diversity of actions and the flexibility of transitions between different actions. The Action Strategy Index is calculated from Action-based Graph and it reveals unpredictability of action patterns. With the graph representations of human motion and suitable assessment mechanism, common human movements can be assessed on the level of preference, intention and diversity.

- In Chapter 4, we proposed a unified framework for motion retrieval and analysis on the multi-people interaction movements. , Again, in our experiment, the results of two-character boxing motions were only demonstrated. However, our framework is easy to generalize into common types of interactions, e.g. "a person is sitting on the chair" (human-object interaction) and " two people are shaking hands" (daily social behavior).Based on the previous research in human motions and scenes understanding [9, 10, 61, 78, 161], the spatial relationship between interacting humans or objects is critical to specify the semantic meaning of the motions. A customized *Interaction Mesh* was proposed upon the interacting characters during the motion. This mesh structure encodes the information of each characters motion and spatial relationships among

interacting characters. The difference between interactions was assessed by proposing Earth Movers Distance onto the corresponding meshes. The intrinsic similarity among different classes of interactions can be easily captured by our framework. Furthermore, interaction-based motion retrieval and analysis mechanisms were designed on the top of the proposed framework, which improves the results in content-based retrieval and semantic-level motion analysis system. According to the experimental results, motion retrieval and analysis by our framework are aligned with human understanding of motion semantics.

- In Chapter 5, a data-driven approach is proposed for crowd motion control via multi-touch device. Basically, our system takes users gesture as input and infers an appropriate output crowd motion. The underlying control scheme is directly learned from the mapping between user preference gestures and corresponding crowd motions, which is not pre-defined by developers. As a result, our system achieves intuitive crowd motion control which does not require users to learn the control scheme before controlling the crowd motion. Specifically, we proposed a set of gesture features which represents the intrinsic properties of different types of users input. These features are invariant to the difference of the users preferred touch input style e.g. the number of fingers used. Subsequently, a set of crowd motion features extracted from crowd motion was proposed using FPCA and GMM. These features allow the modelling of the crowd into different sub-groups and are irrelevant to the number of agents inside the crowd. We then build up a two-layer database which includes the collected gestures from different users and corresponding crowd movements. During run-time, when a user inputs an arbitrary gesture, our system will extract the gesture features and conduct a KNN search in the gesture feature database. Finally, the corresponding crowd motions in the database are interpolated to generate new crowd movement that is approximately aligned with user input. According to the experimental results, our system is accurate and efficient enough to achieve real-time crowd motion control. It can be used for in real-time applications, e.g. video games and interactive animation controls.

6.2 Discussions and Future Works

This section discusses some potential research directions for our subsequent works. The following three subsections outline our subsequent works on Chapter 3, 4 and 5 respectively.

6.2.1 Extend Motion Graph in Computer Animation

In Chapter 3, we proposed graph representations on the sports motions to extract high-level information of sports skills. However, such representations have a wider range of potential in computer animation.

In the future, we wish to extend the proposed algorithm to the field of computer animation. Currently, when synthesizing animations by motion graphs, experienced animators are required to tell what motions are missed or badly captured. With our system, it is possible to analyze the connectivity and variety of a motion set, which are two critical factors in motion synthesis. However, how to generalize these findings to give the high-level suggestion, such as proposing the motions to capture, remains an open problem.

In addition, Combining with semantic information extracted from the framework in Section 4.5.2, we would like to develop a visualization system to take the adversarial nature of sports. For instance, although two boxers might have roughly the same skill level, in a match, one's skill composition might give him/her advantages over the other. This kind of analysis would be very useful in preparation for a game or predicting the result.

6.2.2 Generalization on Interaction Analysis

In Chapter 4, a 3D volumetric mesh representation of two characters interaction is proposed, and a customized Earth Movers Distance is put forward to assess the topological and geometric difference between two meshes. The above assessment captures the semantic similarities among different interactions that are closely aligned with human perception. Such a framework can be used in other areas of research, e.g. object retrieval and scene understanding

One future direction is to apply our system to assess human-environment interaction. The challenge is to design an algorithm to sample feature points on the environment to generate the interaction graph. Unlike the human body, the environment does not have a fixed topology, and therefore points sampled may be different depending on the geometric features. Our method does not need to uniquely identify the sampled points, which is opposed to previous work [62], making it suitable for the points sampled on the environment.

Another future direction is to adapt our method for modelling and evaluating interactions to the 3D object retrieval research. Retrieving 3D objects is also an active research in computer graphics, which it has broad applications in industries. The major difficulty is that there are a large topological and geometric variations in intra-class 3D objects [162–164]. For example, in our daily life, we have seen a lot of chairs with different supporting topology structures such as five wheels or four legs. Retrieving such objects via topological and geometric analysis raises a large false positive and false negative. Inspired by previous works on [10, 165], we found that the semantic of the 3D object is also implied by an interaction between human and object and be irrelevant to local geometric variations. For example, a chair can be represented by "a person is sitting it" and a desk can be represented by "a person is writing on it". The proposed framework can be used to extract such interaction information from the human-object scenario and improve the retrieval results.

Similar to 3D object retrieval, understanding 3D scenes is also a popular research direction in the field of computer graphics and computer vision. According to the previous research on [140,166–168], human-object and object-object interactions information are introduced into a complex 3D environment to achieve accurate and efficient 3D scenes recognition. For instance, in the scene of a parlour, the interaction information is represented by spatial relationship between different types of furniture, e.g. "tea table is placed next to the sofa, and TV is in front of the sofa", and between human and furniture, e.g. "a person is sitting on the sofa and facing the TV". Based on such spatial information extracted by our framework, 3D objects in the complex scene can be accurately identified and labelled.

6.2.3 Further Improvement on Crowd Motion Control

In Chapter 5,two representative features on users gesture and crowd motions are proposed. These features help our system quickly and accurately generate appropriate crowd motions from arbitrary users input gestures. This framework can be used for real-time control in video games. However, there is still a space for improvement.

Our crowd control system analyzes the gesture in a discrete manner. Each gesture controls the crowd in a short time interval. One possible solution is to apply the continuous recognition algorithms proposed in [67], in which the input gesture continues to be recognized using a variable size sliding window.

An interesting research direction is to introduce more intra-class differences in the crowd motion. For instance, a spread-out *translate* crowd motions and a condensed one can be generated. Subsequently, the corresponding gesture inputs can be collected from the user into the database. As a result, a small gesture difference will generate a small variation of the crowd motion, which helps to achieve the fine control of the crowd.

Another interesting direction is to embed the dense interactions among agents in the crowd motion. Previous research proposed to model the interaction among different characters as spatial-temporal constraints and to synthesize such interactions by the space-time optimization [169]. However, the computational cost for such optimization increases exponentially in the crowd motion, and it is difficult to achieve real-time control. Combining with our works in Chapter 4, the optimization process is expected to be simplified, and the crowd motion with detailed interactions is expected to be controlled in real time among agents of the crowd.

Bibliography

[1] Thomas B. Moeslund and Erik Granum. A survey of computer vision-based human motion capture. *Computer Vision and Image Understanding*, 81(3):231 – 268, 2001.

[2] A. Hilton, D. Beresford, T. Gentils, R. Smith, and Wei Sun. Virtual people: capturing human models to populate virtual worlds. In *Proceedings Computer Animation 1999*, pages 174–185, 1999.

[3] N. Badler. Virtual humans for animation, ergonomics, and simulation. In *Proceedings IEEE Nonrigid and Articulated Motion Workshop*, pages 28–36, Jun 1997.

[4] Huiyu Zhou and Huosheng Hu. Human motion tracking for rehabilitationa survey. *Biomedical Signal Processing and Control*, 3(1):1 – 18, 2008.

[5] Meinard Muller, Tido Roder, Michael Clausen, Bernhard Eberhardt, Bjrn Krger, and Andreas Weber. Documentation mocap database hdm05, 2007.

[6] S.Maddock M.Meredith. Motion capture file formats explained. `http://staffwww.dcs.shef.ac.uk/people/S.Maddock/publications/Motion%20Capture%20File%20Formats%20Explained.pdf`. Accessed January 10, 2001.

[7] Jianyuan Min and Jinxiang Chai. Motion graphs++: a compact generative model for semantic motion analysis and synthesis. *ACM Transactions on Graphics*, 31(6):153, 2012.

[8] Edmond S. L. Ho, Taku Komura, and Chiew-Lan Tai. Spatial relationship preserving character motion adaptation. *ACM Trans. Graph.*, 29(4):33:1–33:8, July 2010.

[9] Xi Zhao, He Wang, and Taku Komura. Indexing 3d scenes using the interaction bisector surface. *ACM Trans. Graph.*, 33(3):22:1–22:14, June 2014.

[10] Ruizhen Hu, Chenyang Zhu, Oliver van Kaick, Ligang Liu, Ariel Shamir, and Hao Zhang. Interaction context (icon): Towards a geometric functionality descriptor. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 34(4):83:1–83:12, 2015.

[11] Eunjung Ju, Myung Geol Choi, Minji Park, Jehee Lee, Kang Hoon Lee, and Shigeo Takahashi. Morphable crowds. *ACM Transactions on Graphics (TOG)*, 29(6):140, 2010.

[12] Jongmin Kim, Yeongho Seol, Taesoo Kwon, and Jehee Lee. Interactive manipulation of large-scale crowd animation. *ACM Transactions on Graphics*, 33(4):1–10, Jul 2014.

[13] A. Fern'ndez-Baena, A. Susn, and X. Lligadas. Biomechanical validation of upper-body and lower-body joint movements of kinect motion capture data for rehabilitation treatments. In *2012 Fourth International Conference on Intelligent Networking and Collaborative Systems*, pages 656–661, Sept 2012.

[14] Brook Galna, Gillian Barry, Dan Jackson, Dadirayi Mhiripiri, Patrick Olivier, and Lynn Rochester. Accuracy of the microsoft kinect sensor for measuring movement in people with parkinson's disease. *Gait & Posture*, 39(4):1062 – 1068, 2014.

[15] Hans Kainz, Martin Hajek, Luca Modenese, David J. Saxby, David G. Lloyd, and Christopher P. Carty. Reliability of functional and predictive methods to estimate the hip joint centre in human motion analysis in healthy adults. *Gait & Posture*, 53:179 – 184, 2017.

[16] André Ebert, Marie Kiermeier, Chadly Marouane, and Claudia Linnhoff-Popien. Sensx: About sensing and assessment of complex human motion. *CoRR*, abs/1703.02847, 2017.

[17] Michael Chin. *Clinical Use of Gait Analysis for the Athlete*, pages 55–65. Springer International Publishing, Cham, 2017.

[18] Georgios Mastorakis and Dimitrios Makris. Fall detection system using kinect's infrared sensor. *Journal of Real-Time Image Processing*, 9(4):635–646, Dec 2014.

[19] E. E. Stone and M. Skubic. Fall detection in homes of older adults using the microsoft kinect. *IEEE Journal of Biomedical and Health Informatics*, 19(1):290–301, Jan 2015.

[20] M. Yeadon. The simulation of aerial movement- iv. a computer simulation model. *Journal of Biomechanics*, 23(1):85–89, 1990.

[21] M. Yeadon. The biomechanics of the human in flight. *The American Journal of Sports Medicine*, 25(4):575–580, 1997.

[22] Simon R. Goodman and Gerald L. Gottlieb. Analysis of kinematic invariances of multijoint reaching movement. *Biological Cybernetics*, 73:313–322, 1995.

[23] R.R. Neptune. Optimization algorithm performance in determining optimal controls in human movement analysis. *Journal of Biomedical Engineering*, 124(2):249–252, 1999.

[24] Gary Yamaguchi. *Dynamic Modeling of Musculoskeletal Motion - A Vectorized Approach for Biomechanical Analysis in Three Dimensions*. Kluwer Academic Publishers, 2001.

[25] MARIEL VAZQUEZ and DE WITT SUMNERS. Tangle analysis of gin site-specific recombination. *Mathematical Proceedings of the Cambridge Philosophical Society*, 136:565–582, 2004.

[26] Okan Arikan and D.A. Forsyth. Motion generation from examples. *ACM Transactions on Graphics*, 21(3):483–490, 2002.

[27] Jehee Lee, Jinxiang Chai, Paul S. A. Reitsma, Jessica K. Hodgins, and Nancy S. Pollard. Interactive control of avatars animated with human motion data. *ACM Transactions on Graphics*, 21(3):491–500, 2002.

[28] Lucas Kovar, Michael Gleicher, and Fre'de'ric Pighin. Motion graphs. *ACM Transactions on Graphics*, 21(3):473–482, 2002.

[29] Keith Grochow, Steven L. Martin, Aaron Hertzmann, and Zoran Popovic'. Style-based inverse kinematics. *ACM Transactions on Graphics (TOG)*, 22(3), August 2004.

[30] Hyun Joon Shin and Jehee Lee. Motion synthesis and editing in low-dimensional spaces. *Computer Animation and Virtual Worlds (Special Issue: CASA 2006)*, 17(3-4):219 – 227, 2006.

[31] Philippe Beaudoin, Stelian Coros, Michiel van de Panne, and Pierre Poulin. Motion-motif graphs. In *Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '08, pages 117–126, Aire-la-Ville, Switzerland, Switzerland, 2008. Eurographics Association.

[32] Okan Arikan, David A. Forsyth, and James F. O'Brien. Motion synthesis from annotations. *ACM Trans. Graph.*, 22(3):402–408, July 2003.

[33] Yan Li, Tianshu Wang, and Heung-Yeung Shum. Motion texture: A two-level statistical model for character motion synthesis. *ACM Trans. Graph.*, 21(3):465–472, July 2002.

[34] Paul S. A. Reitsma and Nancy S. Pollard. Evaluating motion graphs for character animation. *ACM Trans. Graph.*, 26(4), October 2007.

[35] Rachel Heck and Michael Gleicher. Parametric motion graphs. In *Proceedings of the 2007 Symposium on Interactive 3D Graphics and Games*, I3D '07, pages 129–136, New York, NY, USA, 2007. ACM.

[36] Shihong Xia, Congyi Wang, Jinxiang Chai, and Jessica Hodgins. Realtime style transfer for unlabeled heterogeneous human motion. *ACM Trans. Graph.*, 34(4):119:1–119:10, July 2015.

[37] Alla Safonova and Jessica K. Hodgins. Construction and optimal search of interpolated motion graphs. *ACM Trans. Graph.*, 26(3), July 2007.

[38] Hyun Joon Shin and Hyun Seok Oh. Fat graphs: Constructing an interactive character with continuous controls. *ACM SIGGRAPH / Eurographics Symposium on Computer Animation*, pages 291–298, 2006.

[39] Hubert P. H. Shum, Taku Komura, Masashi Shiraishi, and Shuntaro Yamazaki. Interaction patches for multi-character animation. *ACM Trans. Graph.*, 27(5):114:1–114:8, Dec 2008.

[40] Hubert P. H. Shum, Taku Komura, and Shuntaro Yamazaki. Simulating multiple character interactions with collaborative and adversarial goals. *IEEE Transactions on Visualization and Computer Graphics*, 18(5):741–752, May 2012.

[41] Kyunglyul Hyun, Kyungho Lee, and Jehee Lee. Motion grammars for character animation. *Computer Graphics Forum*, 35(2):103–113, 2016.

[42] Edmond S. L. Ho and Taku Komura. A finite state machine based on topology coordinates for wrestling games. *Computer Animation and Virtual Worlds*, 22(5):435–443, 2011.

[43] Edmond S.L. Ho and Taku Komura. Character Motion Synthesis by Topology Coordinates. *Computer Graphics Forum*, 2009.

[44] Pierre Plantard, Hubert P. H. Shum, and Franck Multon. Filtered pose graph for efficient kinect pose reconstruction. *Journal of Multimedia Tools and Applications*, 76(3):4291–4312, 2017.

[45] Pierre Plantard, Hubert P. H. Shum, Anne-Sophie Le Pierres, and Franck Multon. Validation of an ergonomic assessment method using kinect data in real workplace conditions. *Applied Ergonomics*, 2016.

[46] Hubert P. H. Shum, Taku Komura, Takaaki Shiratori, and Shu Takagi. Physically-based character control in low dimensional space. In *Proceedings of the Third International Conference on Motion in Games*, volume 6459 of *MIG '10*, pages 23–34, Berlin, Heidelberg, Nov 2010. Springer-Verlag.

[47] Hubert P. H. Shum, Edmond S. L. Ho, Yang Jiang, and Shu Takagi. Real-time posture reconstruction for microsoft kinect. *IEEE Transactions on Cybernetics*, 43(5):1357–1369, 2013.

[48] Edmond S. L. Ho, Hubert P. H. Shum, Yiu-ming Cheung, and P. C. Yuen. Topology aware data-driven inverse kinematics. *Comp. Graph. Forum*, 32(7):61–70, Oct 2013.

[49] Neil D Lawrence. Gaussian process latent variable models for visualisation of high dimensional data. In *Proceedings of the 2003 Neural Information Processing Systems*, 2003.

[50] J. M. Wang, D. J. Fleet, and A. Hertzmann. Gaussian process dynamical models for human motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(2):283–298, Feb 2008.

[51] Daniel Holden, Jun Saito, and Taku Komura. A deep learning framework for character motion synthesis and editing. *ACM Trans. Graph.*, 35(4):138:1–138:11, July 2016.

[52] Zhiguang Liu, Liuyang Zhou, Howard Leung, and Hubert P. H. Shum. Kinect posture reconstruction based on a local mixture of gaussian process models. *IEEE Transactions on Visualization and Computer Graphics*, 2016.

[53] L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, Feb 1989.

[54] K. Hara, T. Omori, and R. Ueno. Detection of unusual human behavior in intelligent house. In *Proceedings of the 12th IEEE Workshop on Neural Networks for Signal Processing*, pages 697–706, 2002.

[55] Jules Françoise, Agnès Roby-Brami, Natasha Riboud, and Frédéric Bevilacqua. Movement sequence analysis using hidden markov models: A case study in tai chi performance. In *Proceedings of the 2Nd International Workshop on Movement and Computing*, MOCO '15, pages 29–36, New York, NY, USA, 2015. ACM.

[56] Matthew Brand and Aaron Hertzmann. Style machines. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, pages 183–192, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[57] L. M. Tanco and A. Hilton. Realistic synthesis of novel human movements from a database of motion capture examples. In *Proceedings Workshop on Human Motion*, pages 137–142, 2000.

[58] Liu Ren, Alton Patrick, Alexei A. Efros, Jessica K. Hodgins, and James M. Rehg. A data-driven approach to quantifying natural human motion. *ACM Trans. Graph.*, 24(3):1090–1097, July 2005.

[59] Hubert P. H. Shum, He Wang, Edmond S. L. Ho, and Taku Komura. Skillvis: A visualization tool for boxing skill assessment. In *Proceedings of the 2016 International Conference on Motion in Games*, MIG '16, pages 145–153, New York, NY, USA, Oct 2016. ACM.

[60] Taku Komura, Hubert P. H. Shum, and Edmond S. L. Ho. Simulating interactions of characters. In *Motion in Games*, Lecture Notes in Computer Science, pages 94–103. Springer, 2008.

[61] Meinard Muller, Tido Roder, and Michael Clausen. Efficient content-based retrieval of motion capture data. *ACM Trans. Graph.*, 24(3):677–685, 2005.

[62] Jeff K.T. Tang, Jacky C.P. Chan, Howard Leung, and Taku Komura. Interaction retrieval by spacetime proximity graphs. *Comp. Graph. Forum*, 31(2pt2):745–754, May 2012.

[63] Edmond S. L. Ho, Jacky C. P. Chan, Yiu-ming Cheung, and Pong C. Yuen. Modeling spatial relations of human body parts for indexing and retrieving close character interactions. In *Proceedings of the 21st ACM Symposium on Virtual Reality Software and Technology*, VRST '15, pages 187–190, New York, NY, USA, 2015. ACM.

[64] Edmond S. L. Ho, Jacky C. P. Chan, Yiu-ming Cheung, and Pong C. Yuen. Modeling spatial relations of human body parts for indexing and retrieving close character interactions. In *Proceedings of the 21st ACM Symposium on Virtual Reality Software and Technology*, VRST '15, pages 187–190, New York, NY, USA, 2015. ACM.

[65] Jehee Lee, Jinxiang Chai, Paul S. A. Reitsma, Jessica K. Hodgins, and Nancy S. Pollard. Interactive control of avatars animated with human motion data. *ACM Trans. Graph.*, 21(3):491–500, 2002.

[66] Lucas Kovar and Michael Gleicher. Automated extraction and parameterization of motions in large data sets. *ACM Trans. Graph.*, 23(3):559–568, 2004.

[67] Hubert P. H. Shum, Taku Komura, and Shu Takagi. Fast accelerometer-based motion recognition with a dual buffer framework. *The International Journal of Virtual Reality*, 10(3):17–24, Sep 2011.

[68] E. S. L. Ho and H. P. H. Shum. Motion adaptation for humanoid robots in constrained environments. In *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, pages 3813–3818, May 2013.

[69] Meinard Müller, Andreas Baak, and Hans-Peter Seidel. Efficient and robust annotation of motion capture data. In *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '09, pages 17–26, New York, NY, USA, 2009. ACM.

[70] Liu Ren, Alton Patrick, Alexei A. Efros, Jessica K. Hodgins, and James M. Rehg. A data-driven approach to quantifying natural human motion. *ACM Trans. Graph.*, 24(3):1090–1097, July 2005.

[71] Mubbasir Kapadia, I-kao Chiang, Tiju Thomas, Norman I. Badler, and Joseph T. Kider, Jr. Efficient motion retrieval in large motion databases. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, I3D '13, pages 19–28, New York, NY, USA, 2013. ACM.

[72] Jeff K. T. Tang, Howard Leung, Taku Komura, and Hubert P. H. Shum. Emulating human perception of motion similarity. *Comput. Animat. Virtual Worlds*, 19(3–4):211–221, 2008.

[73] R. Vemulapalli, F. Arrate, and R. Chellappa. Human action recognition by representing 3d skeletons as points in a lie group. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 588–595, June 2014.

[74] C. Chen, Y. Zhuang, F. Nie, Y. Yang, F. Wu, and J. Xiao. Learning a 3d human pose distance metric from geometric pose descriptor. *IEEE Transactions on Visualization and Computer Graphics*, 17(11):1676–1689, Nov 2011.

[75] Daniel Holden, Jun Saito, and Taku Komura. A deep learning framework for character motion synthesis and editing. *ACM Transactions on Graphics*, 35(4), 2016.

[76] Kiwon Yun, Jean Honorio, Debaleena Chattopadhyay, Tamara L. Berg, and Dimitris Samaras. Two-person interaction detection using body-pose features and multiple instance learning. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on*. IEEE, 2012.

[77] Edmond S. L. Ho and Taku Komura. Character motion synthesis by topology coordinates. *Computer Graphics Forum*, 28(2):299–308, 2009.

[78] Edmond S. L. Ho and Taku Komura. Indexing and retrieving motions of characters in close contact. *IEEE Transactions on Visualization and Computer Graphics*, 15(3):481–492, 2009.

[79] Hang Si and Klaus Grtner. Meshing piecewise linear complexes by constrained delaunay tetrahedralizations. In *In Proceedings of the 14th International Meshing Roundtable*, pages 147–163. Springer, 2005.

[80] Rami Ali Al-Asqhar, Taku Komura, and Myung Geol Choi. Relationship descriptors for interactive motion adaptation. In *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '13, pages 45–53, New York, NY, USA, 2013. ACM.

[81] Vladimir Ivan, Dmitry Zarubin, Marc Toussaint, Taku Komura, and Sethu Vijayakumar. Topology-based representations for motion planning and generalization in dynamic environments with interactions. *Int. J. Rob. Res.*, 32(9-10):1151–1163, August 2013.

[82] Yossi Rubner, Carlo Tomasi, and Leonidas J. Guibas. A metric for distributions with applications to image databases. In *Proceedings of the Sixth International Conference on Computer Vision*, pages 59–, Washington, DC, USA, 1998.

[83] Dirk Helbing, Illes Farkas, and Tamas Vicsek. Simulating dynamical features of escape panic. *Nature*, 407(6803):487–490, 2000.

[84] Hengchin Yeh, Sean Curtis, Sachin Patil, Jur van den Berg, Dinesh Manocha, and Ming Lin. Composite agents. In *Proceedings of the 2008 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 39–47. Eurographics Association, 2008.

[85] Jan Ondřej, Julien Pettré, Anne-Hélène Olivier, and Stéphane Donikian. A synthetic-vision based steering approach for crowd simulation. *ACM Transactions on Graphics (TOG)*, 29(4):123, 2010.

[86] Rahul Narain, Abhinav Golas, Sean Curtis, and Ming C Lin. Aggregate dynamics for dense crowd simulation. In *ACM Transactions on Graphics (TOG)*, volume 28, page 122. ACM, 2009.

[87] Mankyu Sung, Michael Gleicher, and Stephen Chenney. Scalable behaviors for crowd simulation. In *Computer Graphics Forum*, volume 23, pages 519–528. Wiley Online Library, 2004.

[88] Adrien Treuille, Seth Cooper, and Zoran Popović. Continuum crowds. In *ACM Transactions on Graphics (TOG)*, volume 25, pages 1160–1168. ACM, 2006.

[89] Kang Hoon Lee, Myung Geol Choi, Qyoun Hong, and Jehee Lee. Group behavior from video: a data-driven approach to crowd simulation. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 109–118. Eurographics Association, 2007.

[90] Alon Lerner, Eitan Fitusi, Yiorgos Chrysanthou, and Daniel Cohen-Or. Fitting behaviors to pedestrian simulations. In *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 199–208. ACM, 2009.

[91] Yingying Jiang, Feng Tian, Xiaolong Zhang, Wei Liu, Guozhong Dai, and Hongan Wang. Unistroke gestures on multi-touch interaction: Supporting flexible touches with key stroke extraction. In *Proceedings of the 2012 ACM International Conference on Intelligent User Interfaces*, IUI '12, pages 85–88, New York, NY, USA, 2012. ACM.

[92] Radu-Daniel Vatavu, Lisa Anthony, and Jacob O. Wobbrock. Gestures as point clouds: A $p recognizer for user interface prototypes. In *Proceedings of the 14th ACM International Conference on Multimodal Interaction*, ICMI '12, pages 273–280, New York, NY, USA, 2012. ACM.

[93] Yosra Rekik, Radu-Daniel Vatavu, and Laurent Grisoni. Match-up & conquer. *Proceedings of the 2014 International Working Conference on Advanced Visual Interfaces - AVI '14*, pages 201–208, 2014.

[94] Martin Thomas Görg, Michael Cebulla, and Sandro Rodriguez Garzon. A framework for abstract representation and recognition of gestures in multi-

touch applications. In *Advances in Computer-Human Interactions, 2010. ACHI'10. Third International Conference on*, pages 143–147. IEEE, 2010.

[95] Dafydd Gibbon, Ulrike Gut, Benjamin Hell, Karin Looks, Alexandra Thies, and Thorsten Trippel. A computational model of arm gestures in conversation. In *INTERSPEECH*, 2003.

[96] Dietrich Kammer, Jan Wojdziak, Mandy Keck, Rainer Groh, and Severin Taranko. Towards a formalization of multi-touch gestures. In *ACM International Conference on Interactive Tabletops and Surfaces*, ITS '10, pages 49–58, New York, NY, USA, 2010. ACM.

[97] Hao Lü and Yang Li. Gesture coder: a tool for programming multi-touch gestures by demonstration. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 2875–2884. ACM, 2012.

[98] Kenrick Kin, Björn Hartmann, Tony DeRose, and Maneesh Agrawala. Proton++: A customizable declarative multitouch framework. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology*, UIST '12, pages 477–486, New York, NY, USA, 2012. ACM.

[99] Hao Lü and Yang Li. Gesture studio: authoring multi-touch interactions through demonstration and declaration. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pages 257–266. ACM, 2013.

[100] Min Je Park. Guiding flows for controlling crowds. *The Visual Computer*, 26(11):1383–1391, 2010.

[101] Qin Gu and Zhigang Deng. Formation sketching: an approach to stylize groups in crowd simulation. In *Proceedings of Graphics Interface 2011*, pages 1–8. Canadian Human-Computer Communications Society, 2011.

[102] Masaki Oshita and Yusuke Ogiwara. Sketch-based interface for crowd animation. In *Smart Graphics*, pages 253–262. Springer, 2009.

[103] Manmyung Kim, Kyunglyul Hyun, Jongmin Kim, and Jehee Lee. Synchronized multi-character motion editing. In *ACM Transactions on Graphics (TOG)*, volume 28, page 79. ACM, 2009.

[104] Taesoo Kwon, Kang Hoon Lee, Jehee Lee, and Shigeo Takahashi. Group motion editing. In *ACM Transactions on Graphics (TOG)*, volume 27, page 80. ACM, 2008.

[105] Jun Kato, Daisuke Sakamoto, Masahiko Inami, and Takeo Igarashi. Multi-touch interface for controlling multiple mobile robots. In *CHI'09 Extended Abstracts on Human Factors in Computing Systems*, pages 3443–3448. ACM, 2009.

[106] Xiaogang Jin, Jiayi Xu, Charlie CL Wang, Shengsheng Huang, and Jun Zhang. Interactive control of large-crowd navigation in virtual environments using vector fields. *Computer Graphics and Applications, IEEE*, 28(6):37–46, 2008.

[107] Sachin Patil, Jur Van Den Berg, Sean Curtis, Ming C Lin, and Dinesh Manocha. Directing crowd simulations using navigation fields. *IEEE Transactions on Visualization and Computer Graphics*, 17(2):244–254, 2011.

[108] Joseph Henry, Hubert P. H. Shum, and Taku Komura. Environment-aware real-time crowd control. In *Proceedings of the 11th ACM SIGGRAPH / Eurographics conference on Computer Animation*, EUROSCA'12, pages 193–200, Aire-la-Ville, Switzerland, Switzerland, 2012. Eurographics Association.

[109] Joseph Henry, Hubert P. H. Shum, and Taku Komura. Interactive formation control in complex environments. *IEEE transactions on visualization and computer graphics*, 20(2):211–222, 2014.

[110] Sybren A. Stuvel, Nadia Magnenat-Thalmann, Daniel Thalmann, A. Frank van der Stappen, Arjan Egges, undefined, undefined, undefined, and undefined. Torso crowds. *IEEE Transactions on Visualization and Computer Graphics*, 23(7):1823–1837, 2017.

[111] Olga Sorkine, Daniel Cohen-Or, Yaron Lipman, Marc Alexa, Christian Rössl, and H-P Seidel. Laplacian surface editing. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pages 175–184. ACM, 2004.

[112] Shigeo Takahashi, Kenichi Yoshida, Taesoo Kwon, Kang Hoon Lee, Jehee Lee, and Sung Yong Shin. Spectral-based group formation control. In *Computer Graphics Forum*, volume 28, pages 639–648. Wiley Online Library, 2009.

[113] Ioannis Karamouzas and Mark Overmars. Simulating the local behaviour of small pedestrian groups. In *Proceedings of the 17th ACM Symposium on Virtual Reality Software and Technology*, pages 183–190. ACM, 2010.

[114] Liping Zheng, Jianming Zhao, Yajun Cheng, Haibo Chen, Xiaoping Liu, and Wenping Wang. Geometry-constrained crowd formation animation. *Computers & Graphics*, 38:268–276, 2014.

[115] Qin Gu and Zhigang Deng. Generating freestyle group formations in agent-based crowd simulations. *IEEE Computer Graphics and Applications*, 33(1):20–31, 2013.

[116] Mingliang Xu, Yunpeng Wu, Yangdong Ye, Illes Farkas, Hao Jiang, and Zhigang Deng. Collective crowd formation transform with mutual information–based runtime feedback. In *Computer Graphics Forum*, volume 34, pages 60–73. Wiley Online Library, 2015.

[117] He Wang, Jan Ondřej, and Carol O'Sullivan. Path patterns: Analyzing and comparing real and simulated crowds. In *ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games 2016*, pages 49–57. ACM, 2016.

[118] Stephen W. Bailey and Bobby Bodenheimer. A comparison of motion capture data recorded from a vicon system and a microsoft kinect sensor. In *Proceedings of the ACM Symposium on Applied Perception*, SAP '12, pages 121–121, New York, NY, USA, 2012. ACM.

[119] A. Fern'ndez-Baena, A. Susn, and X. Lligadas. Biomechanical validation of upper-body and lower-body joint movements of kinect motion capture data for rehabilitation treatments. In *Intelligent Networking and Collaborative Systems (INCoS), 2012 4th International Conference on*, pages 656–661, Sept 2012.

[120] Cmu graphics lab motion capture database. `http://mocap.cs.cmu.edu/`.

[121] Hubert P. H. Shum and Edmond S. L. Ho. Real-time physical modelling of character movements with microsoft kinect. In *Proceedings of the 18th ACM Symposium on Virtual Reality Software and Technology*, VRST '12, pages 17–24, New York, NY, USA, Dec 2012. ACM.

[122] Jinxiang Chai and Jessica K. Hodgins. Performance animation from low-dimensional control signals. In *SIGGRAPH '05: ACM SIGGRAPH 2005 Papers*, pages 686–696, New York, NY, USA, 2005. ACM.

[123] Jochen Tautges, Arno Zinke, Björn Krüger, Jan Baumann, Andreas Weber, Thomas Helten, Meinard Müller, Hans-Peter Seidel, and Bernd Eberhardt. Motion reconstruction using sparse accelerometer data. *ACM Trans. Graph.*, 30(3):18:1–18:12, May 2011.

[124] Huajun Liu, Xiaolin Wei, Jinxiang Chai, Inwoo Ha, and Taehyun Rhee. Realtime human motion control with a small number of inertial sensors. In *Symposium on Interactive 3D Graphics and Games*, I3D '11, pages 133–140, New York, NY, USA, 2011. ACM.

[125] Liuyang Zhou, Zhiguang Liu, Howard Leung, and Hubert P. H. Shum. Posture reconstruction using kinect with a probabilistic model. In *Proceedings of the 20th ACM Symposium on Virtual Reality Software and Technology*, VRST '14, pages 117–125, New York, NY, USA, Nov 2014. ACM.

[126] Edmond S. L. Ho, Jacky C. P. Chan, Donald C. K. Chan, Hubert P. H. Shum, Yiu-ming Cheung, and P. C. Yuen. Improving posture classification accuracy for depth sensor-based human activity monitoring in smart environments. *Computer Vision and Image Understanding*, 2016.

[127] Daniel Holden, Jun Saito, Taku Komura, and Tom Joyce. Learning motion manifolds with convolutional autoencoders. In *ACM SIGGRAPH ASIA 2015 Technical Briefs*. ACM, 2015.

[128] Pierre Plantard, Hubert P. H. Shum, and Franck Multon. In *Proceedings of the 2016 International Digital Human Modeling Symposium*, DHM '16, Jun 2016.

[129] Yijun Shen, Joseph Henry, He Wang, Edmond S. L. Ho, Taku Komura, and Hubert P. H. Shum. Data-driven crowd motion control with multi-touch gestures. *Comp. Graph. Forum*, 2018.

[130] Yijun Shen, He Wang, Edmond S. L. Ho, Longzhi Yang, and Hubert P. H. Shum. Posture-based and action-based graphs for boxing skill visualization. *Computers and Graphics*, 69(Supplement C):104–115, 2017.

[131] Yijun Shen, Jingtian Zhang, Longzhi Yang, and Hubert P. H. Shum. Depth sensor based facial and body animation control. In *Handbook of Human Motion*. Springer International Publishing, Cham, 2016.

[132] Taku Komura, Atsushi Kuroda, and Yoshihisa Shinagawa. Nicemeetvr: Facing professional baseball pitchers in the virtual batting cage. In *Proceedings of the 2002 ACM Symposium on Applied Computing*, SAC '02, pages 1060–1065, New York, NY, USA, 2002. ACM.

[133] Benoit Bideau, Richard Kulpa, Stephane Menardais, Laetitia Fradet, Franck Multon, Paul Delamarche, and Bruno Arnaldi. Real handball goalkeeper vs. virtual handball thrower. *Presence: Teleoper. Virtual Environ.*, 12(4):411–421, 2003.

[134] Tom Molet, Amaury Aubel, Tolga Capin, Stphane Carion, Elwin Lee, Nadia Magnenat-Thalmann, Hansrudi Noser, Igor Pandzic, Gal Sannier, and Daniel Thalmann. Anyone for tennis? *Presence: Teleoperators and Virtual Environments*, 8(2):140–156, 1999.

[135] Kevin Wampler, Erik Andersen, Evan Herbst, Yongjoon Lee, and Zoran Popović. Character animation in two-player adversarial games. *ACM Trans. Graph.*, 29(3):26:1–26:13, July 2010.

[136] Julian Gold Thore Graepel, Ralf Herbrich. Learning to fight. In *Proceedings of the International Conference on Computer Games: Artificial Intelligence, Design and Educatio*, 2004.

[137] M. Li and H. Leung. Multiview skeletal interaction recognition using active joint interaction graph. *IEEE Transactions on Multimedia*, 18(11):2293–2302, Nov 2016.

[138] Edmond S. L. Ho, Jacky C. P. Chan, Taku Komura, and Howard Leung. Interactive partner control in close interactions for real-time applications. *ACM Trans. Multimedia Comput. Commun. Appl.*, 9(3):21:1–21:19, July 2013.

[139] Lucas Kovar and Michael Gleicher. Flexible automatic motion blending with registration curves. In *SCA '03: Proceedings of the 2003 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 214–224, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.

[140] Taku Komura, Myung-Geol Choi, and Xi Zhao. Character-object interaction retrieval using the interaction bisector surface. *Computer Graphics Forum*, 36(2):119–129, 5 2017.

[141] Hubert P. H. Shum, Taku Komura, and Shuntaro Yamazaki. Simulating competitive interactions using singly captured motions. In *Proceedings of the 2007 ACM symposium on Virtual Reality Software and Technology*, VRST '07, pages 65–72, New York, NY, USA, Nov 2007. ACM.

[142] Yosra Rekik, Laurent Grisoni, and Nicolas Roussel. Towards Many Gestures to One Command: A User Study for Tabletops. In *INTERACT - 14th IFIP TC13 Conference on Human-Computer Interaction*, Cape Town, South Africa, September 2013. Nelson Mandela Metropolitan University, CSIR Meraka Institute and the University of Cape Town, Springer.

[143] Mark Micire, Munjal Desai, Amanda Courtemanche, Katherine M Tsui, and Holly A Yanco. Analysis of natural gestures for controlling robot teams on multi-touch tabletop surfaces. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces*, pages 41–48. ACM, 2009.

[144] Yongjoon Lee, Kevin Wampler, Gilbert Bernstein, Jovan Popović, and Zoran Popović. Motion fields for interactive character locomotion. *Communications of the ACM*, 57(6):101–108, Jun 2014.

[145] Jacob O. Wobbrock, Andrew D. Wilson, and Yang Li. Gestures without libraries, toolkits or training: A $1 recognizer for user interface prototypes. In *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology*, UIST '07, pages 159–168, New York, NY, USA, 2007. ACM.

[146] Cristian Constantin Lalescu. Two hierarchies of spline interpolations. practical algorithms for multivariate higher order splines. *arXiv*, 2009.

[147] Godfried Toussaint. Solving geometric problems with the rotating calipers, 1983.

[148] Donald J. Berndt and James Clifford. Using dynamic time warping to find patterns in time series. In Usama M. Fayyad and Ramasamy Uthurusamy, editors, *KDD Workshop*, pages 359–370. AAAI Press, 1994.

[149] Christopher M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, Oxford, UK, 1996.

[150] Jianyuan Min, Yen-Lin Chen, and Jinxiang Chai. Interactive generation of human animation with deformable motion models. *ACM Transactions on Graphics*, 29(1):1–12, Dec 2009.

[151] Jianyuan Min and Jinxiang Chai. Motion graphs++. *ACM Transactions on Graphics*, 31(6):1, Nov 2012.

[152] Ward Cheney and David R. Kincaid. *Linear Algebra: Theory and Applications*. Jones and Bartlett Publishers, Inc., USA, 1st edition, 2008.

[153] Nicolas Bonneel, Michiel van de Panne, Sylvain Paris, and Wolfgang Heidrich. Displacement interpolation using lagrangian mass transport. *Proceedings of the 2011 SIGGRAPH Asia Conference on - SA '11*, 2011.

[154] Michiel van de Panne. Parameterized gait synthesis. *IEEE Computer Graphics and Application*, March:40–49, 1996.

[155] Fumihiko Asano and Masaki Yamakita. Virtual gravity and coupling control for robotic gait synthesis. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 31(6):737–745, 2001.

[156] Yan Li, Tianshu Wang, and Heung-Yeung Shum. Motion texture: A two-level statistical model for character motion synthesis. *ACM Transactions on Graphics*, 21(3):465–472, 2002.

[157] Katherine Pullen and Christoph Bregler. Motion capture assised animation: Texturing and synthesis. *ACM Transactions on Graphics*, 21(3):501–516, 2002.

[158] J. van den Berg, Ming Lin, and D. Manocha. Reciprocal velocity obstacles for real-time multi-agent navigation. In *2008 IEEE International Conference on Robotics and Automation*, pages 1928–1935, May 2008.

[159] Daniel Holden, Taku Komura, and Jun Saito. Phase-functioned neural networks for character control. *ACM Trans. Graph.*, 36(4):42:1–42:13, July 2017.

[160] Hyun Joon Shin and Hyun Seok Oh. Fat graphs: constructing an interactive character with continuous controls. In *SCA '06: Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 291–298, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association.

[161] Liangliang Nan, Ke Xie, and Andrei Sharf. A search-classify approach for cluttered indoor scene understanding. *ACM Trans. Graph.*, 31(6):137:1–137:10, November 2012.

[162] Meng Yu, I. Atmosukarto, Wee Kheng Leow, Zhiyong Huang, and Rong Xu. 3d model retrieval with morphing-based geometric and topological feature maps.

In *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, volume 2, pages II–656–61 vol.2, June 2003.

[163] Konstantinos Sfikas, Theoharis Theoharis, and Ioannis Pratikakis. Non-rigid 3d object retrieval using topological information guided by conformal factors. *The Visual Computer*, 28(9):943–955, Sep 2012.

[164] Ruizhen Hu, Oliver van Kaick, Youyi Zheng, and Manolis Savva. Siggraph asia 2016: Course notes directions in shape analysis towards functionality. In *SIGGRAPH ASIA 2016 Courses*, SA '16, pages 8:1–8:326, New York, NY, USA, 2016. ACM.

[165] Ruizhen Hu, Oliver van Kaick, Bojian Wu, Hui Huang, Ariel Shamir, and Hao Zhang. Learning how objects function via co-analysis of interactions. *ACM Transactions on Graphics (Proc. SIGGRAPH)*, 35(4):47:1–47:13, 2016.

[166] Abhinav Gupta, Scott Satkin, Alexei A. Efros, and Martial Hebert. From 3d scene geometry to human workspace. In *Computer Vision and Pattern Recognition(CVPR)*, 2011.

[167] Yun Jiang, Hema Koppula, and Ashutosh Saxena. Hallucinated humans as the hidden context for labeling 3d scenes. In *Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '13, pages 2993–3000, Washington, DC, USA, 2013. IEEE Computer Society.

[168] Xi Zhao, Ruizhen Hu, Paul Guerrero, Niloy Mitra, and Taku Komura. Relationship templates for creating scene variations. *ACM Trans. Graph.*, 35(6):207:1–207:13, November 2016.

[169] Manmyung Kim, Kyunglyul Hyun, Jongmin Kim, and Jehee Lee. Synchronized multi-character motion editing. In *ACM SIGGRAPH 2009 Papers*, SIGGRAPH '09, pages 79:1–79:9, New York, NY, USA, 2009. ACM.