Automatic Extraction of Discriminative Semantic Features for Action Recognition

by

Tran Thang Thanh

Submitted to
Japan Advanced Institute of Science and Technology
in partial fulfillment of the requirements
for the degree of
Doctor of Philosophy

Supervisor: Associate Professor Kazunori Kotani

School of Information Science
Japan Advanced Institute of Science and Technology

Sep, 2014

Abstract

In our everyday lives, we are constantly confronted with human motion: we watch other people as they move, and we perceive our own movements. Motion, in terms of physics, is a change in the position of a body with respect to time. Since the human body is not simply a rigid block but rather a complex aggregation of flexibly connected limbs and body parts, human motion can have a very complex spatial-temporal structure. Deeper understanding on human actions is required in many applications, e.g., action recognition (security), animation (sport, 3D cartoon movies and virtual world), etc.

With the development of the technology like 3D specialized markers, we could capture the moving signals from marker joints and create a huge set of 3D action motion capture (MOCAP) data which is capable of accurately digitizing a motions spatial-temporal structure for further processing on a computer. Recently, motion capture data have become publicly available on a larger scale, e.g. CMU, HDM05. The task of automatic extraction of semantic action features is gaining in importance. The underlying questions are how to measure similarity between motions or how to compare motions in a meaningful way. The main problem is that the granularity of MOCAP data is too fine for our purpose: Human actions typically exhibit global and local temporal deformation, i.e. different movement speed and timing difference. The similar types of motions may exhibit significant spatial and temporal variations. The irrelevant details (like noise) as well as spatial pose deformations may interfere with the actual semantics that we are trying to capture. The problems require the identification and extraction of logically related motions scattered within the data set. This leads us to the field of motion analysis for identifying the significant spatial and temporal features of an action.

To automatically extract the action features from 3D MOCAP data, we proposed two approaches dealing at features levels: *1) Extract of Discriminate Patterns from Skeleton Sequences approach* provides a foundation in lower dimensional representation for the movement sequence analysis, retrieval, identification and synthesis; and *2) Automatic Extraction of Semantic Action Features approach* which focuses on solving the high-dimensional computational problems arising from the human motion sequences. They support the follow-up stages of processing the human movement on a natural language level. As one common underlying concept, the proposed approaches contain a retrieval component for extracting the above-mentioned features.

Firstly, the extraction of discriminative patterns as local features and the utilization of a statistical approach in text classification to recognize actions. In text classification, documents are presented as vectors where each component is associated to a particular word from the code book. Traditional weighting methods like Term Frequency Inverse Document Frequency (TF-IDF) are used to estimate the importance of each word in the document. In this approach, we use the beyond TF-IDF weighting method to extract discrimination patterns which obtain a set of characteristics that remain relatively constant to separate different categories. This weighting defines the importance of words in representing specific categories of documents. It not only reduces the number of feature dimension compared to the original 3D sequence of skeletons, but also reduces the viewing time of browsing, bandwidth, and computational requirement of retrieval.

Secondly, we propose the semantic annotation approach of the human motion capture data and use the Relational Feature concept to automatically extract a set of action features. For each action class, we propose a statistical method to extract the common sets. The features extracted is used to recognize the action in real-time. We extract the set of action features automatically based on the velocity feature of body joints. We consider this set as action spatial information. We combine both spatial and temporal processes to extract the action features and use them for action recognition. In our experiments, we use the 3D motion capture database HDM05 for performance validation. With few training samples, our experiment shows that the features extracted by this method achieves high accuracy in recognizing actions on testing data. Our proposed method gets high accuracy comparing to others state-of-art approaches.

Keywords: Discriminate Semantic Features; Automatic Extraction Features; Semantic Action Features; Action Recognition; Joint Velocity; Local Descriptor; Relational Feature; 3D Motion Capture Data; Deep Learning; Depth Architecture.

Acknowledgment

I wish to express my sincere gratitude to principal advisor Associate Professor Kazunori Kotani of Japan Advenced Institute of Science and Techology(JAIST) for the constant encouragement and kind guidance during my study. I would like to thank Assistance Professor Fan Chen for the helpful discussions, valuable suggestions and kind encouragements during my research.

To Professor Akira Shimazu, Associate Professor Atsuo Yoshitaka of JAIST and Associate Professor Le Hoai Bac of University of Science VNU-HCM, Vietnam who inspired me constant activities on information science, computer vision, data mining and gave me valuable suggestions and kind encouragements.

I am grateful to Professor Ho Tu Bao of Knowledge School - JAIST gave me a chance to do research, affected and suggested my areas of research.

I would like to thank the FIVE-JAIST Joint Education Program (between Japan and Vietnam) and the Research Grants Foudation from JAIST for supporting my research.

Sincere thanks to my parents who always encourage me to study. I devotes my sincere thanks, appreciation my friends and colleagues for helping me to brain storm some idea and enjoy the life.

Contents

Abstract **i**

Acknowledgments **iii**

1 Introduction **1**
1.1 Objective 1
1.2 Contributions 2
1.3 Summary of chapters 4
1.4 Related Works 6

2 Marker-based Motion Capture Data **10**
2.1 Marker-based Motion Capture Data 10
2.2 Technical Motion Capture Data 11
2.3 HDM05 Database 13

3 Basic Concepts **16**
3.1 Extraction of Discriminative Patterns from Skeleton Sequences for Accurate Action Recognition 16
3.1.1 Three-Dimensional Skeleton Histogram 16
3.1.2 Extraction of Discriminative Patterns 17
3.2 Automatic Extraction of Semantic Action Features 19
3.2.1 Relational Features 19
3.2.2 Action recognition 21

4 Extraction of Discriminate Patterns from Skeleton Sequences **24**
4.1 Extraction of Discriminate Patterns 24
4.1.1 Key-frames Extraction 24
4.1.2 Patterns Definition 26
4.1.3 Increasing the discriminative patterns values 27
4.2 Action Recognization 28
4.3 Experimental Results 29
4.4 Conclusion 32

5 Automatic Extraction of Semantic Action Features **33**
5.1 Automatic Extraction of Active Parts and Active Frames 34
5.1.1 Skeleton and Body Joints 35
5.1.2 Velocity of Body Joints and Parts 35
5.1.3 Extract Active Parts 35

5.1.4 Extract Active Frames 36
5.2 Extract Action Spatial Information 36
5.3 Extract Action Temporal Information 37
5.3.1 Sequence and sub-sequence of action signals 38
5.3.2 Common action features sets 38
5.3.3 Temporal Information from Common Action Features 40
5.3.4 Action Recognition 40
5.4 Experiments 41
5.5 Conclusion 43

6 Conclusions and Future Work **45**
6.1 Future work 45

References **50**

Publications **53**

APPENDICES **54**

A Relational Feature **54**
A.1 Relational Feature Design 54
A.2 Choosing Thresholds 57
A.3 Robust Threshold 58

B Using Action Features for Real-time Action Recognition **59**
B.1 Introduction 59
B.2 Depth Architecture Network 59
B.3 Action Recognition 62
B.4 Experiments 62
B.4.1 Experiment 1: Multilevel-networks and Single-network with the changing window size 62
B.4.2 Experiment 2: The combining depth architecture networks 65

C Dealing with the Large Number of Actions **68**
C.1 Automatic Extraction Action Features 69
C.2 Deep Architecture Networks and Real-time Action Recognition 71
C.2.1 Depth Architecture Networks 71
C.2.2 Sequences of Images 73
C.2.3 Real-time Action Recognition 73

List of Figures

1.1 Research Diagram 3

2.1 Passive optical motion capture system based on retro-reflective markers attached to the actor's body [25]. 11
2.2 Comparison of corresponding poses from a C3D file (left, point cloud) and an AMC file (right, skeleton) [25]. 12
2.3 Skeletal kinematic chain model: rigid bones, joint names. 14
2.4 Some examples of different action categories. 14
2.5 Some examples of semantic action categories. 15

3.1 Skeleton histogram computes the distribution of joint points in different divisions. 17
3.2 Visualizations of the selected discriminative patterns from different actions. 18
3.3 Relational feature *"Right Foot is behind Left Leg"*. 20
3.4 Comparing different actions recognition accuracy between AECAC to EDPSS methods. 22

4.1 Illustration of automatic key-frames selection 25
4.2 The whole sequence of 3D skeleton is divided into unit actions. Each unit action is represented from a key-frame, which includes the central frame and the period it covers. These key-frames are then identified as different patterns, according to their poses. 26
4.3 Increasing the discriminative patterns values 28
4.4 Confusion matrix for testing (A) and training (B) data sets 30
4.5 The ROC curve of the true positive rate with respect to false positive rate under various $conf$. 31
4.6 The accuracy has been improved on both testing and training datasets, by increasing the amount of discriminative patterns with pattern expansion. . 32

5.1 Flow chart of the proposed approach. 34
5.2 Calculation of Active Frames from Velocity Matrix. 36
5.3 Testing accuracy with the number of training samples. 42

B.1 Flow chart of the proposed approach. 60
B.2 A five-layers HTM deep learning structure. 61
B.3 The accuracy of two depth architecture networks with the changing of window size $wsize$ (Eq.(B.5)). 64
B.4 Confusion matrix of *multilevel-networks* for five actions No.1-5 with $wsize = 20$. 64
B.5 Depth architecture for 10 actions. 65

B.6 Confusion matrix 10 actions. 67

C.1 Flow chart of the proposed approach. 69
C.2 Groups and sub-groups of actions. 70
C.3 The structure of node group G_i. 72
C.4 Deep architecture network. 72

List of Tables

3.1 List experiment actions 22

4.1 Algorithm for the expansion process of sample patterns. 28
4.2 List of experiment actions 30

5.1 The top 10 candidate extracted RFs from action $walkRightCircle4StepsRstart$. 38
5.2 Two candidates middle sequences selected from action *walk2StepsLstart*. . 40
5.3 List of 10 similar walking actions. 41
5.4 List of experiment actions. 42
5.5 Comparison of state-of-art methods. 43

A.1 List some manual relational features. 57

B.1 List of experiment actions. 63
B.2 Networks parameters. 63

Chapter 1

Introduction

1.1 Objective

Action recognition has been widely researched and applied in many domains, including visual surveillance, human computer interaction and video retrieval. Motion, in terms of physics, is a change in the position of a body with respect to time. We want our computers to automatically interpret the activities that people perform. Given a motion sequence, we expect the program to automatically identify the actions being performed by the human subject. Motivated by this, human motion analysis has been a highly active research area in computer vision, whose goal is to automatically segment, capture, recognize and predict ongoing human activities in real-time. With deeper understanding on human actions, this research can be widely applied to various domains. Analysis and synthesis of animation in virtual world as sport or 3D cartoon movies. Home care for elderly people and children could use live video streaming from an integrated home monitoring system to prompt timely assistance. Moreover, automatic human motion analysis can be used in HumanComputer/Robot Interaction, video retrieval, virtual reality, computer gaming and many other fields.

With the development of the technology like 3D specialized markers, we could capture the moving signals from marker joints and create a huge set of 3D action MOCAP data which is capable of accurately digitizing a motion's spatial-temporal structure for further processing on a computer. Recently, motion capture data has become publicly available on a larger scale. These databases is a digital representation of the complex structure of human motion and widely used for the synthesis of realistic computer-generated characters in data-driven computer animation and also plays an important role in motion analysis tasks such as activity recognition, retrieval, and classification. Regarding synthesis and analysis, the human perceptual system sets high quality standards: our vast experience with natural human motion enables us to easily detect unnatural or synthetic motions, and even low-level motion analysis tasks that seem so easy to us think of segmenting continuous motion into basic behavioral units. The more we understand the human action, the better we could apply to many applications.

The task of automatic extraction of semantic action features within the data set is gaining in importance. This leads us to the field of motion analysis and action recognition. Since the human body is not simply a rigid block but rather a complex aggregation of flexibly connected limbs and body parts, human motion can have a very complex spatial-temporal structure. The underlying questions are:

- *How to measure similarity between motions,* and
- *How to compare motions in a meaningful way.*

The main problem is that the granularity of MOCAP data is too fine for our purposes:

- Human actions typically exhibit global and local temporal deformation, i.e. different movement speed and timing difference.
- The similar types of motions may exhibit significant spatial and temporal variations.
- The irrelevant details (like noise) as well as spatial pose deformations may interfere with the actual semantics that we are trying to capture.

A key issue of feature extraction is to identify the significant spatial and temporal features of an action.

1.2 Contributions

We propose in this thesis novel approaches to automatically extract the action features from 3D MOCAP data for real-time action recognition. In particular, we contribute to two different areas dealing with variation at different features levels as showed in the Figure 1.1., i.e., *1) Extract of Discriminate Patterns from Skeleton Sequences approach (EDPSS)* is based on features that are close to the raw data. It provides a foundation in lower dimensional representation for the movement sequence analysis, retrieval, identification and synthesis; and *2) Automatic Extraction of Semantic Action Features approach (AESAF)* focuses on solving the high-dimensional computational problems arising from the human motion sequences. It supports the follow-up stages of processing the human movement on a natural language level. As one common underlying concept, the propose approach contains a retrieval component for extracted action features. Because of the computationally searching expensive, the content-based approach are used for the offline action retrievals while the semantic-based approach can be used for real-time action recognition.

Firstly, we extract the discriminate patterns as local features and the utilization of a statistical approach in text classification to recognize actions. In text classification, documents are presented as vectors where each component is associated to a particular word from the code book. Traditional weighting methods like Term Frequency Inverse Document Frequency (TF-IDF) weighting are used to estimate the importance of each word in the document. In this work, we use the beyond TF-IDF weighting method to extract discrimination patterns which obtain a set of characteristics that remain relatively constant to separate different categories. This weighting defines the importance of words in representing specific categories of documents. It not only reduces the number of feature dimension compared to the original 3D sequence of skeletons, but also reduces the viewing time of browsing, bandwidth, and computational requirement of retrieval.

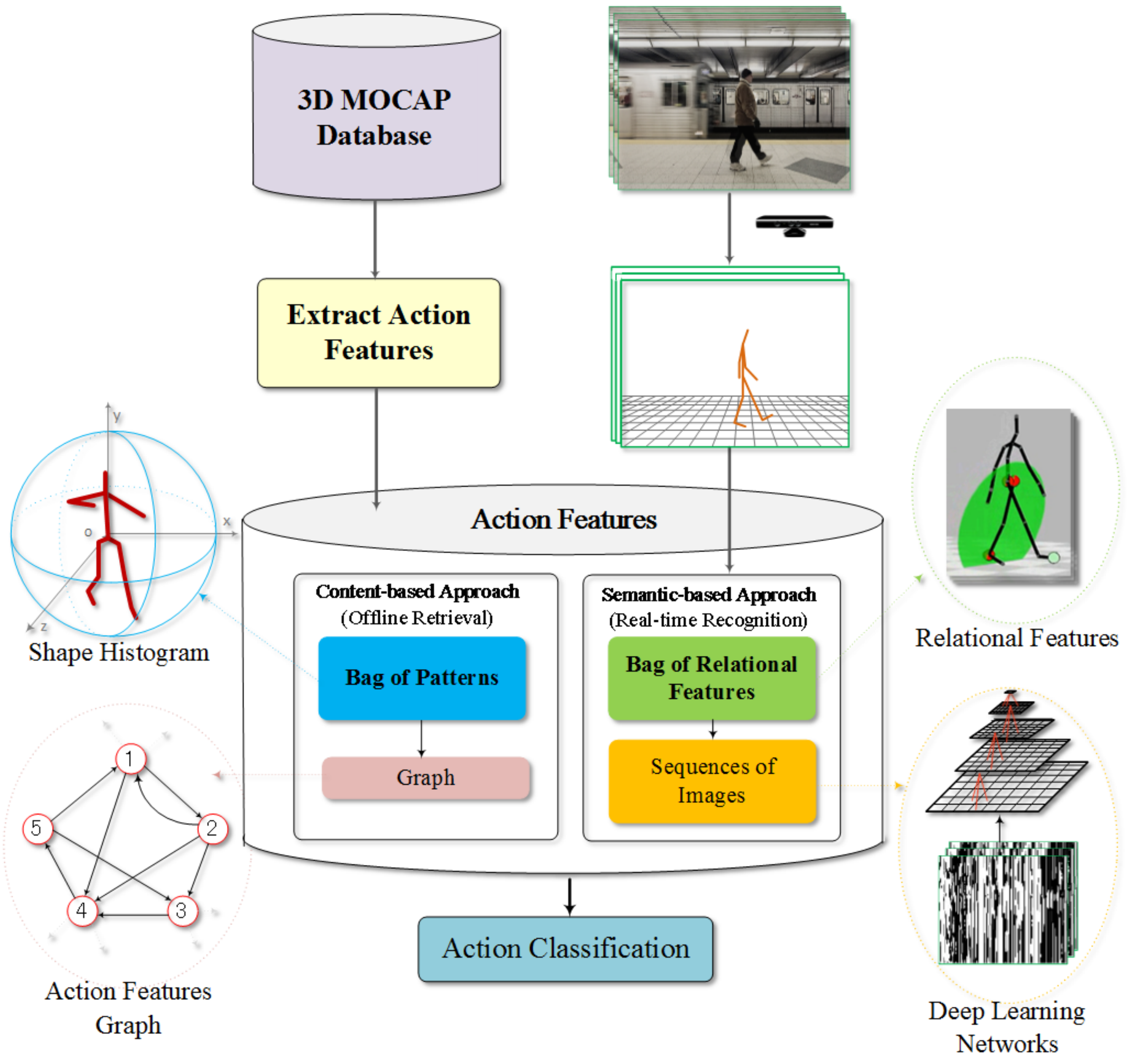


Figure 1.1: Research Diagram

Secondly, the first approach lefts some semantic ambiguous between extracted patterns because of dealing with the raw data. It requires us to further improve the works to semantic annotation approach. We introduce the semantic annotation approach of the human motion capture data and use the Relational Feature (RF) concept to automatically extract a set of action features. For each action class, we propose a statistical method to extract the common sets. The features extracted is used to recognize the action. We extract the set of action features automatically based on the velocity feature of body joints. They are considered as action spatial information. We combine both spatial and temporal processes to extract the action features and use them for action recognition. In experiments, we use the 3D MOCAP database for performance validation. By using the semantic features, we show that this proposed method is very robust to the different actors' skeletons and gives the higher accuracy than previous approach in action recognition. The semantic action features extracted could provide very high accuracy in the action recognition, which demonstrate the efficiency of our method in extracting discriminate semantic action features and comparable to others state-of-art methods.

1.3 Summary of chapters

Chapter 2: Marker-based Motion Capture Database

We provide in this chapter some general information on MOCAP databases. We begin with a brief summary of marker-based MOCAP data, the technical recording system and data file formats (e.g. marker-based and skeleton-based). We review the HDM05 database which is mainly used in our experiments.

Chapter 3: Basic Concepts

We overview in this chapter the basic concepts on the main work flow of the features extraction for action recognition process. We briefly explain the difficulties at each stage of the process and highlight our motivation of the proposed methods. We first propose the Extract of Discriminate Patterns from Skeleton Sequences approach dealing with the raw 3D MOCAP data. We extract the discriminate patterns as local features and the utilization of a statistical approach to recognize actions. Because of using a weighting factor - Shape Histogram, it creates the semantic ambiguous in the recognize results. To solve this issue, we propose the semantic annotation approach of the human motion capture data and use the Relational Feature concept to automatically extract a set of action features. By using the same data set, we have a comparison between our propose approaches.

Chapter 4: Extract of Discriminate Patterns from Skeleton Sequences for Accurate Action Recognition

In this chapter, we introduce a method to extract *discriminate patterns* as local features to classify skeleton sequences in human action recognition. Based on skeleton histogram, we extract *key-frames* from 3D videos and define patterns as local features. We use the weighting factor beyond TF-IDF method in document analysis, to estimate the importance of discriminate patterns for a specific categorization. We also propose an algorithm for further increasing the number of discriminate patterns. The whole processing can be divided into four main parts:

- *Extract key-frames:* We first compute histograms of the 3D skeleton sequences from different actions, and perform time-warping to extract the key-frames from the sequences of those histograms. We use the shape histogram to represent the 3D skeleton data. A 3D skeleton is defined as a set of bone segments that lie inside the model. Around the skeleton center of the model, the 3D space of this skeleton is transformed to a spherical coordinates system. Given an action sequence, we first divide it into a series of short temporal segments. From each segment, we choose one frame as its representative state, which is named as a “key-frame”. We use a graph-based method to automatic extract key-frames to summarize 3D video sequences.

- *Define patterns:* Each key-frame is defined as a central frame with the neighborhood size it represents. We define patterns from extracted key-frames and then use a statistical approach to evaluate the discriminate level for each pattern.

- *Extract discriminate patterns:* Inspired by ideas in natural language processing, we regard the list of patterns as a document, and consider each pattern as a word. We use a statistical metric to evaluate the discriminate capability of each pattern; we further define the bag of reliable patterns as local features. The problem of recognizing an action is then turned into a problem of classifying the document into the right category.

- *Action recognition:* Given a testing sequence, we first convert it to a sequence of discriminate patterns. We use the extracted discriminate patterns to calculate the spatial (the pattern weight) and temporal (the order appearance of patterns) confidence of this testing sequence in belonging to a certain action class.

We present some experimental results of using discriminate patterns extracted for action recognition. The results indicate that the action recognition model using these discriminate patterns gives high accuracy on the testing data-set.

Chapter 5: Automatic Extraction of Semantic Action Features from 3D Action MOCAP Database

In semantic-based approach, we automatically extract the common sets for each action class from 3D MOCAP data. We propose the semantic annotation approach of the human motion capture data and use the relational feature concept to automatically extract a set of action characteristics. For each action class, we propose a statistical method to extract the common sets. The features extracted is used to recognize the action. The whole process contains three main parts:

- *Extract Active Frames from 3D MOCAP data:* For each action category, we calculate the velocity of body joints and select the most active parts (as sets of joints) which have high average velocities. Based on these active parts, we can extract the active-frames for each sample in the database.

- *Extract Spatial Information:* We consider the General Relational Features, which express the geometric relationship among the set of some 3D points, are the action units. We apply them to different set of body joints and extract the common action features on Active-Frames in the database. The sets of action characteristics in each action are considered as words of each document category. We use the weighting factor beyond TF-IDF to calculate the words weighting. We select the top-N RFs which have highest weighting and considered as the spatial action information.

- *Extract Temporal Information:* For each action class and its spatial space, we propose a statistical method to extract the common sets as action temporal information. We combine both spatial and temporal information features and use them for action recognition.

We use the 3D motion capture data from HDM05 motion capture database for performance validation in our experiments. With few training samples, our experiment shows that the features extracted by this method achieves high accuracy in recognizing actions on testing data. This method gets high accuracy comparing to others state-of-art approaches.

Chapter 6: Conclusion and Future Work

We conclude our contributions and have some discussion for future works in this chapter.

1.4 Related Works

In this section, we briefly review the features of widely used in representing action data and their extraction. A key issue of feature extraction is to identify the spatially and temporally significant characteristic of an action. Some dimension reduction methods use spatial-temporal interest points or global statistics, which are reviewed in [1]. Previous methods of feature extraction can be divided into two categories, i.e., *content-based* and *semantic-based*.

- Content-based approaches provide a foundation in lower dimensional representation for the movement sequence analysis, retrieval, identification and synthesis;
- Semantic-based approaches focus on solving the high-dimensional computational problems arising from the human motion sequences. They support the follow-up stages of processing the human movement on a natural language level.

Content-based approaches

Previous methods in content-based approaches can be roughly divided into two categories, i.e., holistic features and local descriptors.

Holistic features represent the shapes of objects in given time period as the global information. These approaches often use the concept of "motion graphs" and the nearest-neighbor-search as a central searching tool. They include methods using joint angles [2], principal components [3] or point clouds [4] as the distance metrics to create the motion graph from the sequences of 3D joints for each action. By using joint angles as the distance metric, Alla et al. [5] represent the desired motion as an interpolation of two time-scaled paths through a motion graph. The graph is constructed to support interpolation and pruned for efficient searching. For each motion, they synthesize optimal or sub-optimal motions that include various behaviors. The A^* search is used to find a globally optimal solution in graphs that satisfies the users specification. This approach not only retains the natural transitions in the motion graphs, but also has ability to synthesize physically realistic variations provided by interpolation. Forbes et al. [6] project sequence of 3D joints in motion data into the PCA space, wherein weights can be applied to both subsections of individual frames, and to entire frames. The motions are modelled by high-dimensional parametric curves, with each sample point representing a single pose. The geometric operations such as scaling and translating are used to modify motions. Their retrieval algorithm is based on a weighted PCA-based pose representation that allows flexible and efficient computation of pose-to-pose distance. The query clip and database are projected into a user-specified PCA space to find the characteristic points. By using sorted data structure, they find the indices of all similar poses in the database, and use Dynamic Time Warping (DTW) to warp the database sub-regions surrounding to match the query clip. Kovar et al. [7] use point cloud distance measure on a normalized window. They build "match webs" based on dense distance matrices. They perform numerical and "logical" similarity searches on collections of captured motion data. Their construction of "match webs" is $O\left(n^2\right)$ where n is the number of frames included in the database.

The searching methods on graphs or DTW on matrices have been used for classifying the query clip. Recently, Krüger et al. [9] described and analyzed medium dimensional feature sets for human actions. These can be used with naturally occurring Euclidean distance measures in standard spatial data structures - specifically *kd*-trees - to perform fast exact and approximate similarity searches in large motion capture databases. By avoid using *ad*-hoc heuristics, it gives similar results to subsequence DTW in practical scenarios and is more general and robust than "match webs". Their novel method is only $O\left(m \log n\right)$ for construction graph.

There are two problems need to be solved in this approach. Firstly, in order to create the motion graphs, it requires optimizing the weights for specific joints to generate the transitions (graph nodes) for each action. However, for different actions there are different parameters to create the transition points. To have a high-quality transition, the users have to manually set the threshold to pick an acceptable trade-off between having good transitions (low threshold) and having high connectivity (high threshold). Secondly, finding paths in the motion graph that satisfy the hard constraints and optimize soft constraints involves a graph search are also difficult. For even a small collection of motions, the action graph has a large number of edges and straightforward search of this graph is computationally prohibitive. The main reason is the need to enumerate too many paths. In general, there are many perfectly satisfactory motions that satisfy the constraints equally well.

Local descriptors extract the information from a limited neighborhood of interest points and thus focus more on local motions. These features are the crucial elements of the actions. Their distribution is then used to represent the entire sequence as global representation. There are only a few of researches considering 3D skeletons as local descriptors, mainly due to the difficulty of reliable skeleton extraction. Various shape descriptors have been proposed to represent the 3D skeleton: Shape Distribution, Spin Image, Shape Histogram, Spherical Harmonics, etc. In this work, we use shape histograms to represent 3D skeletons instead of 3D surface meshes. The 3D shape histogram was first introduced by Ankerst [13]. There are different properties between 3D surface meshes and 3D skeletons. With the same position of body parts, two surface meshes could be different due to their clothes or body shapes while two skeletons are nearly the same. Huang et al. [14] presented a performance evaluation of shape similarity metrics for 3D video sequences of people with unknown temporal correspondence. They used optimal parameter setting for each approach and compared different similarity measures. By evaluating the self-similarity for a list of actions, they concluded that the Shape Histograms with volume sampling, which is also used in this work, consistently give the best performance for different people and motions in measuring 3D shape similarity with unknown correspondence.

Kilner et al. [10] use an appropriate feature descriptor and distance metric to extract action key-poses in 3D human videos. They used a key-pose detector to identify each key-pose in the target sequence. The Markov model is used to match the library to the target sequence and estimate the matched action. Baak et al. [11] proposed a novel key-frame-based method for automatic classification and retrieval of capturing motion data. They label individual frame from a set of relational features which describe geometric relations between specified points of poses, then use them to extract key-frames. They proposed algorithms to measure the hits and matching between two sequences of key-frames for retrieving. By this way, however, it is hard to minimize the set of relational features for different actions because different actions should have different sets of relational features. There are also some researches [1,12] that combine both holistic features and local descriptors to further improve the performance.

Semantic-based approaches

Semantic-based approaches focus on solving the high-dimensional computational problems arising from the human motion sequences. They support the follow-up stages of processing the human movement on a natural language level. The semantic-based processing is still less developed. They focus on how to measure similarity between motions, how to compare motions in a meaningful way or how to use extraction features for real-time action recognition. It requires the understanding of the capture motion action and expresses definition of object motion models. Arikan et al. [15] proposed a semi-automatic annotation procedure, where a user is required to annotate a small portion of the database. The user annotations are then generalized to the entire database in a frame-wise fashion using SVM classifiers.

Muller et al. [16] propose the concept of Relational Features which describe the Boolean geometric relations between specified body points of a pose. RFs provide a compact, semantically meaningful representation of motions and are largely invariant to various kinds of spatial deformations of joints. This annotation strategy is segment-based instead of frame-based, thus resulting in semantically more meaningful units. Muller et al. [8] use binary geometric features and index structures. These binary geometric features are well suited for defining notions of logical similarity of motions and for coming up with matrices as “motion templates”. They use sub-sequence DTW in searching for the query clip with the complexity in $O(mn)$ where m is the size of the query, n is the number of frames included in the database. Muller and Roder [17] further develop the concept of motion templates which capture consistent and varying aspects of a set of motions and a fast index-based motion retrieval procedure. Gao et al. [18-19] adopt a pre-computation strategy in the indexing structure from RF concept [16], which can speed up the retrieval process. They present a scene description language to describe motions effectively, which bridges the gap between user’s vague perception and explicit motion scene description. Baak et al. [20] further present a novel key-frame based algorithm using RFs concept. A key-frame query consists of a sequence of key-frames, where each key-frame is specified by a Boolean feature vector that describes features relations of a specific pose.

For action retrieval or recognition, the above systems require the user to incorporate some prior knowledge about the intended motion hits by selecting suitable features or definition of semantic relational features. This manual interaction is a disadvantage in view of fully automatic retrieval, as required for automatic motion reuse applications. The main contribution of this work fills this gap by automatically extracting the common sets for each action class. We propose the semantic annotation approach of the human motion capture data and use the relational feature concept to automatically extract a set of action features. For each action class, we propose a statistic method to extract the common sets and mining their logical order. The action features extracted is used to recognize action.

Chapter 2

Marker-based Motion Capture Data

We provide in this chapter some general information on motion capture (MOCAP) data and the main issues of extracting features for action recognition. We begin with a brief summary of marker-based MOCAP data in section 2.1. We have an overview the technical and recording system, files formats (e.g. marker-based and skeleton-based) in section 2.2. We review HDM05 database which is mainly used in our experiments in section 2.3.

2.1 Marker-based Motion Capture Data

In the past two decades, MOCAP systems have been developed that allow to track and record human motions at high spatial and temporal resolutions. The resulting motion capture data is used to analyze human motions in fields such as sports sciences and bio-metrics (person identification), and to synthesize realistic motion sequences in data driven computer animation. Such applications require efficient methods and tools for the automatic analysis, synthesis and classification of motion capture data, which constitutes an active research area with many yet unsolved problems. There are some main purpose to use the motion capture data: *reuse in the production of animations* or *extract action features*.

The idea of motion capturing originates from the field of gait analysis, where locomotion words of humans and animals were investigated using arrays of analog photographic cameras. With technological progress, motion capture data or simply MOCAP data became popular in computer animation to create realistic motions for both films and video games. Here, the motions are performed by live actors, captured by a digital MOCAP system, and finally mapped to an animated character. The reuse of MOCAP data as well as methods for modifying and adapting existing motion clips are gaining in importance. Applying editing, morphing, and blending techniques for the creation of new, realistic motions from prerecorded motion clips has become an active field of research [21-22]. Such techniques depend on motion capture databases covering a broad spectrum of motions in various characteristics.

Prior to reusing and processing motion capture material, one has to solve the fundamental problem of identifying and extracting logically related motions scattered in a given database. In this context, automatic and efficient methods for motion analysis, comparison, classification, and retrieval are required that only access the raw MOCAP data itself and do not rely on manually generated annotations [23-24]. Such methods also play an important role in fields such as sports sciences, bio-mechanics, and computer vision.

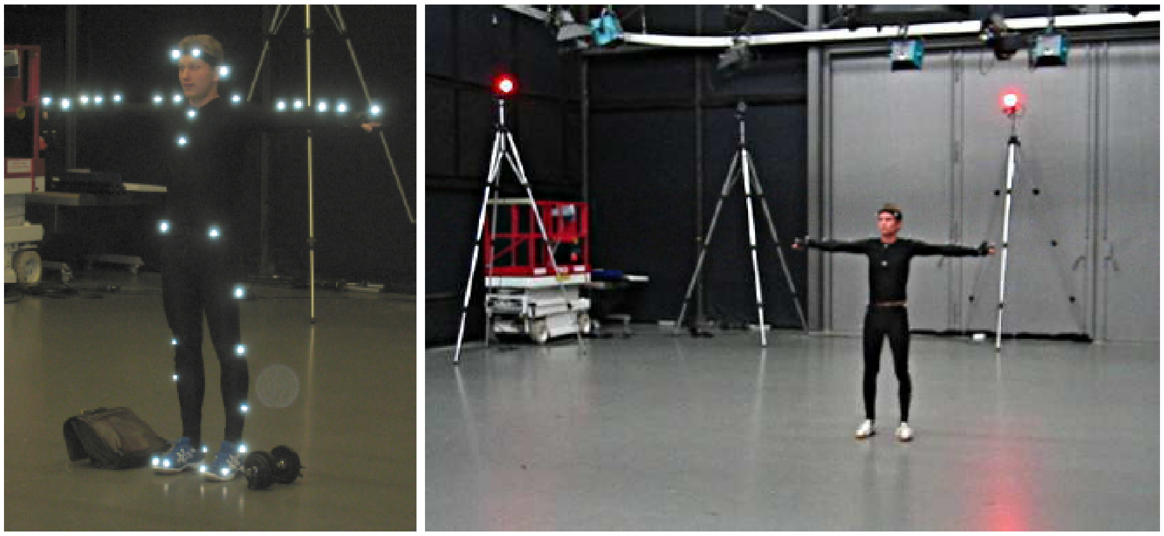


Figure 2.1: Passive optical motion capture system based on retro-reflective markers attached to the actor's body [25].

One of the first publicly available is the CMU MOCAP database [26] which has been provided in the year 2003. It contains several hours of motion data comprising various motions ranging from locomotion over sports to pantomime. The CMU database has been extensively used by the academic research community, thus providing important test data for the aforementioned research fields. Furthermore, the CMU database is a first step towards a common database that can be used by the research community for an objective comparison of different motion analysis and synthesis methods as well as a comprehensible evaluation of research results.

The HDM05 [27] database supply an additional set of motion capture data and is mainly used in our experiments in this thesis. HDM05 contains more than three hours of systematically recorded and well documented motion capture data in the 3D trajectory-based and skeleton-based files format. Furthermore, the cutting capture sets consists of 130 classes, obtained from 2337 samples and made by 5 different actors.

2.2 Technical Motion Capture Data

Other MOCAP database is the TUM Kitchen Data Set [28] which is a collection of activity sequences recorded in a kitchen environment equipped with multiple complementary sensors. The recorded data consists of observations of naturally performed manipulation tasks as encountered in everyday activities of human life. Several instances of a table-setting task were performed by different subjects, involving the manipulation of objects and the environment. This database includes the original video sequences, full-body motion capture data recorded by a marker-less motion tracker, RFID tag readings and magnetic sensor readings from objects and the environment, as well as corresponding action labels. The MSR Action3D [29] is a newer dataset, created by Microsoft Research in 2010, that aims at providing 3D data for action recognition. This dataset, first used in [21], is made up of twenty actions performed by ten actors, where each action has three instances. The depth maps of these actions were obtained with Microsoft Kinect sensor

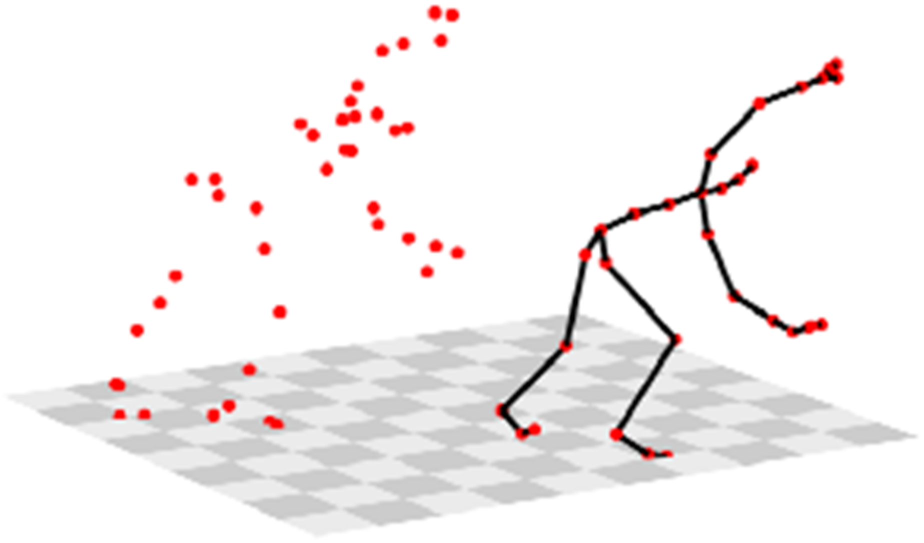


Figure 2.2: Comparison of corresponding poses from a C3D file (left, point cloud) and an AMC file (right, skeleton) [25].

and the skeletons were extracted using the method in [22]. This dataset is a challenging one due to the noise in the extracted skeletons. In particular, the obtained skeletons have more noise than the ones in the TUM dataset, and even more than the ones in HDM05 and CMU.

Modern marker-based MOCAP technology is capable of accurately tracking and recording human motions at high spatial and temporal resolutions. Such systems use cameras in order to record image data of the scene that contains markers as showed in Figure 2.1. The markers either reflect [23], or actively emit [24] light. In order to ease the detection of the marker positions, some systems use infrared light sources and cameras that work in the infrared domain [23]. From the recognized 2D pixel positions and the calibration of the cameras, 3D positions of the markers can be computed, see Figure 2.2 (left) for an example. From the 3D marker positions, the motion of the underlying skeleton can be reconstructed. To this end, the skeleton is modeled as a kinematic chain [25]. Although methods for automatically computing a suitable skeleton from just a sequence of 3D marker positions exist [30], a template skeleton is typically provided. In contrast to automatically generated skeletons, one gains full control over the admissible degrees of freedom of the movements by using a manually designed template skeleton. Using inverse kinematics optimization procedures [31], joint angles of the skeleton can be determined from the 3D marker positions, see Figure 2.2 (right) for a resulting 3D pose of the skeleton. A temporal sequence of joint angles or joint positions is referred to as MOCAP data.

There are three important differences between C3D data (3D trajectory-based) and ASF/AMC data (skeleton-based).

1. ASF/AMC data comprises an explicit skeleton structure, providing information about bones, joints, and the assembly of these basic elements into a skeleton, whereas hierarchy information for C3D data can only be deduced by the names of the markers, see also Figure 2.2.

2. Consider the bone lengths for the two data formats: fixing a pair of markers (in C3D data) or joints (in ASF/AMC data) that are attached to the same bone, the bone length can be approximated as the 3D distance of the markers/joints. Bone lengths will be constant in the case of the skeleton-based ASF/AMC format and not constant in the case of the C3D format. In fact, major variations of bone lengths over the course of a motion may be observed in C3D data. Such variations are caused by skin shifting, shifting of the nylon suit worn during recording, wobbling mass, and violations of the assumption that the human skeleton is a kinematic chain.

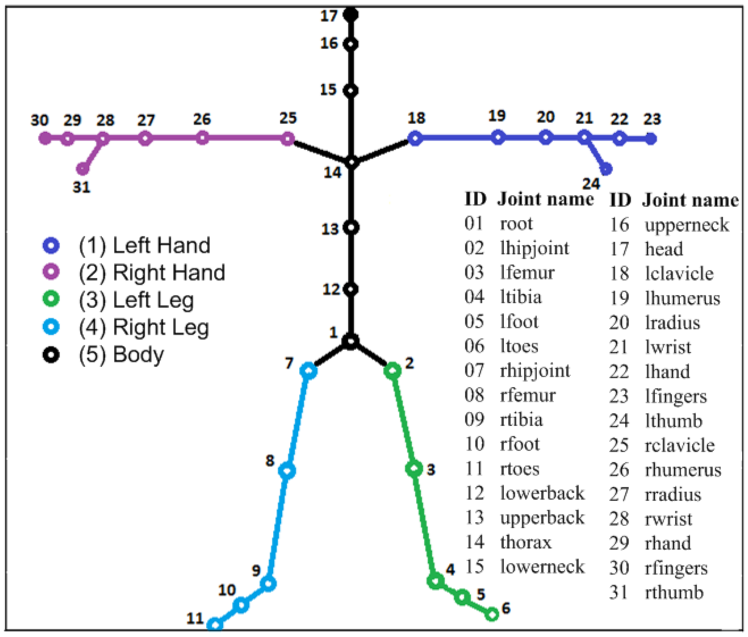
3. C3D data contains a lot of redundant markers clustered around certain joints of the human skeleton, whereas ASF/AMC data usually has only one virtual joint for each real-world joint, see also Figure 2.2.

2.3 HDM05 Database

The motion capture HDM05 database [27] is supplied free motion capture data for research purposes. This database is created from Vicon MX system comprising twelve high-resolution cameras, six of which operated in the visible red and six of which operated in the infrared spectral range. All recordings were performed at a sampling rate of 120 Hz. The cameras were set up to yield a viewing volume diameter of about five meters. Based on a script containing detailed instructions on the motions that were to be recorded, there are five actors performing several repetitions of each motion sequence.

HDM05 contains more than three hours of systematically recorded and well documented motion capture data in the C3D as well as in the ASF/AMC data format. Furthermore, HDM05 dataset consists of 130 action categories, obtained from 2337 samples, made by 5 different actors. For example, the motion class 'CartwheelLeft' contains 21 variations of a cartwheel motion, all starting with the left hand. Supplying a set of systematically recorded and well-documented set of motions that contains multiple realizations for each motion class.

The name of 31 body joints in AMC file are showed in Figures 2.3. We can divided them into five parts: Left hand, Right hand, Left leg, Right leg and the Body.



ID	Joint name	ID	Joint name
01	root	16	upperneck
02	lhipjoint	17	head
03	lfemur	18	lclavicle
04	ltibia	19	lhumerus
05	lfoot	20	lradius
06	ltoes	21	lwrist
07	rhipjoint	22	lhand
08	rfemur	23	lfingers
09	rtibia	24	lthumb
10	rfoot	25	rclavicle
11	rtoes	26	rhumerus
12	lowerback	27	rradius
13	upperback	28	rwrist
14	thorax	29	rhand
15	lowerneck	30	rfingers
		31	rthumb

Figure 2.3: Skeletal kinematic chain model: rigid bones, joint names.

We divide this database into two different action data sets: *the different actions* and *the similar actions*. In the different actions set, each action class is different from the other, e.g. 'running', 'walking', 'hopping', etc. Some examples in this set are showed in Figure 2.4. In the similar action set, one action have similar meaning to the others, e.g. for action 'walking' we have 'walking on plane', 'walking right circle', 'walking cross front', 'walking backward',etc. Some examples in this set are showed in Figure 2.5.

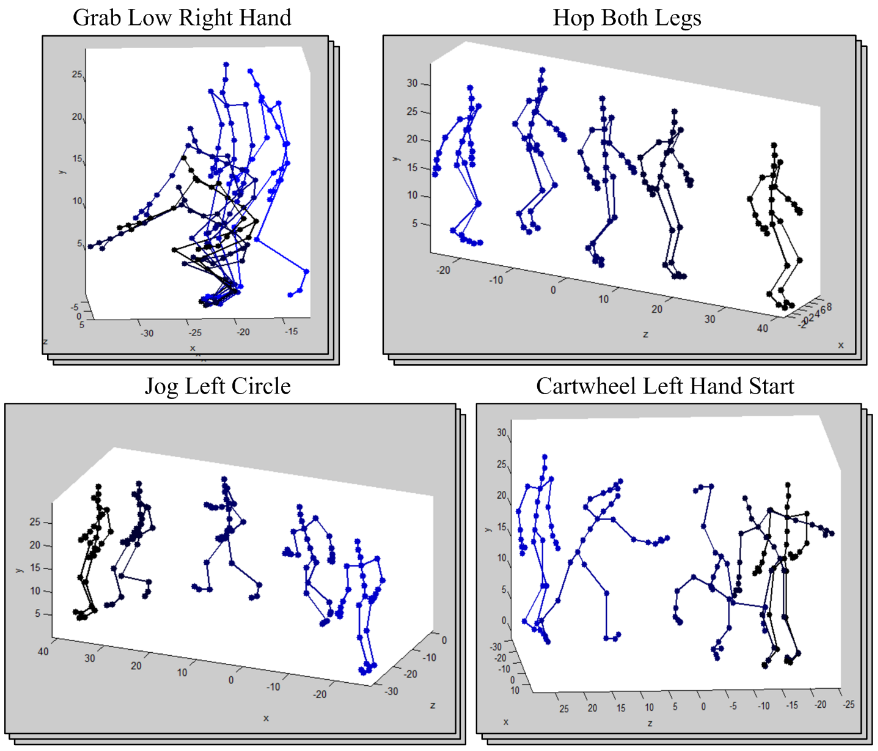


Figure 2.4: Some examples of different action categories.

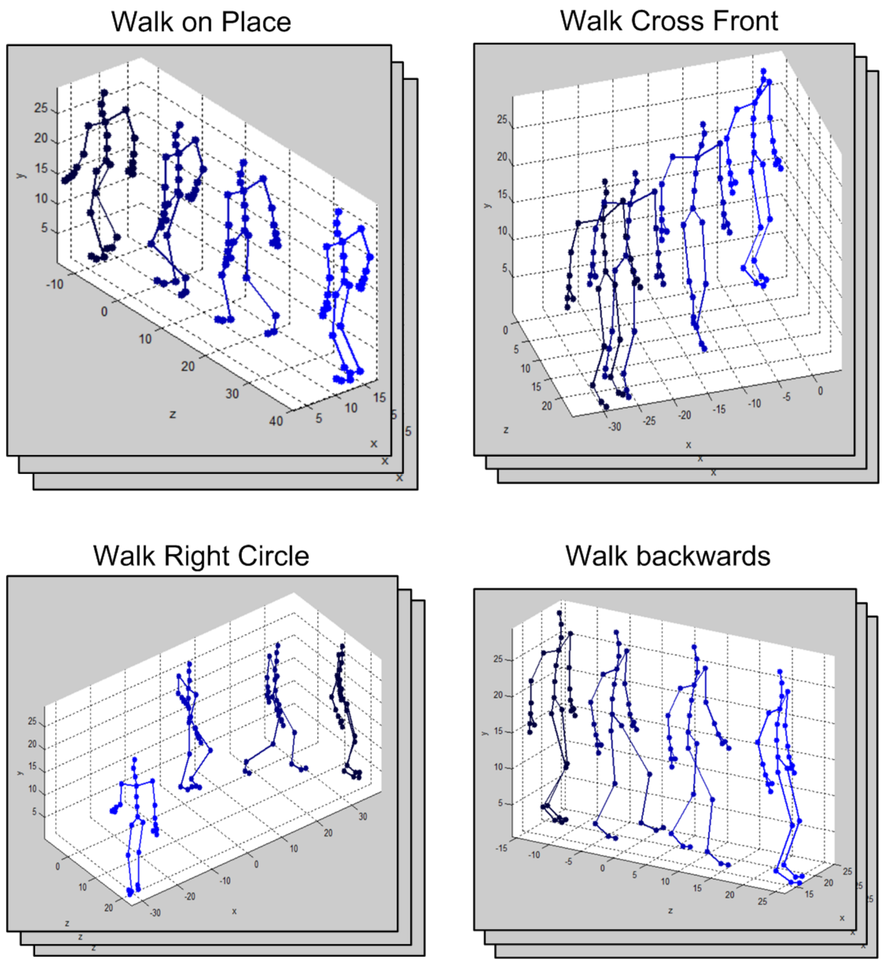


Figure 2.5: Some examples of semantic action categories.

Chapter 3

Basic Concepts

We overview in this chapter the basic concepts on the main work flow of the features extraction for action recognition process. We briefly explain the difficulties at each stage of the process and highlight our motivation of the proposed methods. The detailed explanation of these methods can be found in latter corresponding chapters.

We first propose the extract of discriminate patterns from skeleton sequences approach dealing with the raw 3D MOCAP data. We extract the discriminate patterns as local features and the utilization of a statistical approach to recognize actions. Because of using a weighting factor - Shape Histogram, it creates the semantic ambiguous in the recognize results. To solve this issue, we propose the semantic annotation approach of the human motion capture data and use the Relational Feature concept to automatically extract a set of action features. We have a comparison between the propose approaches by using the same data set.

3.1 Extraction of Discriminative Patterns from Skeleton Sequences for Accurate Action Recognition

We first propose the content-based approach dealing with the raw 3D MOCAP data. We extract the discriminate patterns as local features and the utilization of a statistical approach to recognize actions. This approach not only reduces the number of feature dimension compared to the original 3D sequence of skeletons, but also reduces the viewing time of browsing, bandwidth, and computational requirement of retrieval.

3.1.1 Three-Dimensional Skeleton Histogram

In order to perform effective 3D object retrieval, it requires a proper mathematical representation of 3D object actions. 3D video data has no hierarchical structure in each frame and no correspondences between successive frames, which makes motion analysis difficult and computationally expensive. On other hand, a 3D video data provides strong geometric information which allows us to compare a pair of frames by measuring their geometric similarity. We use the Shape Histogram [13] to represent the 3D skeleton data. A 3D skeleton is defined as a set of bone segments that lie inside the model. Around the skeleton center of the model, the 3D space of this skeleton is transformed to a spherical coordinates system (see Figure 3.1). A skeleton histogram is then constructed by accu-

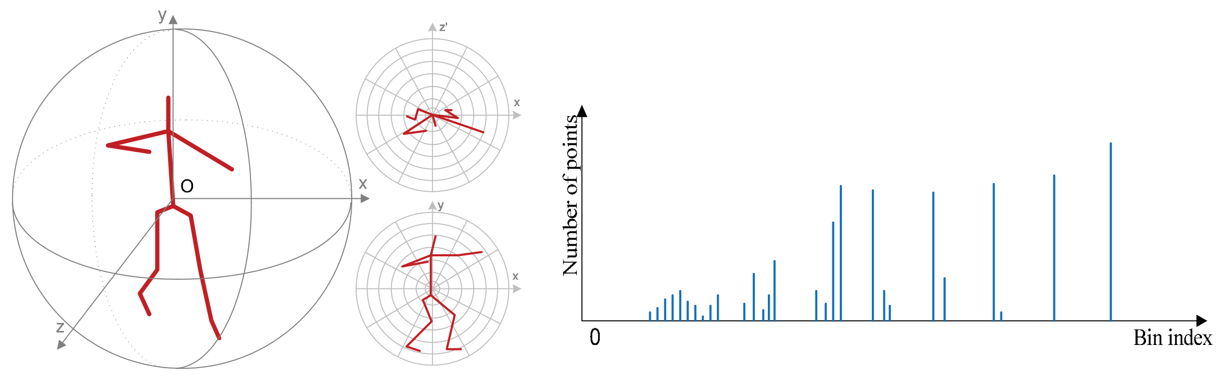


Figure 3.1: Skeleton histogram computes the distribution of joint points in different divisions.

mulating the points at each bin. The shape histogram of a point is thus a measure of the distribution of relative positions of neighboring points. The distribution is defined as a joint histogram. Each histogram axis represents a parameter in a polar coordinate system. Bins around each point are spatially defined. The number of the neighboring points in these bins is assimilated as a context to the point. Since it is computed on the gravity center of the skeleton, this histogram is translation invariant.

Along the radial direction as described in Figure 3.1, bins are arranged uniformly in log-polar space increasing the importance of nearby points with respect to points farther away. If there are X bins for the radius, Y bins for the azimuth and Z bins for the elevation, there are $X \times Y \times Z = L$ bins for the 3D shape histogram in total.

For any two skeletons A and B, let their shape histograms be denoted by $h_A(l)$ and $h_B(l)$, where $l =$ 1,2,3,...,L. The similarity between these two histograms is computed by using the χ^2 distance:

$$d(A, B) = 2 \sum_{l=1}^{L} \frac{(h_A(l) - h_B(l))^2}{h_A(l) + h_B(l)} \tag{3.1}$$

which evaluates the normalized difference at each bin of two histograms. A lower distance value between two histograms means a higher similarity between two skeletons.

Based on skeleton histogram, we extract *key-frames* from 3D videos and define patterns as local features. We use the weighting factor to estimate the importance of discriminate patterns for a specific categorization.

3.1.2 Extraction of Discriminative Patterns

Although the pattern has been defined, it is not sure to be suitable for classifying the action. In this section, we intend to include the classification information contained in the training data to filter out those discriminative patterns. In our understanding, a discriminative pattern should have two major features: 1) It appears quite often in the target sequence, so as to be considered as a reliable pattern for representing the action; 2) It appears much more often in one action than in all other actions, so that we can use it as a clue for identifying the action.

We consider to evaluate the discriminative capability of pattern p_i in classifying category C_k. Our discriminative patterns are calculated in a similar way to [1]. We estimate

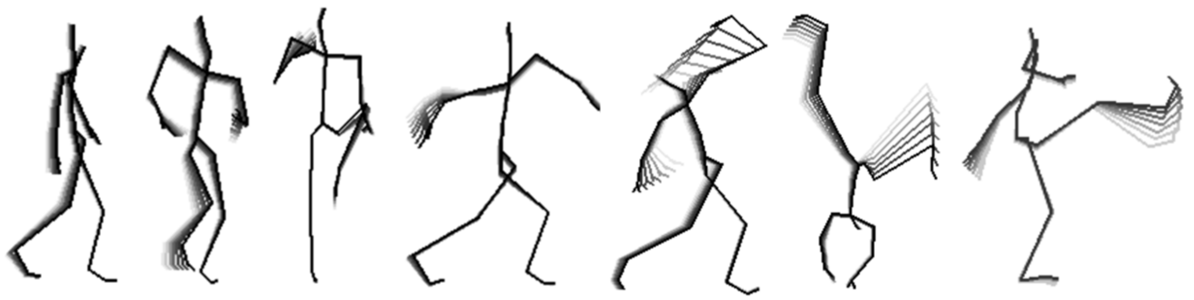


Figure 3.2: Visualizations of the selected discriminative patterns from different actions.

the proportion of sequences containing pattern p to be:

$$\tilde{p} = \frac{x_p + 0.5\mathrm{z}^2_{\alpha/2}}{n + \mathrm{z}^2_{\alpha/2}} \tag{3.2}$$

where $\tilde{p}$ is the Wilson proportion estimate [32]; x_p is the number of documents containing pattern p in the collection (category/categories); n is the size of the collection; $\mathrm{z}^2_{\alpha/2}$ is a value which makes $\Phi\left(\mathrm{z}_{\alpha/2}\right) = \alpha/2$ and Φ is the t-distribution (Student's law) function. We assume that the possibility of a pattern belonging to a specification class is a normal distribution. In our experiment, it gets the highest accuracy with 95% confidence interval, i.e., $\alpha = 5\%$ ($conf = 1.96$), which is calculated as:

$$\tilde{p} \pm conf\sqrt{\frac{\tilde{p}(1-\tilde{p})}{n + 3.84}} \tag{3.3}$$

For a given action category C_j, we denote the set of positive documents as $\tilde{p}_+$ and the set of negative documents in other categories as $\tilde{p}_-$. From Eq. (4.13), we calculate the lower range of the confidence interval of $\tilde{p}_+$, labeled as $MinPos$, and the higher range of $\tilde{p}_-$, labeled as $MaxNeg$. The strength of pattern p_i as an indicator for class C_k is calculated as:

$$Str\left(p_i \in C_k\right) = \begin{cases} \log_2\left(\frac{2MinPos}{MinPos + MaxNeg}\right) & if\ MinPos > MaxNeg \\ 0 & otherwise \end{cases} \tag{3.4}$$

which means that pattern p_i should appear more often in positive documents than negative documents, and the difference should be as significant as possible.

We calculate the discriminative value of each pattern as:

$$maxStr\left(p_i \in C_j\right) = \max_{k=1,2,\ldots,N_{cat}} \left(Str\left(p_i \in C_k\right)\right)^2 \tag{3.5}$$

In each category, the discriminative patterns are the patterns which have the high $maxStr$ values. We select some high discriminate patterns from different action classes as showed in the Figure 3.2.

Given a testing sequence, we first convert it to a sequence of discriminate patterns. We use the extracted discriminate patterns to calculate the spatial (the patterns weights) and temporal (the order appearance of patterns) confidence of this testing sequence in belonging to a certain action class. The experimental results indicate that the action recognition model using these discriminate patterns gives high accuracy on the testing data-set.

This approach not only reduces the number of feature dimension compared to the original 3D sequence of skeletons, but also reduces the viewing time of browsing, bandwidth, and computational requirement of retrieval. We describe detail this approach in chapter 4. Beside of the advantages, it creates some considering issues:

1. It is hard to set accurately the centers of shape histograms at the skeletons in the real applications because of the noises and the recording equipment's conditions.

2. The semantic meaning of patterns could be ambiguous when we use the distance measure between histograms (Equation (3.1)) because it does not specify the bins for different body parts. For example, the semantic of patterns "kick left leg" and "kick right leg" are similar when they have the same weighting value. It is due to the face that those histograms are not specific which bins are for "left leg", which ones are for "right leg".

The semantic annotation approach is required to solve above issues. By dividing the set of all body joints into different body parts, we develop our works by proposing the semantic annotation approach of the human motion capture data concept to automatically extract action features. Instead of using the raw skeleton sequences as features, we convert them into a sequence of binary codes using relational features which describe the Boolean geometric relations between specified body joints of a pose or between short sequences of poses, and show how these features induce a temporal segmentation of motion capture data streams.

3.2 Automatic Extraction of Semantic Action Features

We introduce the semantic annotation approach of the human motion capture data and use the Relational Feature concept [16] to automatically extract a set of action features. The relational features are used as an underlying feature to transform MOCAP data into a space that is invariant to global translations and rotations. Moreover, by projecting the MOCAP data onto semantically meaningful relations, important and discriminate aspects of the motion are retained while a high degree of invariant to subtle and person-specific details in the motions is achieved. For each action class, we propose a statistical method to extract the common sets. The features extracted is used to recognize the action. We extract the set of action features automatically based on the velocity feature of body joints. They are considered as action spatial information. We combine both spatial and temporal processes to extract the action features and use them for action recognition.

3.2.1 Relational Features

The concept of relational features was first introduced by Muller et al. [16]. Relational features describe the boolean geometric relations between specified body joints of a pose or between short sequences of poses, and show how these features induce a temporal

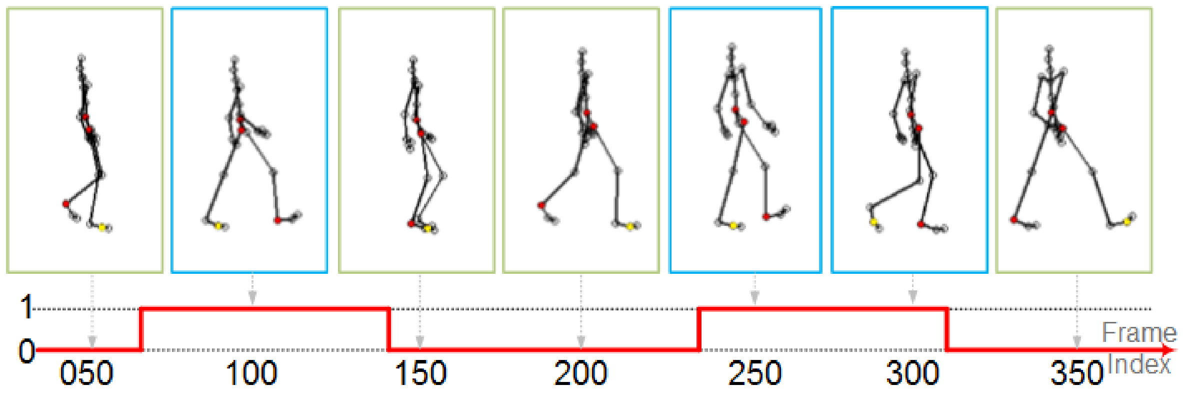


Figure 3.3: Relational feature *"Right Foot is behind Left Leg"*.

segmentation of motion capture data streams. For example, assume that we have four specific body joints, i.e. j_1="*root*", j_2="*left-hip*",j_3="*left-toes*", and j_4="*right ankle*", the generic relational feature that expresses "*The right foot is behind the left leg*" can be formulated as the joint j_4 lies in front of the plane $F_{plane}(j_1, j_2, j_3)$ that is spanned by joints (j_1, j_2, j_3). This relational feature is demonstrated in the Figure 3.3. The right foot is behind the left leg when it gets value 1 as showed as the blue rectangles. The four joints can also be picked in various meaningful ways for representing different unit actions, and result in different generic relational features.

The set of RFs is created by a small number of Generic Relational Features (GRFs) which encode certain joint constellations in 3D space and time. Each of these features maps a given joint to the set of (0, 1) and depends on a set of joints, denoted by $j_1, j_2, ...$ to create the set of relational features.

The relational feature is a function stream f frames that only assumes the values zero and one:

$$F : P \to \{0, 1\}^f \tag{3.6}$$

Most of our features have been constructed from the following six GRFs, which encode certain geometric and kinematic joint constellations.

1. $\mathrm{F_{plane}} = \mathrm{F}^{(\mathrm{j_1,j_2,j_3;j_4})}_{\theta,\mathrm{plane}}$ assumes the value one iff joint j_4 has a signed distance greater than $\theta \in \mathbb{R}$ from the oriented plane spanned by the joints j_1, j_2 and j_3.

2. $\mathrm{F_{nplane}} = \mathrm{F}^{(\mathrm{j_1,j_2,j_3;j_4})}_{\theta,nplane}$ is similar to $\mathrm{F_{plane}}$, but the plane in terms of a normal vector given by joints j_1, j_2, and passes through the anchor joint j_3. The Hesse normal form yields the signed distance of joint j_4.

3. $\mathrm{F}_{angle} = \mathrm{F}^{(\mathrm{j_1,j_2,j_3;j_4})}_{\theta,\mathrm{angle}}$ assumes the value one iff the angle between the directed segments determined by line (j_1, j_2) and line (j_3, j_4) is within the threshold range $\theta \in [0, \pi]$.

4. $\mathrm{F}_{nmove} = \mathrm{F}^{(\mathrm{j_1,j_2,j_3;j_4})}_{\theta,\mathrm{nmove}}$ considers the velocity of joint j_4 relative to joint j_3 and assumes the value one iff the component of this velocity in the direction determined the line segment from joint j_1 to j_2 is above .

5. $\mathrm{F}_{move} = \mathrm{F}^{(\mathrm{j_1,j_2,j_3;j_4})}_{\theta,\mathrm{move}}$ is similar semantics to F_{nmove}, but the direction is given by the normal vector of the oriented plane spanned by joints j_1,j_2 and j_3.

6. $\mathrm{F}_{fast} = \mathrm{F}^{(\mathrm{j}_1)}_{\theta,\mathrm{fast}}$ assumes the value one iff joint j_1 has an absolute velocity above θ.

The detail of Features Design of these GRFs, Choosing Thresholds and Robust Threshold θ are described in Appendix A.

Regarding GRFs as the letters in the alphabet, we treat the set of RFs as the bag of words created from these letters. The set of relational features is created by applying the set of GRFs to special poses and joints. Each relational feature has a different semantic meaning. For each action class, we propose a statistical method to extract the spatial and temporal information:

- *Extract Spatial Information:* The sets of action characteristics in each action are considered as words of each document category. We use the weighting factor to calculate the words weighting. We select the top-N RFs which have highest weighting and considered as the spatial action information.

- *Extract Temporal Information:* For each action class and its spatial space, we propose a statistical method to extract the common sets as action temporal information. We combine both spatial and temporal information features and use them for action recognition.

3.2.2 Action recognition

In order to recognize the unknown testing action, we consider to classify the new document to different categories. In this way, the list of actions is correlated to the category of different documents. The sets of action features in each action present for the common words of each document category.

We make experiments by use the 3D MOCAP database for performance validation. The semantic action features extracted could provide very high accuracy in the action recognition, which demonstrate the efficiency of our method in extracting discriminate semantic action features. With few training samples, our experiment shows that the features extracted by this method achieves high accuracy in recognizing actions and comparable to others state-of-art methods. The further description can be found in chapter 5.

We compare our proposed EDPSS to this AESAF approach by using the same data set. From 3D MOCAP database HDM05 [27], we collect sixteen different actions as show in the Table 3.1. Each action has from thirteen to fifty-one samples. We randomly sampled two-thirds of the action sequences for training and used the remaining one-thirds for testing. We repeated the experiment 10 times and averaged the results.

Table 3.1: List experiment actions

No.	Action name	*NoS	No.	Action name	*NoS
0	clap5Reps	17	8	jumpingJack1Reps	52
1	depositFloorR	32	9	kickLFront1Reps	29
2	elbowToKnee1RepsLelbowStart	27	10	lieDownFloor	20
3	grabFloorR	16	11	punchLFront1Reps	30
4	hitRHandHead	13	12	rotateArmsBothBackward1Reps	16
5	hopBothLegs1hops	36	13	runOnPlaceStartAir2StepsLStart	15
6	jogLeftCircle4StepsRstart	17	14	shuffle2StepsLStart	13
7	jumpDown	14	15	sitDownChair	20

(*)NoS is the Number of Samples.

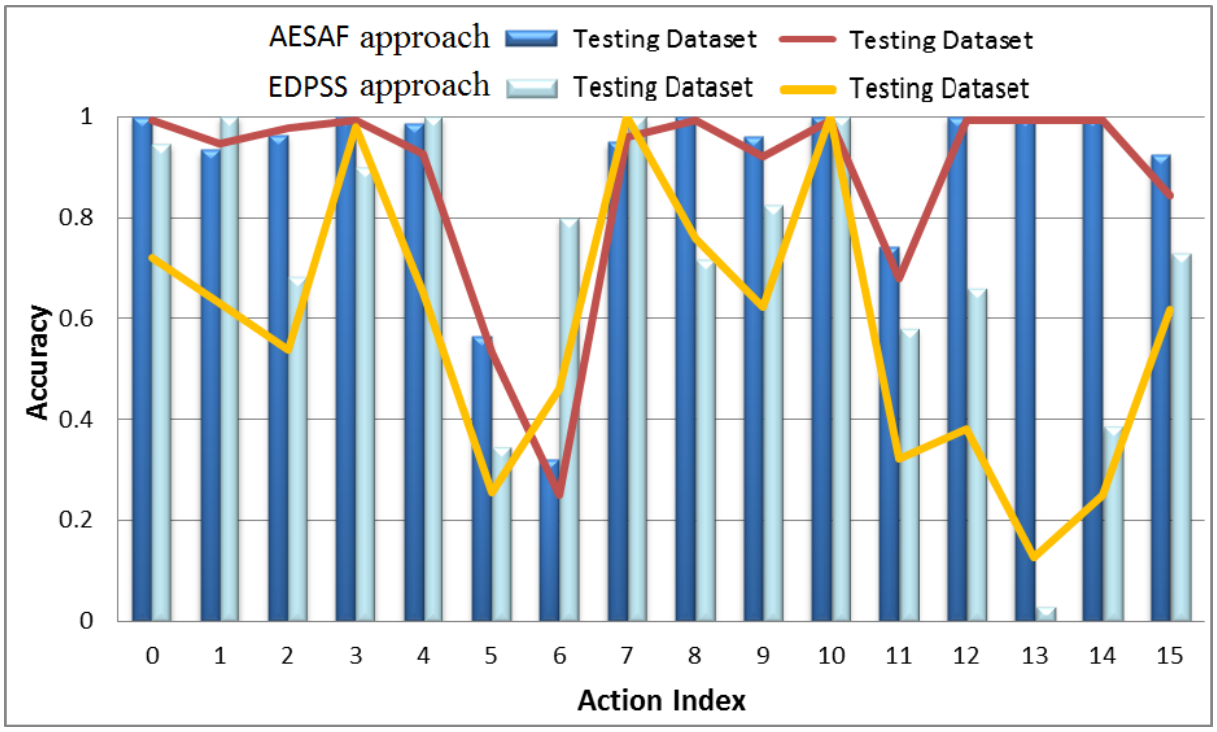


Figure 3.4: Comparing different actions recognition accuracy between AECAC to EDPSS methods.

The results of comparing different actions between two methods are showed in the Figure 3.4. It shows that the accuracy from AESAF approach is higher than EDPSS approach. The method AESAF approach gets 89.69 percent for training data-set and 87.51 percent for testing, while the EDPSS approach only gets 72.54 percent of training data-set and 58.17 percent for testing. Although EDPSS approach performs data normalization by spatially aligning the skeletons, the quantized histogram of the skeletons still includes more personal characteristics than the semantically defined relational features. This definition confuses between different patterns, which reduces the action recognition. By using the semantic definition, the proposed method AESAF approach is very robust to the different actors' skeletons and gives the higher accuracy recognition. It shows that the AESAF approach is better than EDPSS approach. We describe the detail of this comparison in chapter 5 and [33].

Summary, we give the explanations on the main work flow of the features extraction for action recognition process in this chapter. We briefly explain the difficulties at each stage of the process and highlight our motivation of the proposed methods. The detailed explanation of these methods can be found in latter corresponding chapters.

Chapter 4

Extraction of Discriminate Patterns from Skeleton Sequences

In this chapter, we extract the discriminate patterns as local features and utilize a statistical approach in text classification to recognize actions. In text classification, documents are presented as vectors where each component is associated to a particular word from the code book. In this approach, we use the beyond TF-IDF weighting method [34] to extract discrimination patterns which obtain a set of characteristics that remain relatively constant to separate different categories. This weighting defines the importance of words in representing a specific categories of documents. It not only reduces the number of feature dimension compared to the original 3D sequence of skeletons, but also reduces the viewing time of browsing, bandwidth, and computational requirement of retrieval.

The structure of this chapter is organized as follows. We describe the extraction and refinement of discriminate patterns in section 4.1. From skeleton sequences, we first automatically extract the key-frames, then group them into different patterns as unit actions. After using a statistical metric to evaluate the discriminate capability of each pattern, we further define the bag of reliable patterns as local features for action recognition. The action recognition is described in section 4.2. Section 4.3 presents our experimental results. We conclude the chapter and explain some future works in section 4.4.

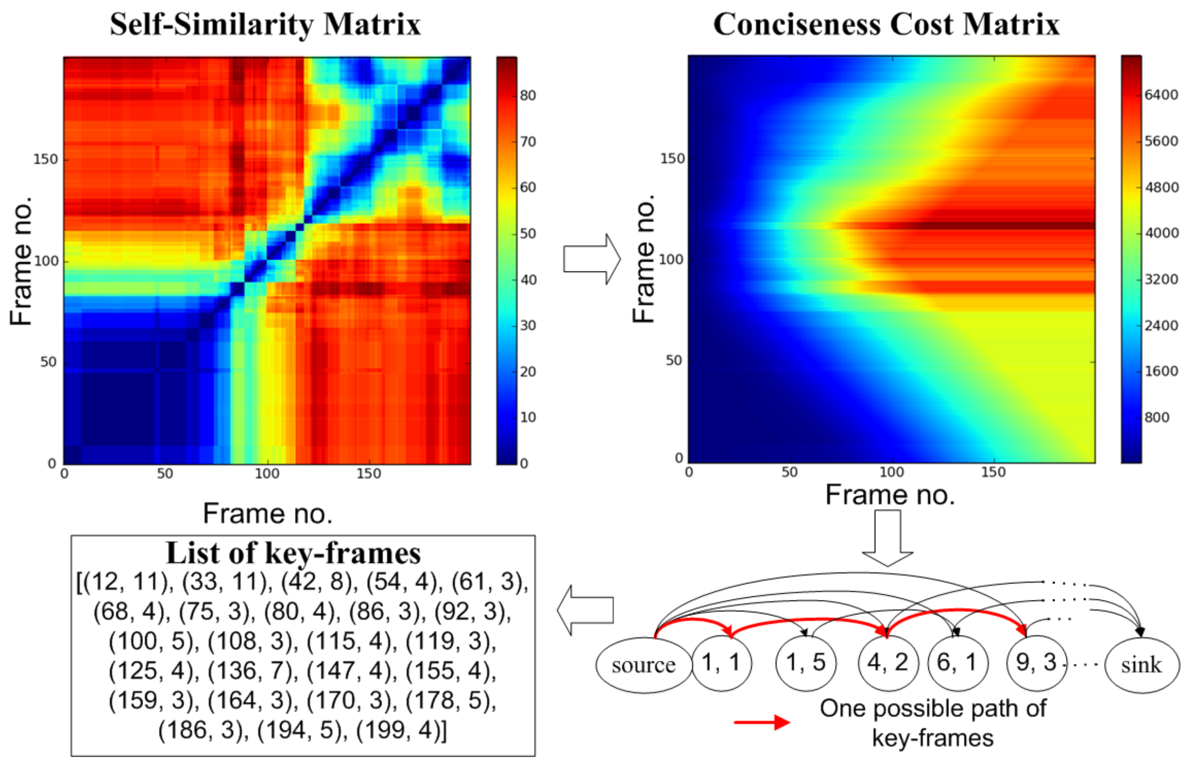
4.1 Extraction of Discriminate Patterns

In this section, we describe the extraction and refinement of discriminate patterns. From skeleton sequences, we first automatically extract the key-frames then label them as different patterns for unit actions. We further use a statistical metric to evaluate the discriminate capability of each pattern, and define the bag of reliable patterns for representing each sequence. For a testing sequence of frames, we convert it into the sequence of nearest patterns. We then calculate the spatial and temporal values for testing, and evaluate the confidence of this sequence belong to a specific action category.

4.1.1 Key-frames Extraction

Given an action sequence, we first divide it into a series of short temporal segments. From each segment, we choose one frame as its representative state, which is named as a

"Key-frame". In this chapter, we use the automatic key-frames selection method proposed by Huang [35] to summarize 3D video sequences. More specifically, the key-frames are extracted by using a graph-based method. We regard each key frame, denoted by both its index and the neighborhood size it represents, as one node in a graph. Edges are only inserted between two nodes that could be neighboring key-frames. The weight of each edge is evaluated by the difference between the key-frames of its two nodes. Since we expect a set of key-frames that have maximized mutual distances, we can intuitively use the shortest path algorithm to find the optimal solution. The whole process is illustrated in Figure 4.1.



[(12, 11), (33, 11), (42, 8), (54, 4), (61, 3), (68, 4), (75, 3), (80, 4), (86, 3), (92, 3), (100, 5), (108, 3), (115, 4), (119, 3), (125, 4), (136, 7), (147, 4), (155, 4), (159, 3), (164, 3), (170, 3), (178, 5), (186, 3), (194, 5), (199, 4)]

Figure 4.1: Illustration of automatic key-frames selection

As mentioned in section 3.1.1, we use the shape histogram to represent the 3D skeleton data. The similarity between these two histograms is computed by using the χ^2 distance (Eq. (3.1)). Formally, this method first computes the self-similarity matrix of shape histograms between all frames of a 3D video sequence,

$$S = (s_{i,j})_{N_f \times N_f} = \{d(h_i, h_j)\}_{N_f \times N_f} \tag{4.1}$$

where N_f is the number of frames in a video.

Each possible key-frame is evaluated by a Conciseness Cost, which is defined as a weighted sum of its representative cost (denoted by its average similarity to its neighbors) and accuracy cost (currently set to 1).

$$c_{i,f_i} = \beta + (1-\beta) \sum_{k=-f_i}^{f_i} s_{i,i+k} \tag{4.2}$$

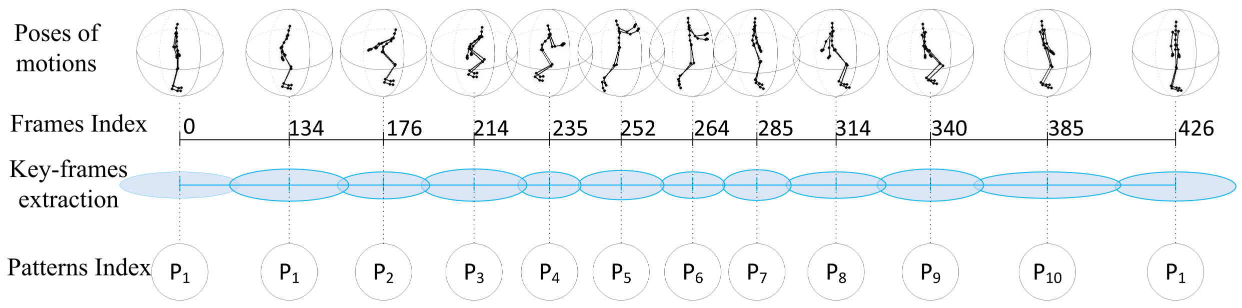


Figure 4.2: The whole sequence of 3D skeleton is divided into unit actions. Each unit action is represented from a key-frame, which includes the central frame and the period it covers. These key-frames are then identified as different patterns, according to their poses.

where β is the parameter to weight the distortion, i is the index of key-frame and f_i is the length of the local time window for all neighboring frames it represents. A Conciseness Matrix is then formed from the costs of all frames in a motion sequence under different neighbor sizes.

$$C = (c_{i,f_i})_{N_f \times neighbourSize} \tag{4.3}$$

A graph is constructed from the Conciseness Cost Matrix. Each element c_{i,f_i} in the conciseness matrix corresponds to a graph node v_{i,f_i}. Two extra nodes v_{source} and v_{sink} are added to represent the starting and ending of the sequence. Two nodes are connected with an edge, if they could appear as consecutive key-frames, i.e. by satisfying the following conditions:

$$v_{i,f_i} \; to \; v_{j,f_j} : (e_{ij} = c_{j,f_j}) \wedge (i < j) \wedge (i + f_i \geq j - f_j) \tag{4.4}$$

$$v_{source} \; to \; v_{i,f_i} : (e_{source,i} = c_{i,f_i}) \wedge (i - f_i = 1) \tag{4.5}$$

$$v_{i,f_i} \; to \; v_{sink} : (e_{i,sink} = 0) \wedge (i + f_i = N_f) \tag{4.6}$$

which require any two neighboring nodes to be temporally overlapped. A list of key-frames is then created by finding the shortest path on the graph from v_{source} and v_{sink}. Each node on the shortest path represents a key-frame. We extract a list of key-frames for each video in the database as:

$$\mathcal{K} = \{\kappa(i, f_i)\}_{i=1}^{m} \tag{4.7}$$

where $\kappa(i, f_i)$ is a key-frame and m is the number of key-frames extracted from that video. In Figure 4.2, the blue circles, in key-frames extraction step, describe the neighborhood that each key-frame represents.

4.1.2 Patterns Definition

We suppose that each action consists of several unit actions. From extracted key-frames, we need to further group them into patterns as unit actions. For each pattern, we specify an acceptance threshold, which means that any key-frames that have a smaller distance than this threshold to a pattern will be recognized as one sample of this pattern, e.g. in Figure 4.2, pattern P_1 includes key-frame 0, 134 and 426.

A pattern $p_i = ((h_i, f_i), \theta_i)$ includes all key-frame $\kappa(j, f_j)$, if it satisfies $d(h_i, h_j) \leq \theta_i$. In this work, the patterns acceptance threshold θ_i is calculated as

$$\theta_i = \max\left(d\left(h_i, h_{i-f_i}\right), d\left(h_i, h_{i+f_i}\right)\right) \tag{4.8}$$

where $d(*)$ being the distance measure from Eq. (3.1). From all lists of key-frames, we create a bag of patterns P, defined as:

$$P = \{p_i\}_{i=1}^{N_p} = \{((h_i, f_i), \theta_i)\}_{i=1}^{N_p} \tag{4.9}$$

where N_p is the total number of patterns.

An action sequence is defined as a collection of patterns [1]:

$$\mathcal{S} = \{p_i\}_{i=1}^{N_{ps}} \tag{4.10}$$

where N_{ps} is the number of patterns in $\mathcal{S}$.

We convert all the lists of key-frames to sequences of correlated patterns. C_k is defined as a set of sequences:

$$C_k = \{\mathcal{S}_i\}_{i=1}^{N_{seq}(k)}, \quad k = 1, 2, ..., N_{cat} \tag{4.11}$$

where $N_{\text{seq}}(k)$ is the number of sequences in category C_k, and N_{cat} is the number of categories.

Although the pattern has been defined, it is not sure to be suitable for classifying the action. We include the classification information contained in the training data to filter out those discriminative patterns. As describing in section 3.1.2, we evaluate the discriminative capability of all patterns in classifying categories.

4.1.3 Increasing the discriminative patterns values

Key-frames extracted from all action sequences are able to be grouped into three different sets of patterns: P^{D}, P^{S} and P^{C}. The P^{D} includes the discriminative patterns which only appear mostly in the sequences of one action category. The P^{S} is the set of the sample patterns which appear only in one action sequence. The P^{C} are the common patterns which appear in most of action categories. Accurate and robust action recognition not only depends on the quality of the discriminative patterns in P^{D}, but also on their amount. In order to collect more discriminative patterns, we keep the present discriminative patterns, but consider the possibility of converting patterns in P^{S} and P^{C} into discriminative patterns P^{D}. We increase the window size of key-frames correlated to P^{S}, so that they become more generic and could be shared by sequences from the set of action samples. At the same time, we decrease the window size of key-frames in P^{C} to make the new patterns more action specific. We present this process in the Figure 4.3.

[1]Note that a sequence is not a set, but a naive collection of patterns, which allows the existence of duplicated patterns

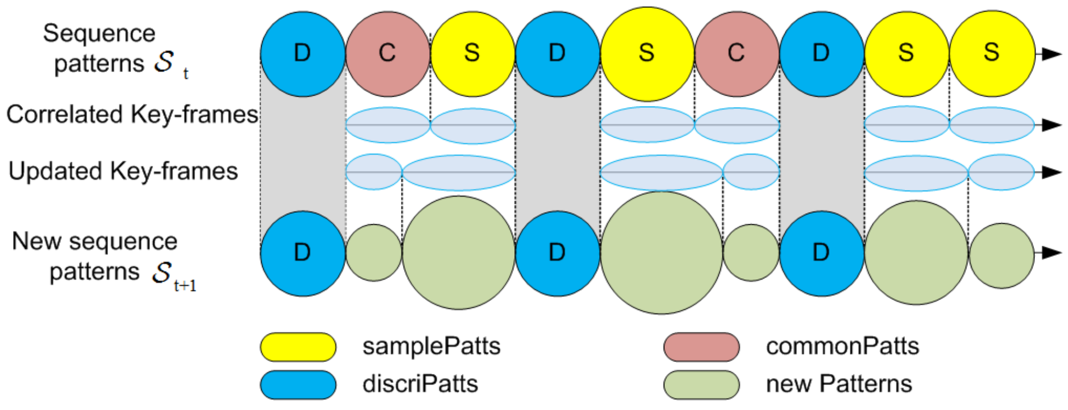


Figure 4.3: Increasing the discriminative patterns values

While keeping the present P^{D}, we update the key-frames correlated to P^{S} and P^{C} until no more key-frames updated. We check every continuous pair of patterns (p_i, p_{i+1}) in $P^{\text{C}} \times P^{\text{S}}$, $P^{\text{S}} \times P^{\text{C}}$ and $P^{\text{S}} \times P^{\text{S}}$. We increase the key-frame window size correlated to P^{S} while decrease the key-frame window size correlated to P^{C}. For pairs in $P^{\text{S}} \times P^{\text{S}}$, we increase one pattern and decrease the other. The processing steps are given in Table 4.1.

Table 4.1: Algorithm for the expansion process of sample patterns.

```
$incSize = 1$
Repeat
    Create all sequences of patterns $\mathcal{S}_j$ from present sequences of key-frames $\mathcal{K}_j$
    Extract $P^{\text{D}}$, $P^{\text{S}}$, $P^{\text{C}}$ from $\mathcal{S}_j$
    For each pair $(p_i, p_{i+1})$ in sequence $\mathcal{S}_j$:
        Get the correlated key-frames $\kappa(i, f_i), \kappa(i+1, f_{i+1})$
        If $(p_i = P^C)$ $and$ $(p_{i+1} = P^S)$ then
            Update $(\kappa(i - incSize, f_i - incSize), \kappa(i - incSize, f_{i+1} + incSize))$
        If $(p_i = P^S)$ $and$ $(p_{i+1} = P^C)$ then
            Update $(\kappa(i + incSize, f_i + incSize), \kappa(i + incSize, f_{i+1} - incSize))$
        If $(p_i = P^S)$ $and$ $(p_{i+1} = P^S)$ then
            Update $(\kappa(i + incSize, f_i + incSize), \kappa(i + incSize, f_{i+1} - incSize))$
    $incSize = incSize + 1$
Until no key-frames updated.
```

4.2 Action Recognization

Given a testing sequence, we first convert it to a sequence of discriminative patterns $\mathcal{S}$. We use the extracted discriminative patterns to calculate the spatial and temporal confidence of this testing sequence in belonging to a certain action class.

Spatial confidence: We define a weight for evaluating the confidence of a sequence belong to a specific action C_k, given the fact that it contains pattern p_i , as:

$$conf\left(\mathcal{S} \in C_k | p_i\right) = TF\left(p_i \in \mathcal{S}\right) \cdot maxStr\left(p_i \in C_k\right) \tag{4.12}$$

where

$$TF(p_i \in \mathcal{S}) = \frac{1}{N_p} \sum_{j=1}^{N_p} \delta_{p_i,p_{\mathrm{j}}} \tag{4.13}$$

It in fact defines the frequency of pattern p_i in the sequence $\mathcal{S}$, which evaluates the importance of p_i in the sequence. The $\delta_{p_i,p_{\mathrm{j}}}$ is the Kronecker delta function, which is defined as $\delta_{p_i,p_{\mathrm{j}}} = \begin{cases} 1 & p_i = p_j, \\ 0 & p_i \neq p_j \end{cases}$

From Eq. (4.12), the spatial confidence of a sequence $\mathcal{S}$ belong to a specific action C_k is calculated by summing up the support of all discrimination patterns:

$$spatialConf\left(\mathcal{S} \in C_k\right) = \sum_{i=1}^{N_p} conf\left(\mathcal{S} \in C_k | p_i\right) \tag{4.14}$$

Temporal confidence: For each action category k, we create the directed graph G_k. The graph nodes are the discriminative patterns in this category. The edge is from node N_i to N_j if node N_i appears earlier than N_j in the training sequences. The temporal confidence of a sequence $\mathcal{S}$ belongs to a specific action C_k is calculated as:

$$temporalConf\left(S \in C_k\right) = \frac{1}{\left|Edges\left(G_k\right)\right|} \sum_{N_i, N_j \in Nodes(G_k)} edgeValue\left(N_i, N_j\right) \tag{4.15}$$

where $edgeValue\left(N_i, N_j\right) = 1$ if there exists a path from N_i to N_j in G_k

From Eqs. (4.14) and (4.15), we calculate the confidence of classifying a testing sequence S to action class C_k as:

$$conf\left(\mathcal{S} \in C_k\right) = spatialConf\left(\mathcal{S} \in C_k\right) + temporalConf\left(\mathcal{S} \in C_k\right) \tag{4.16}$$

If $conf\left(\mathcal{S} \in C_k\right)$ gets the highest value in all action classes $\{C_j, j = 1, 2, ..., N_{cat}\}$, $\mathcal{S}$ is classified to the action class C_k.

4.3 Experimental Results

Motion capture systems, ranging from marker-based system to the recent Kinect, can be used to reconstruct the action of moving subjects by measuring the 3D skeletons. The skeleton in 3D videos is a collection of accurate 3D positions of body parts, which is very helpful to extract the hidden action features. We present some experimental results of using discriminative patterns extracted for action recognition. We take 25 actions from HDM05 motion capture database [27] as showed in the Table 4.2. Each action has twelve 3D videos samples. We randomly sampled two-thirds of the action sequences for training and used the remaining one-thirds for testing. We repeated the experiment 10 times and report the averaged results.

The average confusion matrices on testing and training data-sets are shown in Figure 4.4. The accuracy is 95.03 for training data-set and 80.05 for testing data-set. For the training data-set, the strong confusion occurs between action *clap*1*Reps* (No.17) and other actions. We consider that it is due to its short length, which is from 38 to 44

Table 4.2: List of experiment actions

No.	Name of Actions	No.	Name of Actions
0	kickLFront1Reps	13	standUpKneelToStand
1	lieDownFloor	14	throwSittingHighR
2	punchLFront1Reps	15	turnLeft
3	rotateArmsBothBackward1Reps	16	turnRight
4	runOnPlaceStartAir2StepsLStart	17	clap1Reps
5	shuffle2StepsLStart	18	depositFloorR
6	walk2StepsLstart	19	elbowToKnee1RepsLelbowStart
7	sitDownChair	20	grabFloorR
8	skier1RepsLstart	21	hitRHandHead
9	sneak2StepsLStart	22	hopBothLegs1hops
10	cartwheelLHandStart1Reps	23	jogLeftCircle4StepsRstart
11	squat1Reps	24	jumpDown
12	staircaseDown3Rstart		

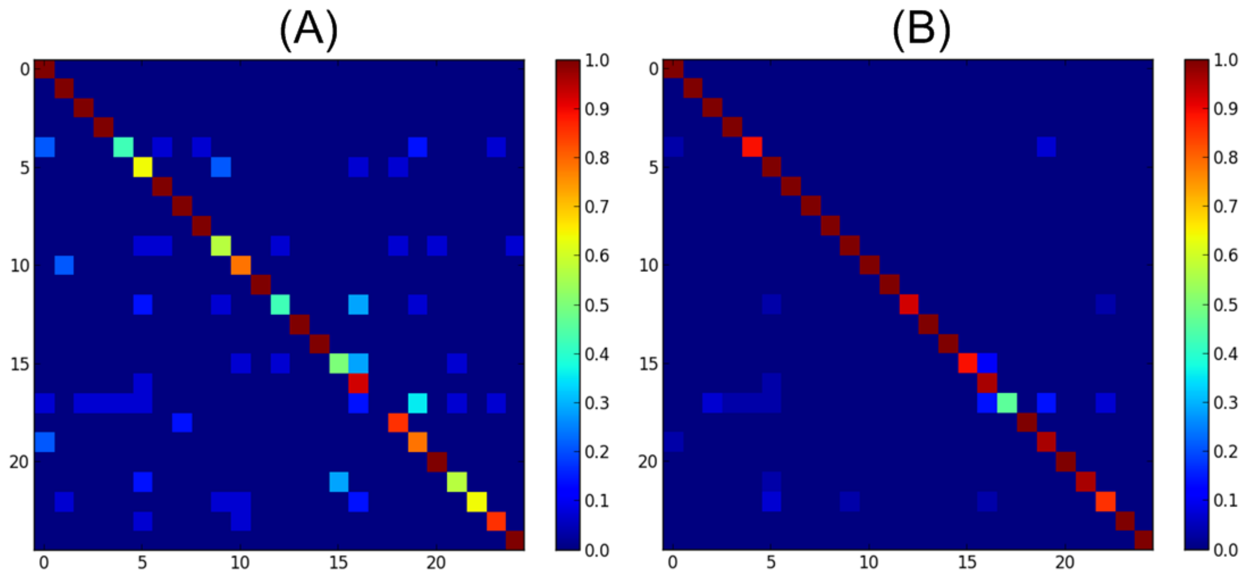


Figure 4.4: Confusion matrix for testing (A) and training (B) data sets

frames. With the same $neighborSize$ as in Eq. (4.3) for being applied to all actions, the number of key-frames for this action is limited. As a result, there are very few discriminate patterns extracted for this action. For the testing data-set, there are some confusions between similar actions between $turnLeft$(No.15) and $turnRight$ (No.16) or $hitRHandHead$ (No.21) and $turnLeft$ (No.15). These actions share similar skeleton positions. Therefore, the number of overlapped patterns between them are high, which reduces the classification accuracy. A possible improvement is thus to further explore the mutual difference between pairs of these similar actions, which is left as one of our future work.

We evaluate the overall classification rate of categories depend on the $conf$ in Eq. (3.3). The contribution of a pattern to a given action class is evaluated by whether the strength in Eq.(4.16) is positive nor not. We calculate the ROC curve with True Positive Rate (TPR) and False Positive Rate (FPR) under different confidence interval of each category C_k:

$$TPR_{C_\mathrm{k}} = \frac{N_\mathrm{k}^\mathrm{TP}}{N_\mathrm{k}^\mathrm{AP}} \qquad FPR_{C_k} = \frac{N_k^\mathrm{FP}}{N_k^\mathrm{TN}} \tag{4.17}$$

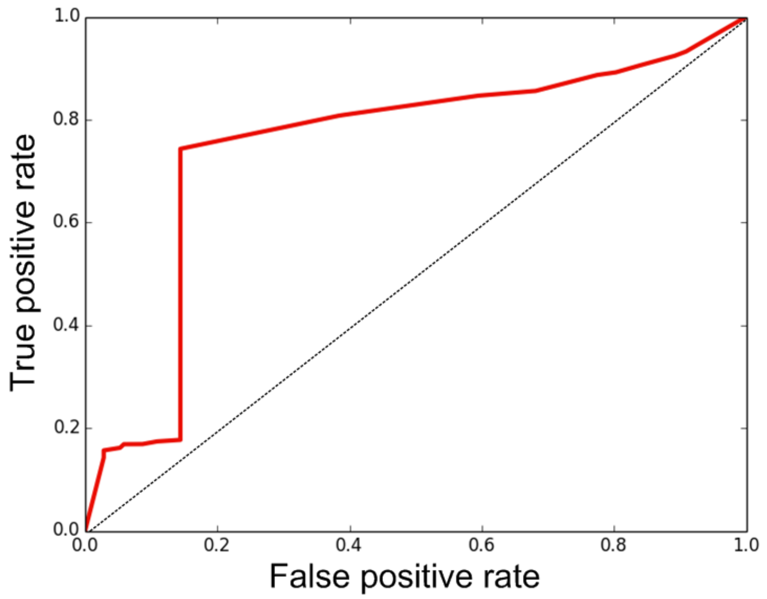


Figure 4.5: The ROC curve of the true positive rate with respect to false positive rate under various $conf$.

In the above equation, Number of True Positive $\mathrm{N}_\mathrm{k}^\mathrm{TP}$ is the number of patterns p in sequence $\mathcal{S}$ satisfy $(\mathcal{S} \in C_\mathrm{k}) \ \wedge \ (Str\,(p \in C_\mathrm{k}) > 0)$. Number of All Positive N_k^AP is the number of patterns p that satisfy $Str\,(p \in C_\mathrm{k}) > 0$. Number of False Positive N_k^FP is the number of patterns p satisfy $(Str\,(p \in C_\mathrm{k}) > 0) \ \wedge \ (Str\,(p \in C_\mathrm{i}) > 0, \ i \neq \mathrm{k})$, and N_k^TN (Number of True Negative) is the number of patterns p satisfy: $(p \in C_\mathrm{k}) \ \wedge$ $(p \in C_i, \ i \neq \mathrm{k}\ )$. TPR evaluates the average positive discriminative power, while FPR evaluates the average mutual ambiguity of patterns. Hence, a higher TPR with a lower FPR implies a better discriminative capability of extracted patterns.

The average ROC curve is shown in Figure 4.5, which is calculated from Eq. (4.17) for all categories. In this experiment, we set $conf$ from 0.5 to 2.5. As expected, there is a trade-off between a high classification rate and a low false alarm rate. From the fact that the curve is higher than the random guess line and getting better when $conf$ is greater than 1.1, we confirm that our method is effective in extracting discriminative patterns for classifying the actions.

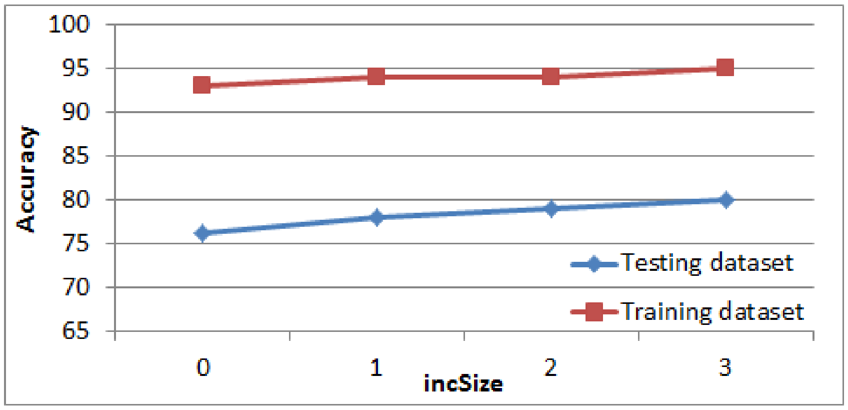


Figure 4.6: The accuracy has been improved on both testing and training datasets, by increasing the amount of discriminative patterns with pattern expansion.

In Figure 4.6, we show the overall classification accuracy while increasing the $incSize$ in the algorithm in section 3.5. In this experiment, we set the maximum of $neighbourSize = 4$ in Eq. (4.3). While $incSize$ increasing, the rate of discriminative patterns is higher which makes the accuracy higher. When $incSize$ increase from 0 to 3, we have the accuracy increases from 93.11 to 95.03 percent on the training dataset and from 76.29 to 80.05 percent on the testing dataset. We present the details of parameter setups, sample database and source codes on our website [36].

4.4 Conclusion

This chapter introduces a method to extract discriminate patterns as local features to classify skeleton sequences in human action recognition. Based on skeleton histogram, we extract key-frames from 3D videos and define patterns as local features. We use classification concept in information retrieval, i.e., the beyond TF-IDF method in document analysis, to estimate the importance of discriminate patterns for a specific categorization. We also propose an algorithm for further increasing the number of discriminate patterns. Experimental results indicate that the action recognition model using these discriminate patterns gives high accuracy around 80% on the testing data-set.

Beside of the advantages, it still lefts some considering issues, e.g. 1) It is hard to set accurately the centers of shape histograms at the skeletons in the real applications because of the noises and the recording equipment's conditions; 2) The semantic meaning of extracted features could be ambiguous when we use the distance measure in this method because it does not specify the role for different body parts, e.g. the semantic of patterns "kick left leg" and "kick right leg" are similar because those histograms are not specific which bins are for "left leg", which ones are for "right leg". The semantic annotation approach is required to solve those issues.

Chapter 5

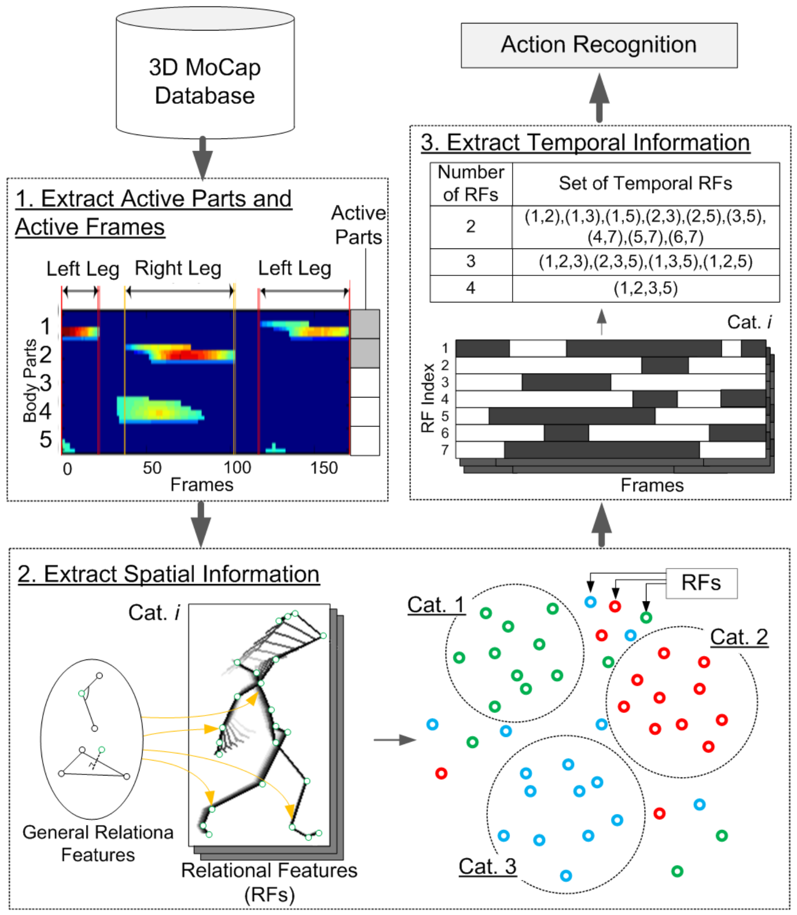
Automatic Extraction of Semantic Action Features

In this chapter, we automatically extract the common sets for each action class from 3D MOCAP data. We propose the semantic annotation approach of the human motion capture data and use the Relational Reature concept to automatically extract a set of action features. For each action class, we propose a statistical method to extract the common sets. The knowledge extracted is used to recognize the action. In this work, instead of manually defining relational features manually as the universal set of action features then use statistical method to pick up the common sets as temporal information. We extract the set of action features automatically based on the velocity feature of body joints. We considered this set as action spatial information. We combine both spatial and temporal processes to extract the action knowledge and use them for action recognition. The whole process can be divided into three main steps as showed in Figure 5.1.

1. Extraction of Active Parts and Active Frames: From 3D MOCAP data, for each action category, we calculate the velocity of body joints and select the most active parts (as sets of joints) which have high average velocities. Base on these active parts, we can extract the Active-Frames for each sample in the database.

2. Extraction of Spatial Information: In this approach, we consider the General Relational Features (GRFs), which express the geometric relationship among the set of some 3D points, are the action units. We apply them to different set of body joints and extract the common action features on Active-Frames in the database. The sets of action features in each action are consider as words of each document category. We use the text retrieval method beyond TF-IDF [34] to calculate the words weighting. We select the top-N words which have highest weighting. These action features are considered as the spatial action information.

3. Extraction of Temporal Action Information: For each action class and its spatial information, we propose a statistical method to extract the common sets as action temporal information. The knowledge extracted is used to recognize the action.



Number of RFs	Set of Temporal RFs
2	(1,2),(1,3),(1,5),(2,3),(2,5),(3,5), (4,7),(5,7),(6,7)
3	(1,2,3),(2,3,5),(1,3,5),(1,2,5)
4	(1,2,3,5)

Figure 5.1: Flow chart of the proposed approach.

The structure of this chapter is given as following: In section 5.1, from the 3D MOCAP data, we propose a method to extract the active parts and active-frames by using velocity feature for each action category. In section 5.2, we extract the spatial action information by using the relational features concept. We apply them to different sets of body joints to extract the common action features on the active frames. In section 5.3, we extract the temporal information and present the action recognition method. We do experiments in action recognize with these extracted features and have some discussions in section 5.4. The conclusion is presented in section 5.5.

5.1 Automatic Extraction of Active Parts and Active Frames

With the great advance of technology, today's motion capturing systems can track specialized markers and map the moving pose into 3D space. In sequences of moving 3D joints, the action properties are expressed through active body parts and active frames which we will extract in this section. We first define the body part from skeleton joints, then calculate the vector velocity of body parts to extract the active parts for each action category and active frames for each sample.

5.1.1 Skeleton and Body Joints

All 3D position joints of every pose create a large space of computation in the 3D MOCAP data. We first denote the actions database D contain all action categories as:

$$D = \{Action_l\}_{l=1}^{N^{ACT}} \tag{5.1}$$

where, N^{ACT} is the number of different actions. The set of samples in each action category is denoted as:

$$Action_l = \{s_{l,i}\}_{i=1}^{NS_l} \tag{5.2}$$

where, NS_l is the number of samples in $Action_l$. The 3D skeleton is a set of all joints:

$$Joints = \{j_i\}_{i=1}^{NJ} \tag{5.3}$$

where, NJ is the number of joints. We divide the set of all joints into five parts: (1) Left leg, (2) Right leg, (3) Left hand, (4) Right hand and (5) Body. The detail of parts and their joints are showed in Figure 2.3.

5.1.2 Velocity of Body Joints and Parts

We denote $Pos_f\,(j_i)$ is the position of joint j_i in three dimension at the frame f. The velocity of joint j_i from frame f_{k-1} to f_k is defined as:

$$v_{f_k}\,(j_i) = \frac{d\left(Pos_{f_k}\,(j_i)\,, Pos_{f_{k-1}}\,(j_i)\right)}{f_k - f_{k-1}} \tag{5.4}$$

where, $d(x, y)$ calculates the distance between two 3D points (x, y). The velocity of the all joints and frames of action sample s is a matrix:

$$V_s = \left[v_{f_k}\,(j_i)\right]_{j_i=[1,NJ]}^{f_k=[1,NF(s)]} \tag{5.5}$$

where, $NF(s)$ is the number of frames in sample s.

In order to extract the important parts for each action category, we calculate the average velocity for each part. The average velocity of $Part_p$ $(1 \leq p \leq 5)$ in $Action_l$ is calculated as:

$$\bar{v}_{Action_l}^{Part_p} = \frac{1}{NS_l} \sum_{u=1}^{NS_l} \sum_{j_i \in Part_p} \sum_{f_k=1}^{NF\left(s_{l,u}\right)} \left(v_{f_k}\,(j_i)\right) \tag{5.6}$$

where, $NF(s_{l,u})$ is the number of frames in sample $s_{l,u}$. The average velocity vector for all parts of $Action_l$ is:

$$\bar{v}_{Action_l} = \left[\bar{v}_{Action_l}^{Part_p}\right]_{p=1}^{5} \tag{5.7}$$

5.1.3 Extract Active Parts

In order to extract the important parts in each action sample (or category), we define the Action Templates which are the five dimensional boolean vectors. Each template expresses the active (value 1) and inactive (value 0) parts during each action sample (or category). We denote the set of all templates as:

$$T = \{Template_k\}_{k=1}^{NT} \tag{5.8}$$

where, NT is the number of templates, and

$$Template_k = [v_i]_{i=1}^{5}, v_i = \{0, 1\} \tag{5.9}$$

We extract the active body parts by finding the best fit template from set T as:

- We get the maximum value max in parts average velocity vector $\bar{v}$,
- The $Template_k$ is chosen from set T, if it satisfies: $(Template_k\,[i] = 1) \wedge (\bar{v}_i \geq max \cdot \theta)$, where $1 \leq i \leq 5$ and θ is the accept threshold (e.g. $\theta = 0.8$).

Our fixed template presents the active parts for each action category. We denote the set of active parts in $Action_l$ as:

$$ActiveParts\,(Action_l) = \{Part_k\}_{k=1}^{NP_l} \tag{5.10}$$

where, NP_l is the number of active parts of action $Action_l$.

5.1.4 Extract Active Frames

In order to extract the active frames for each sample, we consider our velocity matrix as an image and convert it to binary image by using threshold Otsu method [37]. The columns contain value 1s are considered as candidate frames and value 0s as inactive frames. The active frames are collected if candidate frames have value 1s in the joints index row which belong to the active parts. This calculation of active frames from velocity matrix is showed in Figure 5.3.

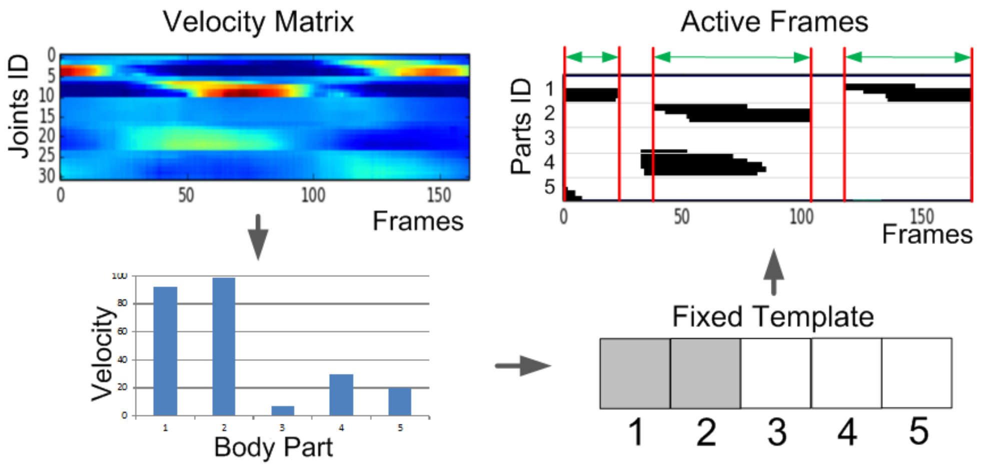


Figure 5.2: Calculation of Active Frames from Velocity Matrix.

5.2 Extract Action Spatial Information

Instead of using the raw skeleton sequences as features, we first convert them into a sequence of binary codes for representing the occurrence of unit actions, namely relational features, which provides semantically meaningful features and allows the application of many deterministic methods. The concept of relational features was first introduced by Muller and Roder [16]. Relational features describe the boolean geometric relations between specified body points of a pose or between short sequences of poses, and show how these features induce a temporal segmentation of motion capture data streams. The set of Relational Features is created by a small number of Generic Relational Features (GRFs) which encode certain point constellations in 3D space and time. The four points can also be picked in various meaningful ways for representing different unit actions, and result in different GRFs as following:

• $\mathrm{F}_{plane} = \mathrm{F}^{(\mathrm{j}_1,\mathrm{j}_2,\mathrm{j}_3;\mathrm{j}_4)}_{\theta,\mathrm{plane}}$ assumes the value one iff point j_4 has a signed distance greater than $\theta \in \mathbb{R}$ from the oriented plane spanned by the points j_1, j_2 and j_3.

• $\mathrm{F}_{nplane} = \mathrm{F}^{(\mathrm{j}_1,\mathrm{j}_2,\mathrm{j}_3;\mathrm{j}_4)}_{\theta,n\mathrm{plane}}$ is similar to $\mathrm{F}_{\mathrm{plane}}$, but the plane in terms of a normal vector given by points j_1, j_2, and passes through the anchor point j_3. The Hesse normal form yields the signed distance of point j_4.

• $\mathrm{F}_{angle} = \mathrm{F}^{(\mathrm{j}_1,\mathrm{j}_2,\mathrm{j}_3;\mathrm{j}_4)}_{\theta,\mathrm{angle}}$ assumes the value one iff the angle between the directed segments determined by line (j_1, j_2) and line (j_3, j_4) is within the threshold range $\theta \in [0, \pi]$.

• $\mathrm{F}_{nmove} = \mathrm{F}^{(\mathrm{j}_1,\mathrm{j}_2,\mathrm{j}_3;\mathrm{j}_4)}_{\theta,\mathrm{nmove}}$ considers the velocity of point j_4 relative to point j_3 and assumes the value one iff the component of this velocity in the direction determined the line segment from point j_1 to j_2 is above θ.

• $\mathrm{F}_{move} = \mathrm{F}^{(\mathrm{j}_1,\mathrm{j}_2,\mathrm{j}_3;\mathrm{j}_4)}_{\theta,\mathrm{move}}$ is similar semantics to F_{nmove}, but the direction is given by the normal vector of the oriented plane spanned by points j_1,j_2 and j_3.

Each of these features maps a given pose to the set of (0, 1) and depends on a set of joints, denoted by $j_1, j_2, ...$ to create the set of relational features. The relational feature is a function stream f frames that only assumes the values zero and one:

$$F : P \to \{0, 1\}^f \tag{5.11}$$

Regarding GRFs as the letters in the alphabet, we treat the set of RFs as the bag of words created from these letters. The set of relational features is created by applying the set of GRFs to special poses and joints. The joints in RFs (except in F_{angle}) can be divided into two parts: observer-joint (j_4) and based-joints (j_1,j_2,j_3). In order to control the number of RFs, the observer-joints are selected from active parts joints and based-joints are from inactive parts joints. For F_{angle}, all the joints are selected from active parts joints. For each action sample, we extract all active relational features which get value 1 in the active frames. These RFs are consider as words in a document and action categories contain same kind of action documents. We use the beyond TF-IDF [34] as described in section 3.1.2 to calculate the words weighting. We select the set of top-N words, which have highest weighting from all categories, is considered as action spatial information:

$$\mathbf{\Phi} = \{F_i\}_{i=1}^{N^{RF}} \tag{5.12}$$

where N^{RF} is the number of relational features. From now, we call relational features from extracted spatial information as action features. We select some top RFs which have high confident from actions: *walk2StepsLstart*(4), *walkOnPlace2StepsLStart*(5), *turnLeft*(18), *turnRight*(19), *depositHighR*(21), *jumpingJackReps*(30), are showed in the Table 5.1. In relational feature index column (RF_{ID}), $F_{x,y}$ denotes for action x and RF index y.

5.3 Extract Action Temporal Information

In order to extract the logical order from action spatial information for each action category, we use the idea of Apriori algorithm which learns the association rules by identifying

Table 5.1: The top 10 candidate extracted RFs from action *walkRightCircle4StepsRstart*.

No.	*GRF*	j_1	j_2	j_3	j_4
1	F_{NPlane}	rclavicle	rwrist	rwrist	lradius
2	F_{NPlane}	head	rradius	rradius	lwrist
3	F_{NPlane}	head	rwrist	rwrist	lhand
4	F_{Plane}	rtibia	rfoot	rradius	lradius
5	F_{Plane}	lowerneck	upperneck	head	lradius
6	F_{NMove}	rfemur	lfemur	lfemur	lhumerus
7	F_{NMove}	rtibia	rwrist	rwrist	lradius
8	F_{NMove}	rhipjoint	rclavicle	rclavicle	lwrist
9	F_{Move}	lfemur	ltibia	thorax	lclavicle
10	F_{Move}	lfemur	ltibia	rhumerus	lclavicle

the frequent item sets that appear sufficiently often in the database. We consider the association dependency between sets of relational features in a more explicit way. Instead of start counting from individual items, we calculate the frequency of sets of items in the database and extend them to larger item sets when they appear sufficiently often in database. We first define the sequence and sub-sequence of action signals which are used to calculate the confidence of relational features sets. We use the Apriori-like algorithm to calculate the frequency of common relational features sets and define the action knowledge.

5.3.1 Sequence and sub-sequence of action signals

The sequence of action signals under a set of m $(m \geq 2)$ action features takes $2^m - 1$ possible values:

$$S_m^{ACTIVE} = \{(0, ..., 1), (0, ..., 1, 0), ..., (1, ..., 1)\} \tag{5.13}$$

For every set of m action features F_{u_1}, F_{u_2},...,F_{u_m}, $u_a \neq u_b; a, b \leq m; u_a, u_b \in \overline{1, N^{\text{RF}}}$, we define the $i-$th sequence of active signals as:

$$p_i = \left\{(k_{i,1}, N^{\text{F}}_{i,1}), (k_{i,2}, N^{\text{F}}_{i,2}), ..., (k_{i,N_i^{\text{Seg}}}, N^{\text{F}}_{i,N_i^{\text{Seg}}})\right\} \tag{5.14}$$

where $k_{i,t} \in S_m^{ACTIVE}$; $N^{\text{F}}_{i,t}$ is the number of continuous frames get value $k_{i,t}$; $N_{\text{i}}^{\text{Seg}}$ is the number of segments in sequence p_i. For the similar measure process, we first define q_i as a sub-sequence of p_i:

$$q_i = \left\{(k_{i,t_1}, N^{\text{F}}_{i,t_1}), (k_{i,t_2}, N^{\text{F}}_{i,t_2}), ..., (k_{i,t_h}, N^{\text{F}}_{i,t_h})\right\} \tag{5.15}$$

where h is the number of segments in the sub-sequence and $1 \leq h \leq N_{\text{i}}^{\text{Seg}}$; t_l is the segment index of the sequence p_i and satisfy $t_l \leq t_{l+1} - 1, l = 1, 2, ..., h-1$.

5.3.2 Common action features sets

We use the Apriori-like algorithm to calculate the frequency of common action features. In the first pass, we count the support of sets of two action features and determine which of them are large, i.e. have minimum support. In each subsequent pass, we start with a seed set of item sets found to be large in the previous pass. We use this seed set for generating new potentially item sets of three action features, called candidate item sets, and calculate the actual support for these candidate item sets during the pass over the data. At the end of the pass, we determine which of the candidate item sets are actually large, and they become the seed for the next pass. This process continues until no new large item sets are found. Each action is a set of N^{RF} sequences. We start with the set of every two action features (F_{u_1}, F_{u_2}). With q_i and q_j are the sequences of (F_{u_1}, F_{u_2}) for different samples, we check every pair of (q_i, q_j) from all examples of this action class. We define a similar process to find a pair of sub-sequences $q_i \in p_i$, $q_j \in p_j$ which satisfies $|q_i| = |q_j|$ and the optimal warping is achieved by $(\hat{q}_i, \hat{q}_j)$ which maximizes the warping distance, i.e,

$$(\hat{q}_i, \hat{q}_j) = \underset{(q_i, q_j)}{\arg\max}\, S_{p_i,p_j|q_i,q_j} \tag{5.16}$$

$$S_{p_i,p_j} = S_{p_i,p_j|\hat{q}_i,\hat{q}_j} \tag{5.17}$$

where $S_{p_i,p_j|q_i,q_j}$ stands for the wrapped similarity between p_i and p_j under wrapping style (q_i, q_j). And $S(p_i, p_j)$ is the maximized similarity between p_i and p_j under optimal warping style $(\hat{q}_i, \hat{q}_j)$. Especially, we define

$$S_{p_i,p_j|q_i,q_j} = \frac{\sum_{t=1}^{|p_i|} (n_{i,t}^F + n_{j,t}^F)\delta_{k_{i,t},k_{j,t}}}{\sum_{t=1}^{|p_i|} n_{i,t}^F + \sum_{t=1}^{|p_j|} n_{j,t}^F} \tag{5.18}$$

where $n_{*,t}^F$ is the number of frames in segment t^{th} of sequence p_*, and $\delta_{k_{i,t},k_{j,t}} = \begin{cases} 1 & k_{i,t} = k_{j,t} \\ 0 & k_{i,t} \neq k_{j,t} \end{cases}$ being the Kronecker delta function. $S_{p_i,p_j|q_i,q_j}$ takes value from 0 to 1, and evaluates the number of matched values between sub-sequence of q_i and q_j by the total active frames in two sequences. It gets value near to 1 if the sequence q_i is the same to sequence q_j. The support of current classification results by set of (F_{u_1}, F_{u_2}) is evaluated by the average similarity between all pairs (p_i, p_j) from all examples of the action A_t:

$$R^{SUP}(F_{u_1}, F_{u_2}) = \frac{1}{\mathsf{C}_{N_{A_t}}^2} \sum_{1 \leq i < j \leq N_{A_t}} S_{p_i,p_j} \tag{5.19}$$

where N_{A_t} is the number of samples in the action class A_t. The confident between F_{u_1} and F_{u_2} is the weight decided as:

$$w_2 = \begin{cases} R^{\text{SUP}}(F_{u_1}, F_{u_2}), & R^{\text{SUP}}(F_{u_1}, F_{u_2}) \geq \theta \\ 0, & otherwise \end{cases} \tag{5.20}$$

The set of all action features which satisfy Eq.(5.19) is a seed set. We use this set for generating new potentially large item sets as candidate item sets. The sequences created from this set contain three different action features. Similarity, we use the Eq.(5.15) to (5.18) to calculate the confidence for $(F_{u_1}, F_{u_2}, F_{u_3})$. The set of all action features satisfied the confidence greater than θ becomes a seed set and is used for generating new potentially larger item sets. This process continues until no more new larger item sets are found. After extract the temporal information, we select two candidate middle sequences samples which have high confident from action *walk2StepsLstart* as showed in Table 5.2.

Table 5.2: Two candidates middle sequences selected from action *walk2StepsLstart*.

	RF_{ID}	*Sequence*				
Sample 1	$F_{4,2}$	1	1	0	1	1
	$F_{4,3}$	1	1	1	1	1
	$F_{18,8}$	1	0	0	0	1
	Frames	47	6	88	10	9

	RF_{ID}	*Sequence*			
Sample 2	$F_{4,0}$	0	1	1	1
	$F_{4,3}$	1	1	1	1
	$F_{19,4}$	1	1	0	1
	Frames	10	48	98	13

5.3.3 Temporal Information from Common Action Features

From the t-th action, where $t = \left[1, N^{ACT}\right]$, we collect all sets of (F_{u_1}, F_{u_2}), $(F_{u_1}, F_{u_2}, F_{u_3})$,... which have the confidence greater than threshold θ. For each set, we save the the average sequence presented for all samples in this action category and get the action knowledge:

$$\mathcal{K} = \{TI_t | t \in \overline{1, N^{ACT}}\} \tag{5.21}$$

where TI_t is temporal information of each action A_t:

$$TI_t = \{(Set_i, midSeq_i)\}_{i=1}^{N_{set}^{A_t}} \tag{5.22}$$

where Set_i is the set of action features which has the confidence greater than threshold θ; the average sequence $midSeq_i$ is chosen among all sample sequences set, if it is the most similar to others; and $N_{set}^{A_t}$ is the total number of sets.

5.3.4 Action Recognition

We use the action knowledge extracted to action recognition. In order to recognize the unknown testing action, we consider to classify the new document to different categories. In this way, the list of actions is correlated to the category of different documents. The sets of action features in each action present for the common words of each document category. We use the word beyond TF-IDF weighting method [34] to classify the new document to different categories. At first, we collect the words for the new document (unknown category). Each document is the action which contains N^{RF} sequences correlated to the set of N^{RF} relational features. Each word is a set of action features. For each action A_t, we collect the *"words"* as following: We calculate the confidence between the sequence from testing action and the trained average sequence. We collect all words for the new document for the action A_t:

$$E_t^{TST} = \bigcup_{i=1}^{N_{set}^{A_t}} Set_i,\ S(seq_i^{\mathrm{TST}}, midSeq_i^{TRN}) \geq \theta \tag{5.23}$$

where seq_i^{TST} and $midSeq_t^{TRN}$ are the testing sequence and trained average sequence in the unknown action for the set of action features Set_i. Then, we use the TF-IDF to calculate each word in the new document *belongs to* action A_t from the action knowledge $\mathcal{K}$. For each word w in the new document, we get its confidence weight from the action TI_t to calculate the TF. IDF is calculated as the number of action categories in the list action $\mathcal{K}$ which contains the word w.

$$C^{TST}(TI_t) = \sum_{w \in E_t^{\mathrm{TST}}} TF(w, TI_t) \times IDF(w, \mathcal{K}) \tag{5.24}$$

where,

$$TF(w, TI_t) = \frac{weight_{TI_t}(w) - \theta}{1 - \theta}$$

$$IDF(w, K) = \frac{1}{\left|\left\{TI_t \in K : w \in TI_t | t \in \overline{1, N^{ACT}}\right\}\right|}$$

where, $weight_{TI_t}(w)$ is the confidence value of the word w in action TI_t. The winner is the action index which gets the highest value from following equation:

$$L^{\mathrm{TST}} = \arg\max_t \left(C^{\mathrm{TST}}(TI_t)\right) \tag{5.25}$$

where, $t = \left[1, N^{ACT}\right]$. If $C^{TST}(TI_t)$ gets the highest value in all action categories $A_j, j = 1, 2, ..., N^{ACT}$, then the testing action takes t as the winner label L^{TST}.

5.4 Experiments

Table 5.3: List of 10 similar walking actions.

ID	Action Name
0	walk2StepsLstart
1	walk2StepsRstart
2	walk4StepsLstart
3	walkBackwards2StepsRstart
4	walkBackwards4StepsRstart
5	walkLeftCircle4StepsRstart
6	walkLeftCircle6StepsRstart
7	walkRightCircle4StepsRstart
8	walkOnPlace2StepsLStart
9	walkOnPlace2StepsRStart

We use the 3D motion capture data from HDM05 motion capture database [27] for performance validation. We collect ten similar actions as showed in Table 5.3 and thirty-one different actions as showed in Table 5.4. Each action has from thirteen to fifty-two samples and performed by five different actors. We randomly sampled 2,4,6 and 8 samples of the action sequences for training and used the remaining for testing. After calculate the confident for each RF, we select top 10 RFs for each action category to further extract the

Table 5.4: List of experiment actions.

ID	Action Name	ID	Action Name
0	kickRFront1Reps	16	throwSittingHighR
1	punchRFront1Reps	17	throwBasketball
2	rotateArmsRForward1Reps	18	turnLeft
3	shuffle2StepsLStart	19	turnRight
4	walk2StepsLstart	20	clap5Reps
5	walkOnPlace2StepsLStart	21	depositHighR
6	sitDownChair	22	depositLowR
7	sitDownFloor	23	elbowToKnee1RepsLelbowStart
8	skier1RepsLstart	24	elbowToKnee1RepsRelbowStart
9	sneak2StepsLStart	25	grabHighR
10	cartwheelLHandStart1Reps	26	grabLowR
11	squat1Reps	27	hopBothLegs2hops
12	staircaseDown3Rstart	28	jogLeftCircle4StepsRstart
13	staircaseUp3Rstart	29	jumpDown
14	standUpKneelToStand	30	jumpingJack1Reps
15	standUpLieFloor		

temporal information. We set $\theta = 0.7$ in Eq. (5.20) and the length of each sequence is less than 20 segments. The testing accuracy with the number of training samples is showed in the Figure 5.3. Our propose method can get 83.82 percent accuracy on testing data-set with only two training samples and reach to 100 percent with eight training samples.

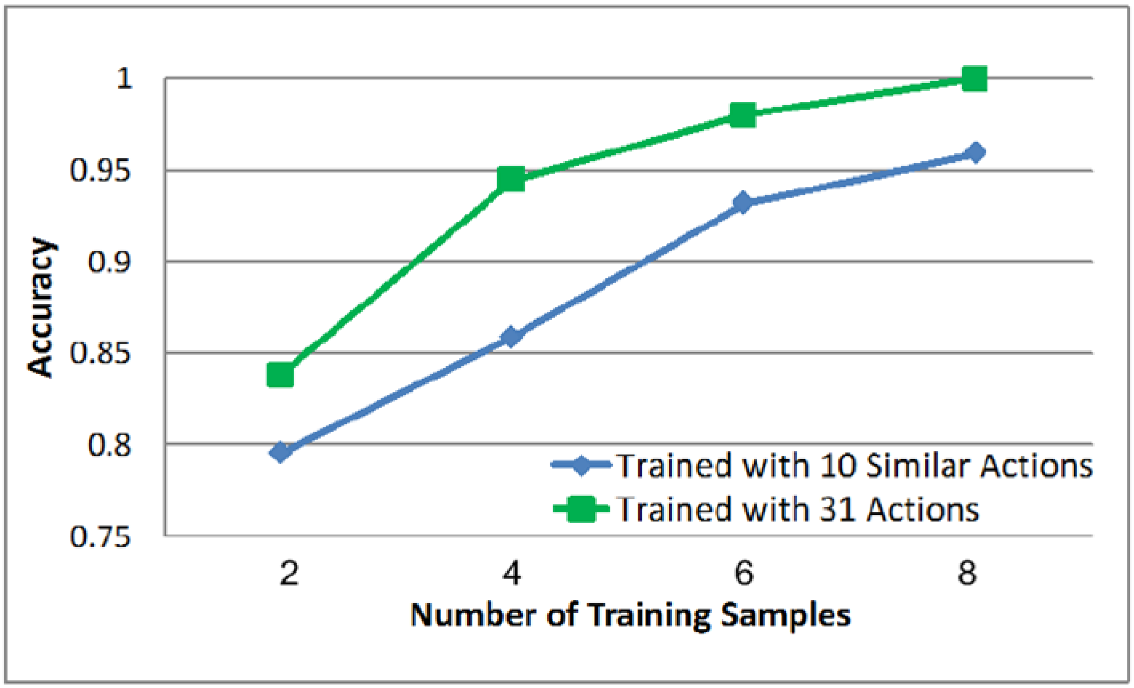


Figure 5.3: Testing accuracy with the number of training samples.

We set the threshold for the relational features in this experiment as: For F_{Move} and F_{NMove}: $\theta = 0.8hl/s$; F_{Plane}: $\theta = 0.5hl$; F_{NPlane}: $\theta = 0.2hl$ and F_{Angle}: $\theta = 110°$ where hl is the *humerus length*. We present the detail of parameter setups, sample data of extracted actions and source code on our website [36].

Table 5.5: Comparison of state-of-art methods.

Methods	Num. of Actions	Accuracy
Actionlet Ensemble [38]	5	98.13
SMIJ [39]	5	98.53
Our propose method	10	95.90
	31	100.00

We compare some present state-of-the-art methods experimented on Action MOCAP database [27] in Table 5.5. Actionlet Ensemble [38] and SMIJ [39] gets very high accuracy with 98.53 percent. However, they only test on five different actions. In our experiments, we show that our method can get high accuracy with only few training samples on the set of 31 actions.

We have the comparison between the our proposed EDPSS approach and this AESAF approach as described in Chapter 3. The results show that the action recognition accuracy from the propose AESAF method is higher than the EDPSS method. Although EDPSS approach performs data normalization by spatially aligning the skeletons, the quantized histogram of the skeletons still includes more personal characteristics than the semantically defined relational features. This definition confuses between different patterns, which reduces the action recognition. By using the semantic definition, this AESAF approach is very robust to the different actors' skeletons and gives the higher accuracy recognition.

The most recent researches in extracting features for action recognition only work for offline actions retrieval. They do not have capability for real-time processing, e.g. two actions "walking left" and "walking right" are considered as the same actions if the processing range is less than n frames (e.g. $n = 80$). Additionally, we do not have the "cutting samples" for testing or recognizing in real practical applications. So that, it requires us the real-time processing methods. In Appendix B and C, we provide some experiments to use the extracted semantic action features for the depth architecture networks. They are combined from multi-deep learning networks. We convert the extracted semantic action features into sequences of images which are used to train the multi-level networks. The depth architecture networks can learn the spatial-temporal features information for real-time action recognition. They not only can focus on the recognizing objects in detail, but also perform actions recognition in real-time.

5.5 Conclusion

In this work, we propose a method to extract the knowledge from the 3D action MOCAP data. We define the set of action features and use a statistical method to extract the common sets for different action classes. For each actions, the knowledge extracted are the sets of frequent common action features. The results show that this extracted action features are robust with the new input data, which gives high accuracy when we use them for testing on recognizing actions. The knowledge extracted from our method is not only performed automatically, but also more relevant in its semantic meaning. In the present research, we use the sets of action features both spatially and temporally. In the sets of action features, we focus on the whole action as global information, but the order of appearance and disappearance of each features was omitted. However, this local information could help us to learn and recognize actions in real-time, which will be discussed in our further works.

Chapter 6

Conclusions and Future Work

We proposed novel approaches in this thesis to automatically extract the action knowledge from 3D MOCAP data for action recognition. In particular, we contribute to two different areas dealing with various between the features level extracted, i.e., *1) Extract of Discriminate Patterns from Skeleton Sequences* and *2) Automatic Extraction of Semantic Action Features approaches.* As one common underlying concept, the propose approaches contain a retrieval component making use of extracted features.

Firstly, we extract the discriminative patterns as local features and the utilization of a statistical approach in text classification to recognize actions. In text classification, documents are presented as vectors where each component is associated to a particular word from the code book. We use weighting methods to estimate the importance of each word in the document. In this approach, we use the beyond TF-IDF weighting method to extract discrimination patterns which obtain a set of characteristics that remain relatively constant to separate different categories. This weighting defines the importance of words in representing specific categories of documents. It not only reduces the number of feature dimension compared to the original 3D sequence of skeletons, but also reduces the viewing time of browsing, bandwidth, and computational requirement of retrieval.

Secondly, we propose the semantic annotation approach of the human motion capture data and use the RF concept to automatically extract a set of action features. For each action class, we propose a statistical method to extract the common sets. The features extracted is used to recognize the action. We extract the set of action features automatically based on the velocity feature of body joints. They are considered as action spatial information. We combine both spatial and temporal processes to extract the action features and use them for action recognition. In our experiments, we use the 3D motion capture data from MOCAP database for performance validation. The results show that the semantic action features extracted could provide very high accuracy in the action recognition, which demonstrate the efficiency of our method in extracting discriminative semantic action features. Our propose methods achieve very high accuracy with only few of training samples and comparable to others state-of-art methods.

6.1 Future work

Motivated by the computers could automatically interpret the activities people perform, human motion analysis has been a highly active research area in computer vision and have some achievements. From the advantage of depth sensors cameras, 3D human body tracking has become feasible for high-level recognition tasks. In particular, the depth sensor cameras have provided robust to human skeleton tracking and 3D scenes reconstruction. Using depth images to reconstruct 3D human model has proved to simplify the task. It has removed the need for markers attached to the body in practical applications and real-time action recognition. In knowledge processing, the deep architectures can learn and recognize complicated functions that can represent high-level abstractions in human actions. It is not only can learn the spatial-temporal features from sequences of information, but also can recognize objects accurately in real-time. We can break a high-level activity into several simpler sub-activities and link them in a hierarchical model for the multiple actions detection. Combining with a 3D body model, the human motion tracking in real-time is proved to possibility which can be built from higher-level algorithms for complex actions involving interactions with other humans and objects.

From 3D MOCAP database, we get some achievements from extracting semantic action features and use them for real-time recognition in high accuracy. By using the relational feature concept, we create the bag of words then use statistic approaches to weight and extract the most common action features. These features are converted into sequences of vision data to feed the depth architecture networks for real-time action recognition. These hierarchical layers networks like the visual attention schema which can learn actions in different groups and recognize them in detail. From above achievements, human motion tracking using a 3D body model will enable the next stage and become feasible for high-level recognition tasks. To further develop our works to a real practical human tracking system, we propose three main tasks: Firstly, the present action recognition model need to be improved for learning and updating new actions more flexible; Secondly, we need the reconstruct 3D scenes (in some contexts) and human body modeling in real-time. Based on these achievements, we create human tracking system which can learn and manage the actions features to recognize actions in different contexts.

The present research achievements in computer vision and human behavior allow us to develop a real human tracking system.

We propose three main tasks on human motion tracking system as:

1. From the resent results in extracting action features from 3D MOCAP database, we develop the methods to recognize action in real-time by using the depth architecture.

2. Reconstruct 3D scenes and human body modeling to manage the objects by computers.

3. Create a tracking and human behavior analysis system by combining the features processing from real-time action recognition and the 3D scenes contexts.

Goal 1: Real-time Action Recognition

From present extracted semantic action features, we propose the methods to recognize actions in real-time. We create a depth architecture model by combining multilevel networks which can focus on the recognizing objects in detail. These networks can learn the extracted features and perform action recognition. This propose model not only can extract the semantic action features from 3D MOCAP data, but also can apply for the real-time action recognition.

Deep learning has been proposed by Geoffrey Hinton, which simulates the hierarchical structure of human brain, processes data from lower level to higher level and gradually composes more and more semantic concepts. It is composed of multiple levels of non-linear operations, such as hidden layers in neural nets and the complicated propositional formulae re-using many sub-formulae. The optimization principle that works well for Deep Belief Networks and related algorithms, based on a greedy, layer-wise, unsupervised initialization of each level in the model. Its continuation method approximates a more general optimization principle, so that a series of gradually more difficult optimization problems are solved. The optimization tasks of searching the parameter space in Deep Belief Networks have recently been proposed notable successful and beaten the state-of-the-art in certain areas.

In order to achievers above advantage, we need convert our extracted semantic features into videos data which are feed to the deep architectures networks. The spatial information from our present research is considered as the pixels in each frame. The relationship in temporal information helps grouping neighborhood pixels into objects segments. We organize spatial-temporal features as continuous sequences of frames which are feed to the deep architectures networks. Because of extracting from the clear 3D MOCAP data, our action recognition results from the deep architecture model are expected outperform in accuracy and high speed.

In order to further improve the performance and judgment of action recognition, we need to focus on density estimation to extract fuzzy logical from unsupervised learning in deep learning. This combination of tractability and flexibility allows us to tackle a variety of probabilistic tasks on high-dimensional datasets. The goal of unsupervised learning is to characterize structure in our data. This features can be used to establish summarized representations by making informed assumptions about the redundancy of unobserved data. Meanwhile, fuzzy logical consists of representation of the data under the assumption of absence information in the regions space to which we've assigned low density. This combination links these two concepts very explicitly. Allocation of probability mass throughout our space precisely quantifies our belief and obviates our assumptions about structure in the data.

Goal 2: Reconstruct 3D scenes and human body modeling

Using depth data from 3D scanning hardware, depth camera, stereo vision and structured lights techniques to reconstruct 3D moving or static objects in the scenes.

With the advantages of 3D scanning hardware, depth camera, stereo vision and structured lights techniques, we can obtain the 3D objects and the scenes for real-time reconstruction. With a low-cost moving depth camera like Microsoft Kinect we not only can capture captured depth maps into the final scene, but also can simplify reconstruct 3D human model. It has removed the need for markers attached to the body in practical applications. Depth images have advantages over intensity images. Firstly, they have good invariance against illumination changes. Secondly, they provide the 3D structure of the scene, and can significantly simplify tasks such as background subtraction, segmentation, and motion estimation.

Only use a moving depth camera and commodity graphics hardware, the recent developed KinectFusion system can reconstruct the indoor scenes accurately in real-time. The robustness of this system lies in that it fuses all of the depth data streamed from a Kinect sensor into a single global implicit surface model of the observed scene. Similar to other techniques, they first de-noise the input raw data with a bilateral filter and a multi-resolution method. Then the truncated signed distance function (TSDF) is used as the data structure for later processing. The global fusion of all depth maps is formed by the weighted average of all individual TSDFs. The resulted 3D model from KinectFusion is of reasonable quality. For further development, we not only focus on reconstruction 3D scenes, but also the real objects. We can define then separate the 3D objects model from the entire 3D scene by identifying the object-of-interest and using 2D segmentation to refine the silhouette from color images.

The researches based on depth imagery to body part detection and poses estimation have great developments. The majority work has focused on fitting a 3D human model to the scene. There are two different approaches body part detection and body pose modelling which based on their knowledge of the human structure and behavior. These systems extract the foreground, convert it to a 3D point cloud, and fit to a body model. Without constraints on the configuration of the joints, in body part detection approach, a basic 3D articulated body model can be a simple skeleton with point cloud by using body part segmentation concepts. Meanwhile, body pose modelling approach is more sophisticated, the kinematic constraints are used to limit the movement. The limb length, length ratio between different body parts, and relative body part positions are used. Moreover, limited degrees-of-freedom of different joints confine the model to a set of valid poses. Instead of using the depth image directly, some researches reconstruct a 3D surface mesh from the depth values; then fit a body model to that 3D data before calibrate the model to the human subject. A common approach for tracking body motion from 3D mesh fitting is Iterative Closest Point. Some researches inferred an articulated 2D human pose from a body silhouette extracted from a single depth image using a Pictorial Structure Model. Instead of a conventional rectangular limb model, they model each limb with a mixture of probabilistic shape templates, which showed promising improvement accuracy.

For future research on 3D human model, there are still some challenges as the requirement for a good initial pose, the iterative approaches in tracking fast movements, multiple people in the scene, occluded body parts, and higher resolution. The 3D position of a person used in activity recognition and the scene captured from a depth camera can be combined with known 3D positions in the environment. Because human actions are characteristic of individuals, there may be many aspects of the scene that still require low-level processing. Combining both types of 2D intensity and depth images for certain cases may increase our systems robustness.

Goal 3: **Tracking system and human behavior analysis**

Create a human tracking system which can manage and control the tracking people in public places. Social Signal Processing aims at developing theories and algorithms that codify how human beings behave while involved in social interactions, putting together perspectives from sociology, psychology, and computer science Method: combine the results from reconstruct 3D objects and real-time action recognition, to manage and control the objects.

Tracking systems and human behavior analysis through visual information has been a highly active research topic in the computer vision community. These systems could gain some achievements:

1. Reduce cost:

These videos can be manually monitored through video walls. Video surveillance acts as a security mechanism to monitor areas prone to issues like theft, drug trafficking, border trespassing, vandalism, fights, etc. It may also be used in home-care systems for monitoring children and old people, or patients in hospitals. The third kind of usage is for pattern analysis where, people behavior and shoppers buying behavior are collected and patterns found. If an area under surveillance has many cameras, it is tedious to monitor all of them manually. It is said that manual supervisors tend to miss some activities when they continuously monitor video walls. This led to the transition of manual video surveillance to automated video surveillance. Automation reduces man power wasted in manual monitoring and subsequent human errors, thus reducing the cost of employment, reducing the cost of storage and leading to a fool-proof monitoring.

Video analytics (see footnote 1) is used for optimizing storage as well as analyzing human behavior. Since storing all the videos requires a lot of memory space, storage can be optimized by not recording static scenes. This is done by triggering the video record sequence only when there is motion in a scene, thereby reducing cost of storage.

2. Behavior need:

In object classification, the blob in the foreground is categorized into object types. In motion tracking, an objects movement is tracked from one frame to another. These phases, i.e. motion detection, object classification and motion tracking form the building blocks of human behavior analysis. With the results obtained from these, a behavior recognition methodology can be formulated using domain specific poses and semantics. A generalized approach to human behavior recognition can be designed for research purposes. Systems which are to be used commercially are preferred to be domain specific. For example, system at a railway station for detecting suspicious activities needs to detect activities like fighting, got hurt, stealing, running, etc.

Visual cues can be used for predicting the behavior of a human being. A system can learn visual cues related to emotions by recognizing certain regions of face or body parts which identify the emotions. Temporal segmentation is a sensitive process. The correct segmentation of each and every atomic action will decide the type of activity predicted.

Bibliography

[1] Krystian M., Hirofumi U., *Action recognition with appearance-motion features and fast search trees*, Computer Vision and Image Understanding, Volume 115 Issue 3, pp.426-438, March 2011.

[2] Arikan O. and Forsyth. D. A., *Interactive motion generation from examples*, In SIGGRAPH, pp.483-490, New York, NY, USA, ACM Press, 2002.

[3] Egges A., Molet T., and Magnenat-Thalmann N., *Personalised real-time idle motion synthesis*, In Pacific Graphics, IEEE Computer Society, pp.121-130, Washington, DC, USA, 2004.

[4] Kovar L., Gleicher M., and Pighin F., *Motion graphs*, In SIGGRAPH, pp.473-482, New York, NY, USA, ACM Press, 2002.

[5] Alla S., Jessica K. H., *Construction and optimal search of interpolated motion graphs*, ACM Transactions on Graphics Journal, SIGGRAPH 2007 Proceedings, August 2007.

[6] K. Forbes and E. Fiume, *An efficient search algorithm for motion data using weighted PCA*,In Proc. 2005 ACM SIGGRAPH/Eurographics Symposium on Computer Animation, pages 6776. ACM Press, 2005.

[7] Kovar L., Gleicher M., *Automated extraction and parameterization of motions in large data sets*, ACM Transactions on Graphics 23, 3 (2004), 559568. SIGGRAPH, 2004.

[8] Muller M, *Information Retrieval for Music and Motion*, ISBN: 978-3-540-74047-6, Springer, 2007.

[9] Kruger B., Tautges J., Weber A., and Zinke A., *Fast local and global similarity searches in large motion capture databases*, 2010 ACM SIGGRAPH, pp.1-10. Eurographics Association, 2010.

[10] Kilner J., Guillemaut J.Y., Hilton A., *3D action matching with key-pose detection.* Computer Vision Workshops (ICCV Workshops), 2009 IEEE 12th International Conference, Kyoto, pp.1-8, 2009.

[11] Baak A., Mller M., Seidel H.P., *An Efficient Algorithm for Keyframe-based Motion Retrieval in the Presence of Temporal Deformations*, The 1st ACM SIGMM Int. Conf. on Multimedia Information Retrieval, 2008.

[12] Sun X., Chen M.-Y., and Hauptmann A., *Action Recognition via Local Descriptors and Holistic Features*, IEEE - CVPR for Human Communicative Behaviour Analysis, Miami Beach, Florida, USA, June 25, 2009.

[13] Ankerst M., Kastenmller G., Kriegel H. P., and Seidl T., *3D shape histograms for similarity search and classification in spatial databases*, Advances in Spatial Databases, 6th International Symposium, SSD99, 1651, pp.207-228, 1999.

[14] Huang P., Hilton A. and Starck J., *Shape Similarity for 3D Video Sequences of People*, In International Journal of Computer Vision (IJCV) special issue on 3D Object Retrieval, Volume 89, Issue 2-3, pp.362-381, September 2010.

[15] Arikan O., Forsyth D. A., OBrien J. F., *Motion synthesis from annotations*, ACM Trans. Graph. 22, 3, 402408, 2003.

[16] Muller M., Roder T., and Clausen M., *Efficient content-based retrieval of motion capture data*, ACM Transactions on Graphics (TOG), 24(3):677685, 2005.

[17] Muller M., Roder T., *Motion templates for automatic classification and retrieval of motion capture data*, In Proceedings of the ACM SIGGRAPH/Eurographics SCA, pages 137146. ACM Press, 2006.

[18] Gao Y., Ma L., Chen Y., and Liu J., *Content-based human motion retrieval with automatic transition*, Advances in C.G., volume 4035 of L.N. in Computer Science, pages 360-371. Springer-Verlag, 2006.

[19] Gao Y., Ma L., Liu J., Wu X., and Chen Z., *An effcient algorithm for content-based human motion retrieval, In Technologies for E-Learning and Digital Entertainment*, volume 3942, pages 970979. Springer-Verlag, 2006.

[20] Baak A., Muller M., and Seidel H., *An efficient algorithm for keyframe-based motion retrieval in the presence of temporal deformations*, In Proceedings of the 1st ACM SIGMM International Conference on MIR, pages 451 458, 2008.

[21] W. Li, Z. Zhang, Z. Liu, *Action recognition based on bag of 3d points*, in: Human Communicative Behavior Analysis Workshop (in conjuntion with CVPR), 2010.

[22] Jamie Shotton, Andrew Fitzgibbon, Mat Cook, Toby Sharp, Mark Finocchio, Richard Moore, Alex Kipman, Andrew Blake, *Real-time human pose recognition in parts from a single depth image*, in: CVPR, 2011.

[23] Vicon, *Vicon motion capture system. http://www.vicon.com/*, 2012.

[24] PhaseSpace, *PhaseSpace motion capture. http://www.phasespace.com*, 2012.

[25] Meinard Muller, Tido Roder, Michael Clausen, *Documentation Mocap Database HDM05*, ISSN 1610-8892, (2007).

[26] CMU, *Carnegie-Mellon Mocap Database. http://mocap.cs.cmu.edu*, 2003.

[27] *Motion Capture Database HDM05*, `http://www.mpi-inf.mpg.de/resources/HDM05/`, (2012).

[28] The TUM Kitchen Data Set of Everyday Manipulation Activities. http://kitchendata.cs.tum.edu

[29] The MSR Action3D. http://research.microsoft.com/en-us/um/people/zliu/ActionRecoRsrc/

[30] Adam G. Kirk, James F. OBrien, and David A. Forsyth, *Skeletal parameter estimation from optical motion capture data*, In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 782788, 2005.

[31] RichardM.Murray, Zexiang Li, and S. Shankar Sastry, *A Mathematical Introduction to Robotic Manipulation*, CRC Press, 1994.

[32] Wilson E.B., *Probable Inference, the Law of Succession and Statistical Inference*, Journal of the American Statistical Association, 22, pp.209-212, 1927.

[33] Tran Thang Thanh, Fan Chen, Kazunori Kotani and Bac Le, *Automatic Extraction of Common Action Characteristics*, The 12^{th} IEEE International Symposium on Signal Processing and Information Technology (ISSPIT), 2012.

[34] Pascal S., Mineau G. W., *Beyond TFIDF Weighting for Text Categorization in the Vector Space Model*, International Joint Conference on AI. Scotland: UK, pp. 1130-1135, 2005.

[35] Huang P., Hilton A. and Starck J., *Automatic 3D Video Summarization: Key Frame Extraction from Self-Similarity*, 4th International Symposium on 3D Data Processing, pp.71-78, Atlanta, GA, USA, June 2008.

[36] Supplemental Materials, http://www.jaist.ac.jp/∼s1020210/FI.htm, 2013.

[37] Otsu N., *A threshold selection method from gray-level histogram*, IEEE Trans. Sys. Man., Cyber. (1) 62-66, 1979.

[38] Jiang W., Liu Z., Wu Y., Yuan J., *Mining actionlet ensemble for action recognition with depth cameras*, In CVPR'12, pages 1290-1297,2012.

[39] Offii F., Chaudhry R., Kurillo G., Vidal R., *Sequence of the most informative joints (SMIJ): A new representation for human skeletal action recognition*, J. Vis Commun. Image R., 2013.

[40] G. Hinton, Learning multiple layers of representation, Trends in Cognitive Sciences, 11, pp. 428-434, 2007.

[41] J. Hawkins, D. George, Hierarchical temporal memory: Concepts, theory and terminology, White Paper, Numenta Inc., 2006.

[42] V. Le, Y. Zou, Y. Yeung, N. Andrew, Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis, in IEEE-CVPR, pp. 3361-3368, 2011.

[43] J. Jang, Y. Park, and H. Suh, Empirical Evaluation on Deep Learning of Depth Feature for Human Activity Recognition, in ICONIP 2013, Part III, LNCS 8228, pp. 576-583, 2013.

Publications

[1] Tran Thang Thanh, Fan Chen, Kazunori Kotani, Bac Le, *Extraction of Discriminative Patterns from Skeleton Sequences for Accurate Action Recognition.* Fundamenta Informaticae 130 (2014) 1-15, DOI 10.3233/FI-2014-890, IOS Press, 2014. (*Journal*)

[2] Tran Thang Thanh, Fan Chen, Kazunori Kotani, Bac Le, *Action Recognition by using Depth Architecture Networks on Relational Features*, The 21th IEEE International Conference on Image Processing (ICIP), 2014 (Accepted).

[3] Tran Thang Thanh, Fan Chen, Kazunori Kotani, Bac Le, *Automatic Extraction of Semantic Action Features*, In The 9th International Conference on Signal-Image Technology and Internet-Based Systems, 978-1-4799-3211-5/13 IEEE, DOI 10.1109/SITIS.2013.35, 2013.

[4] Tran Thang Thanh, Fan Chen, Kazunori Kotani, Bac Le, *An Apriori-like algorithm for automatic extraction of the common action characteristics*, In Visual Communications and Image Processing, 2013.

[5] Tran Thang Thanh, Fan Chen, Kazunori Kotani, Bac Le, *Automatic Extraction of the Common Action Features*, The IEEE RIVF International Conference on Computing and Communication Technologies, 2013.

[6] Tran Thang Thanh, Fan Chen, Kazunori Kotani, Bac Le, *Extraction of Discriminative Patterns from Skeleton Sequences for Human Action Recognition.* The IEEE RIVF International Conference on Computing and Communication Technologies, 978-1-4673-0309-5/12 IEEE, 2012.

[7] Tran Thang Thanh, Fan Chen, Kazunori Kotani, Bac Le, *Automatic Extraction of Action Features from 3D Action MOCAP Database*, in the 1th Japan Advanced Institute of Science and Technology (JAIST) Poster Challenge, 10/2013.

[8] Tran Thang Thanh, Fan Chen, Kazunori Kotani and Bac Le, *Automatic Extraction of Common Action Characteristics*, The 12th IEEE International Symposium on Signal Processing and Information Technology (ISSPIT), 2012.

Appendix A

Relational Feature

A.1 Relational Feature Design

The generic feature $\mathrm{F}_{\theta,\mathrm{plane}}^{(\mathrm{j}_1,\mathrm{j}_2,\mathrm{j}_3;\mathrm{j}_4)}$

Given four joints $p_1, ..., p_4 \in \mathbb{R}^3$, the signed distance of p_4 to the plane $\langle p_1, p_2, p_3 \rangle$ can be computed as follows. First, determine a unit normal vector

$$n\left(p_1, p_2, p_3\right) := \frac{\left(p_1 - p_3\right) \times \left(p_2 - p_3\right)}{\left\|\left(p_1 - p_3\right) \times \left(p_2 - p_3\right)\right\|}, \tag{A.1}$$

The orientation of which is given by the right-hand-rule. Then evaluate the Hesse normal form, which yields the signed distance of p_4 as:

$$d_{plane}\left(p_1, p_2, p_3; p_4\right) := \langle n\left(p_1, p_2, p_3\right), p_4 - p_3 \rangle, \tag{A.2}$$

where positive values of $d(p_1, p_2, p_3; p_4)$ indicate that p_4 lies in the open half space into which the normal $n(p_1, p_2, p_3)$ joints. Next, we apply the threshold function

$$H_\theta(x) = \begin{cases} 1 & if\ x \geq \theta \\ 0 & otherwise, \end{cases} \tag{A.3}$$

where $\theta, x \in \mathbb{R}$. For a given pose $P \in \mathcal{P}$, we can then compute the feature value as

$$\mathrm{F}_{\theta,\mathrm{plane}}^{(\mathrm{j}_1,\mathrm{j}_2,\mathrm{j}_3;\mathrm{j}_4)}\left(P\right) = H_\theta\left(d_{plane}\left(P^{j1}, P^{j2}, P^{j3}; P^{j4},\right)\right). \tag{A.4}$$

In other words, F_{plane} assumes the value one iff joint j_4 has a signed distance of at least θ from the oriented plane spanned by the joints j_1, j_2 and j_3.

The generic feature $\mathrm{F}_{\theta,nplane}^{(\mathrm{j}_1,\mathrm{j}_2,\mathrm{j}_3;\mathrm{j}_4)}$

A similar test is performed by $\mathrm{F}_{\theta,nplane}^{(\mathrm{j}_1,\mathrm{j}_2,\mathrm{j}_3;\mathrm{j}_4)}$, but here we define the plane directly in terms of a normal vector and an anchor joint. For $p_1, ..., p_4 \in \mathbb{R}^3$, let

$$n\left(p_1, p_2\right) := \frac{p_2 - p_1}{\left\|p_2 - p_1\right\|}. \tag{A.5}$$

Then the Hesse normal form yields the signed distance of p_4 as

$$d_{nplane}\left(p_1, p_2, p_3; p_4\right) := \langle n\left(p_1, p_2\right), p_4 - p_3\rangle, \tag{A.6}$$

where the plane passes through the anchor joint p_3. This allows us to define the feature value for a given pose $P \in \mathcal{P}$ as

$$\mathrm{F}^{(\mathrm{j_1,j_2,j_3;j_4})}_{\theta,nplane}\left(P\right) = H_\theta\left(d_{nplane}\left(P^{j1}, P^{j2}, P^{j3}; P^{j4}\right)\right). \tag{A.7}$$

The generic feature $\mathrm{F}^{(\mathrm{j_1,j_2,j_3;j_4})}_{\theta,\mathrm{angle}}$

This feature assumes the value one iff the angle enclosed between the directed segments determined by (j_2, j_1) and (j_3, j_4) is within the angle range $\theta \subseteq [0, \pi]$. Given four joints $p_1, ..., p_4 \in \mathbb{R}^3$, we start by computing the unit vectors $n(p_1, p_2)$ and $n(p_3, p_4)$ pointing in the directions of the two segments. The angle enclosed between $n(p_1, p_2)$ and $n(p_3, p_4)$ is denoted as $(p_1, p_2; p_3, p_4) \in [0, \pi)$; more precisely, we mean the smaller of the enclosed angles. This angle can now be computed as

$$\alpha\left(p_1, p_2; p_3, p_4\right) := \arccos\langle n\left(p_1, p_2\right), n\left(p_3, p_4\right)\rangle. \tag{A.8}$$

Note that $n(p_1, p_2)$ joints from p_1 to p_2 and $n(p_3, p_4)$ joints from p_3 to p_4. In case $p_1 = p_3 = p$, this definition allows for the interpretation that there is a common joint between the two segments joining at p, both of which point away from p. The feature value for a pose $P \in \mathcal{P}$ is defined as

$$\mathrm{F}^{(\mathrm{j_1,j_2;j_3,j_4})}_{\theta,angle}\left(P\right) = \chi_\theta\left(\alpha\left(P^{j1}, P^{j2}; P^{j3}, P^{j4}\right)\right), \tag{A.9}$$

where $\chi_\theta : [0, \pi] \rightarrow \{0, 1\}$ is the characteristic function of the angle range $\theta \subseteq [0, \pi]$.

There are three reasons why we recompute angles from 3D joint positions instead of directly using the joint angle information from the animated skeleton. First, we aim at keeping our features independent of specific data formats, focusing exclusively on 3D joint coordinates. Second, the joint angles that are specified in typical MOCAP files are usually offset against our features' joint angles by an angle that may vary between MOCAP files; for example, the knee angle for a stretched knee is denoted as 0° in many MOCAP files, whereas the above computations would yield 180°. Finally, our method also works for two bones or two joint-driven line segments that are not connected by a common joint; for example, it could make sense to measure the angle of the arm against the direction of the spine.

The generic feature $\mathrm{F}^{(\mathrm{j_1})}_{\theta,fast}$

The three remaining generic features operate on velocity data that is approximated from the 3D joint trajectories of the input motion. Given a MOCAP data stream $D : [1 : T] \rightarrow \mathcal{P}$, we focus on a single joint $j \in \mathcal{J}$ and consider its 3D trajectory D^j , which we think of as a sequence $(p(1), ..., p(T))$ in $\mathbb{R}^3$. From this trajectory, we can compute approximate velocity vectors by taking the "discrete derivative"

$$v(t) = \begin{cases} \frac{p(t+1)-p(t)}{\Delta t} & if\ 1 \leq t \leq T \\ v(T-1) & t = T, \end{cases} \tag{A.10}$$

where Δt is the inverse of the sampling rate, for example $\Delta t = \frac{1}{120}s$. Note that we duplicate the last velocity vector to maintain the length of the original trajectory.

MOCAP data may contain significant high-frequency noise components, which typically leads to very noisy velocity data since the discrete derivative corresponds to a high-pass filter.

Such spatial noise can be due to calibration and tracking errors, to inadequate data cleanup such as unresolved marker occlusions and marker confusions, and to skeletal fitting errors. We make the simple assumption that the three dimensions of the noise components in the 3D velocity vectors are uncorrelated and the underlying stochastic process is stationary, so suitable averaging over several velocity vectors should allow for some of the noise to cancel out. To this end, we apply a moving average filter to the 3D velocity vectors.

We start by extending the sequence of velocity vectors using symmetric padding before frame 1 and after frame T, yielding the infinite sequence $\tilde{v} : \mathbb{Z} \to \mathbb{R}^3$ defined by

$$\tilde{v}(t) = \begin{cases} \tilde{v}(1-t) & if\ t \leq 1 \\ \tilde{v}(2T+1-t) & if\ T > 1 \\ v(t) & otherwise. \end{cases} \tag{A.11}$$

The convolution filter C^h convolves its input sequence with the filter coefficients $(h_k)_{k\in[-\mathrm{K}:\mathrm{K}]}$, where convolution between a one-dimensional sequence and a sequence of 3D vectors is meant in a coordinate-wise sense. Choosing

$$h_k := \frac{e - \frac{1}{2}\left(\beta\frac{2k}{2K+1}\right)^2}{\sum_{k=-K}^{K} e - \frac{1}{2}\left(\beta\frac{2k}{2K+1}\right)^2} \tag{A.12}$$

for $k \in [-\mathrm{K} : \mathrm{K}]$, we obtain a *Gaussian lowpass filter mask* of length $2K + 1$, where the parameter corresponds to the inverse of the variance of the corresponding Gaussian distribution. We chose $\beta = 2.5$. The noise-reduced sequence is then given by

$$\overline{v}(t) := C^h[\tilde{v}](t) = (h * \tilde{v})(t) = \sum_{k=-K}^{K} h_k \tilde{v}(t-k), \tag{A.13}$$

which we evaluate for $t \in [1 : T]$. Note that this is equivalent to filtering the 3D data prior to computing the discrete derivative due to the linearity of convolution.

Writing the filtered velocity sequence corresponding to the trajectory of joint $j \in \mathcal{J}$ in operator notation as $\overline{v}[D^j]$, we then define the feature value for a pose $D(t) = P \in \mathcal{P}$ as

$$\mathrm{F}^{(\mathrm{j}_1)}_{\theta,fast}(D(t)) := H_\theta\left(\left\|\overline{v}[D^{j1}](t)\right\|\right). \tag{A.14}$$

Note that we require P to be embedded within the context of a MOCAP data stream D.

The generic feature $\mathrm{F}^{(\mathrm{j}_1,\mathrm{j}_2,\mathrm{j}_3;\mathrm{j}_4)}_{\theta,\mathrm{nmove}}$

This feature considers the velocity of joint j_4 relative to joint j_3 and assumes the value one iff the component of this velocity in the direction determined by the line segment from j_1 to j_2 is above θ. The said velocity component can also be viewed as the signed, one-dimensional velocity of joint j_4 relative to the plane determined by a normal vector (given by j_1 and j_2) and the anchor joint j_3. This was the motivation for the abbreviation "nmove", where the 'n' stands for "normal".

We compute the desired velocity component in the following way. Given four points $p_1 = D^{j_1}(t), ..., p_4 = D^{j_4}(t) \in \mathbb{R}^3$ from frame $t \in [1 : T]$ of a MOCAP data stream D, we first determine the unit normal vector $n(p_1, p_2)$, cf. (3.6). The filtered relative velocity vector

$$\bar{v}(p_3, p_4) := \bar{v}[D^{j_4} - D^{j_3}](t) \tag{A.15}$$

between joints j_3 and j_4 at time t is then projected onto $n(p_1, p_2)$ by means of the inner product, yielding the scalar velocity

$$\bar{v}(p_1, p_2, p_3; p_4) := \langle n\left(p_1, p_2\right), \bar{v}\left(p_3, p_4\right)\rangle, \tag{A.16}$$

The feature value for a pose $D(t) = P \in \mathcal{P}$ is then defined as

$$\mathrm{F}_{\theta,nmove}^{(\mathrm{j_1,j_2,j_3;j_4})}\left(D(t)\right) = H_\theta\left(\bar{v}(D^{j_1}(t), D^{j_2}(t), D^{j_3}(t); D^{j_4}(t))\right) \tag{A.17}$$

for a suitable velocity threshold θ.

The generic feature $\mathrm{F}_{\theta,\mathrm{move}}^{(\mathrm{j_1,j_2,j_3;j_4})}$

This feature has similar semantics as $\mathrm{F}_{\theta,\mathrm{move}}^{(\mathrm{j_1,j_2,j_3;j_4})}$, but the direction is given by the normal vector of the oriented plane spanned by j_1, j_2, and j_3.

We manually create 28 relational features from above six GRFs as showed in the Table A.1.

Table A.1: List some manual relational features.

ID	Type	j_1	j_2	j_3	j_4	θ_1	θ_2	Description
F1/F2	F_{nmove}	neck	rhip	lhip	rwrist	1.8 hl/s	1.3hl/s	rhand moving forwards
F3/F4	F_{nplane}	chest	neck	neck	rwrist	0.2 hl	0hl	rhand above neck
F5/F6	F_{move}	belly	chest	chest	rwrist	1.8hl/s	1.3 hl/s	rhand moving upwards
F7/F8	F_{angle}	relbow	rshoulder	relbow	rwrist	[0^0, 110^0]	[0^0, 120^0]	relbow bent
F9	F_{nplane}	lshoulder	rshoulder	lwrist	rwrist	2.5sw	2sw	hands far apart, sideways
F10	F_{move}	lwrist	rwrist	rwrist	lwrist	1.4 hl/s	1.2 hl/s	Hands approaching each other
F11/F12	F_{move}	rwrist	root	lwrist	root	1.4 hl/s	1.2 hl/s	rhand moving away from root
F13/F14	F_{fast}	rwrist				2.5 hl/s	2 hl/s	rhand fast
F15/F16	F_{plane}	root	lhip	ltoes	rankle	0.38 hl	0 hl	rfoot behind lleg
F17/F18	F_{nplane}	$(0,0,0)^T$	$(0,1,0)^T$	$(0,5,0)^T$	rankle	1.2 hl	1 hl	rfoot raised
F19	F_{nplane}	lhip	rhip	lankle	rankle	2.1hw	1.8 hw	feet far apart, sideways
F20/ F21	F_{angle}	rknee	rhip	rknee	rankle	[0^0, 110^0]	[0^0, 120^0]	rknee bent
F22/ F23	F_{angle}	neck	root	rshoulder	relbow	[25^0, 180^0]	[20^0, 180^0]	rhumerus abducted
F24/F25	F_{angle}	neck	root	rhip	rknee	[50^0, 180^0]	[45^0, 180^0]	rfemur abducted
F26	F_{plane}	rankle	neck	lankle	root	0.5 hl	0.35 hl	root behind frontal plane
F27	F_{angle}	neck	root	$(0,0,0)^T$	$(0,1,0)^T$	[70^0, 110^0]	[60^0, 120^0]	spine horizontal
F28	F_{fast}	root				2.3hl/s	2hl/s	root fast

The length unit "hl", "sw" and "hw" for "humerus length", "shoulder width" and "hip width".

A.2 Choosing Thresholds

Selecting appropriate generic features and suitable combinations of joints only determines a part of the semantics of a relational feature. The other part comes from a semantically meaningful choice of the threshold parameter θ. As an important point, note that variations of θ may completely change the meaning of a relational feature, which in turn influences the motion aspect that can be grasped by that feature. According to the different types of generic features, there are different interpretations of such threshold values θ: can be thought of as shifting the respective plane in the direction of the plane's normal vector.

- Angle ranges $\theta \subseteq [0, \pi]$ for the generic feature $F_{\theta,angle}$ may be used to define the notion of a "bent" or "stretched" joint, or they can define more general angle relations between joint-driven or absolute segments such as the spine horizontal detector.
- Velocity thresholds $\theta \in \mathbb{R}$ for the generic features $F_{\theta,fast}$, $F_{\theta,move}$, and $F_{\theta,nmove}$ tune the features' sensitivity to the speed of motions. For small values of , even minute motion details will lead to a feature value of one. For larger values of , increasingly brisk motions are required to trigger the feature.

As the last step of computing relational feature values, we typically apply the threshold function H_θ to continuous values such as distances or velocities, which may be measured in different units (inches, centimeters, or arbitrary multiples thereof) depending on the specific motion capture file. To account for such differences in scale and to make motions performed by different characters comparable, we express our distance and velocity thresholds in terms of certain constant, skeleton-defined distances, such as the length of the upper arm (humerus), which scales well with the absolute size of the skeleton. We abbreviate the resulting length unit as "hl" for "humerus length". Similarly, we also use the relative length units "sw" and "hw", standing for "shoulder width" and "hip width".

A.3 Robust Threshold

The simple quantization scheme using the threshold function H_θ as described for the generic features above is prone to strong output fluctuations if the input value fluctuates slightly around the threshold. To alleviate this problem, we employ a robust quantization strategy with two thresholds: a stronger threshold, θ_1, and a weaker threshold, θ_2.

If the actor is standing with slightly spread legs such that the right ankle is very close to the test plane, small changes of the ankle position or the hip orientation can lead to strong zero/one-fluctuations of the Boolean feature value. By introducing the weaker threshold $\theta_2 = 1.0hw$, insignificant fluctuations can be filtered out in the following way: we only let the feature value switch over from one to zero if the distance falls below θ_2. We refer to this strategy as robust threshold, in the literature it is also known as hysteresis threshold. It turns out that this heuristic suppresses undesirable zero-one fluctuations in relational feature values very effectively.

For $\theta_1 > \theta_1$, we replace the threshold function H_θ by the robust threshold operator $H^{robust}_{\theta_1\theta_2}$, which acts on a time-dependent sequence $x : [1 : T] \to \mathbb{R}$ as follows:

$$H^{robust}_{\theta_1\theta_2}[x](t) := \begin{cases} 1 & if\ x(t) \geq \theta_1 \\ 0 & if\ x(t) < \theta_2 \\ H^{robust}_{\theta_1\theta_2}[x](t-1) & if\ x(t) < \theta_1 \wedge x(t) \geq \theta_2 \end{cases} \tag{A.18}$$

where $t \in [1 : T]$ and $H^{robust}_{\theta_1\theta_2}[x](0) := H_{\theta_1}(x(1))$. A similar robust threshold operator can replace the characteristic function $\mathcal{X}_\theta$ for the case of angle ranges $\theta \subseteq [0, \pi]$.

Appendix B

Using Action Features for Real-time Action Recognition

B.1 Introduction

In this chapter, we introduce the method which uses the depth architecture networks to learn the semantic action features for real-time action recognition. After extracting the semantic action features from 3D MOCAP database, we convert them into sequences of images which are used to train the multilevel networks. These networks not only can learn the action features, but also can focus on the recognizing objects in detail and perform real-time actions recognition. The whole process is illustrated in Figure B.1.

The structure of this paper is given as following: In section B.2, we propose the depth architecture combined from multilevel networks which are trained from the extracted features to perform real-time action recognition. After that, we describe the action recognition method in section B.3. Some experiments are presented in section B.4. The conclusion and further works are presented in section B.5.

B.2 Depth Architecture Network

The deep learning has been proposed by Geoffrey Hinton [40], which simulates the hierarchical structure of human brain, processes data from lower level to higher levels and gradually composes more and more semantic concepts in human actions. It is composed of multiple levels of non-linear operations, such as hidden layers in neural nets and the complicated propositional formula re-using many sub-formula. Its continuation method approximates a more general optimization principle, so that a series of gradually more difficult optimization problems are solved. The optimization tasks of searching the parameter space in Deep Belief Networks have recently been proposed notable successful and beaten the state-of-the-art in certain areas. It is not only can learn the spatial-temporal features from sequences of information, but also can recognize objects accurately in real-time.

In this work, we organize them into the depth architectures to specify each network functions. One structure example of two-level depth architecture networks is showed in the Figure B.1(B). We use the extracted semantic action features to train these networks for real-time action recognition. Firstly, the extracted semantic features from previous section are converted into sequences of images. An image is not only contain the spatial

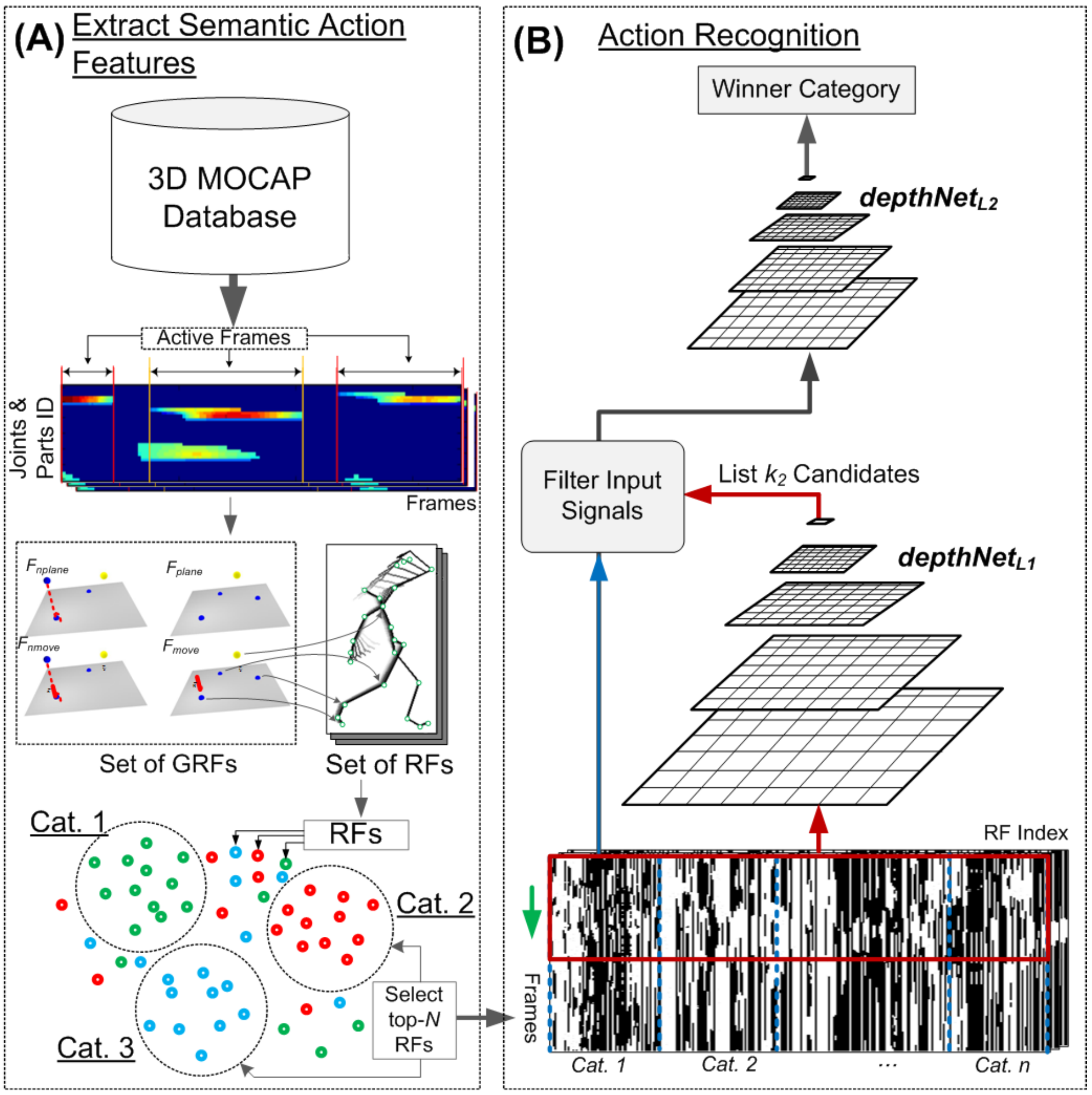


Figure B.1: Flow chart of the proposed approach.

information as the order of RFs, but also have the temporal information from the window slicing on those results. These images are feed to the deep architecture; which contains n multilevel networks: $depthNet_{L_i}$, $i = \overline{1,n}$. Trained by N_{Cat} categories, the $depthNet_{L_1}$ gives top k_1 ($k_1 < N_{Cat}$) candidates from the input image. The $depthNet_{L_2}$, which is trained from sequences of images containing k_1 categories as a time, gives top k_2 ($k_2 < k_1$) candidates to the higher networks. The top network gets information from its lower networks and the input image to recognize the final winner category.

In this work, we use the Hierarchical Temporal Memory (HTM) [41] to create the multilevel networks. Each network have multi-layers, the lowest layer get the input signal from the sequences of images. The nodes in each layer will combine input signals, groups them by the Markov chains networks then send output to the higher nodes. The top node covers all the signals from lower layers and classify the input to different categories. Layer by layer, the network learns sequences of input information. By this way, it learns the signals from each image as spatial features and the order of those appearance as temporal features. One structure example five layers of HTM deep learning network is showed in the Figure B.2.

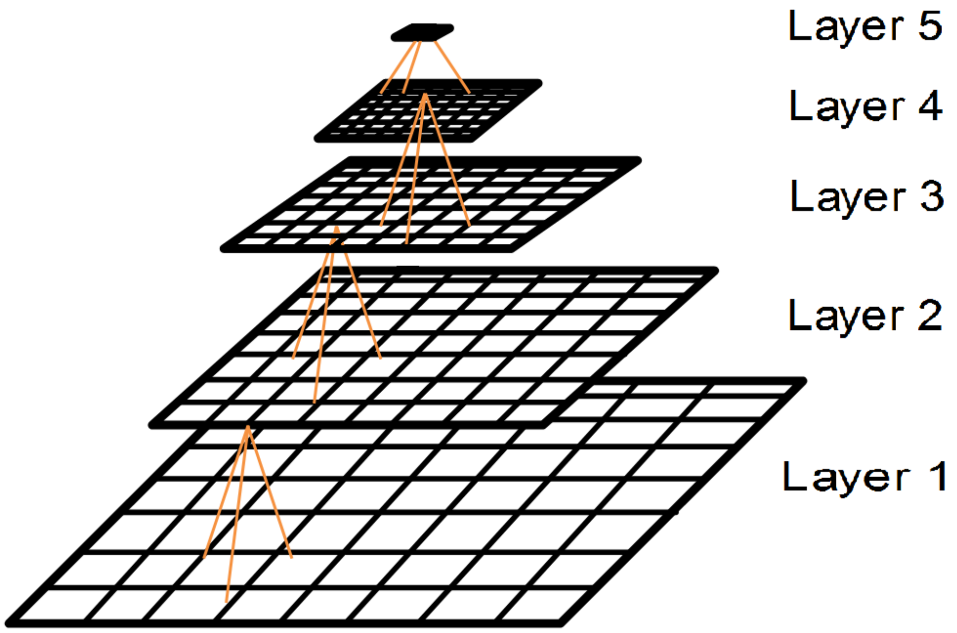


Figure B.2: A five-layers HTM deep learning structure.

At each time slot t, the projection result from the set of NJ body joints into the set of all RFs $RF(A)$ is a vector:

$$v_{RF(A)}(t) = v_{1,1}v_{1,2}...v_{N_{RFCat},N_{Cat}} \tag{B.1}$$

where the length of this vector is the number of RFs in set A, $N_{RF(A)} = N_{RFCat}.N_{Cat}$

Each action sample $\mathcal{S}$ is a collection of moving body joints in 3D. The projection result from $\mathcal{S}$ into the set $RF(A)$ is a matrix:

$$M\left(NF(\mathcal{S}), N_{RF(A)}\right) = \left[v_{RF(A)}(t)\right]_{t=1}^{NF(\mathcal{S})} \tag{B.2}$$

where $NF(\mathcal{S})$ is the number frames of samples $\mathcal{S}$.

An image Img is a sliding window on the matrix M. The sequence of images from samples $\mathcal{S}$ projected on the set $RF(A)$ as:

$$Seq_{RF(A)}(\mathcal{S}) = \left\{Img_j\left(wsize, N_{RF(A)}\right)\right\}_{j=1}^{l(S)} \tag{B.3}$$

where $wsize$ is the width of image and $l(\mathcal{S}) = NF(\mathcal{S}) - wsize + 1$.

In order to get sequences of images for $depthNet_{L_i}(i \geq 2)$, we need to get all sets of k_i items from N_{Cat} categories:

$$Set(k_i) = \left\{A_u^{k_i}\right\}_{u=1}^{\complement_{N_{Cat}}^{k_i}} \tag{B.4}$$

where $A_u^{k_i}$ is a set of k_i categories.

Similar to the $depthNet_{L_1}$, the sequences of images for $depthNet_{L_i}$ from action sample $\mathcal{S}$ by projected on the set of RFs $RF(A_u^{k_i})$, $u = 1, 2, ..., \complement_{N_{Cat}}^{k_i}$ are:

$$Seq_{RF(A_u^{k_i})}(\mathcal{S}) = \left\{ Img_j \left(wsize, N_{RF(A_u^{k_i})} \right) \right\}_{j=1}^{l(\mathcal{S})} \tag{B.5}$$

where $RF(A_u^{k_i})$ is the set of RFs belongs to set $A_u^{k_i}$ and $N_{RF(A_u^{k_i})}$ is the number of RFs belong to the set $A_u^{k_i}$.

For every sample S_j in each action category, we calculate sequences $Seq_{RF(A)}(S_j)$ and $Seq_{RF(A_u^{k_i})}(S_j)$ $(1 \leq u \leq \complement_{N_{Cat}}^{k_i}; 2 \leq i)$ which are used to train $depthNet_{L_1}$ and $depthNet_{L_i}$.

B.3 Action Recognition

After trained from sequences of images, the depth architecture networks can recognize actions in real-time as following:

• At each time slot $t_j (t_j > wsize)$, we get image $Img_{t_j}(wsize, N_{RF(A)})$ by projecting the set of NJ body joints into the set of all RFs $RF(A)$ from $t_{j-wsize+1}$ to t_j.

• This image Img_{t_j} is an input to the $depthNet_{L_1}$. From the output, we select set of k_1 highest values candidates $A_{u'}^{k_1}$.

• We create the image $Img_{t_j}(wsize, N_{RF(A_{u'}^{k_1})})$ by projecting the set of NJ body joints into the set of RFs $RF(A_{u'}^{k_1})$ from $t_{j-wsize+1}$ to t_j. This image is an input to the higher network $depthNet_{L_2}$ to get k_2 candidates.

• This process is continuous until it reaches the top network. The final output is the recognition result.

Our propose method not only can recognize actions in real-time, but also from the middle of the actions.

B.4 Experiments

We use the 3D motion capture data from HDM05 motion capture database [27] for performance validation. To extract the action features, we collect twenty different actions as showed in Table B.1. Each action has from eight to thirty-five samples and performed by five different actors. After calculating the confident for each RF, we select top 50 RFs for each action category.

B.4.1 Experiment 1: Multilevel-networks and Single-network with the changing window size

We use the description of Hierarchical Temporal Memory (HTM) architecture by Davide [42] to create our HTM networks. After converting the extracted action features into sequences of images for training, we use the Numenta Platform for Intelligent Computing (NUPIC) [43] to combine these networks into depth architectures. We use HTM [42] to create two depth networks $depthNet_{L_1}$ trained with number of categories $k_1 = 5$ and $depthNet_{L_2}$ with $k_2 = 4$. The $depthNet_{L_1}$ has eight layers with the size of input sensor is 128×128, while $depthNet_{L_2}$ has seven layers and the input sensor size is 64×64. The numbers of nodes in each layer of each network are described in Table B.2. Each node in higher layer gets the input from its four child nodes except the node in layer one gets

Table B.1: List of experiment actions.

No.	Original Actions (*NoS)	No.	Semantic Testing Actions (*NoS)
1	kickRFront1Reps (35)	11	kickRSide2Reps (20)
2	punchRFront1Reps (35)	12	punchRSide1Reps (33)
3	shuffle2StepsLStart (18)	13	shuffle2StepsRStart (18)
4	walk2StepsLstart (36)	14	walkLeftCircle4StepsLstart (7)
5	sitDownChair (25)	15	sitDownFloor (25)
6	sneak2StepsLStart (21)	16	sneak2StepsRStart (21)
7	standUpLieFloor (25)	17	standUpSitFloor (25)
8	throwSittingHighR (19)	18	throwSittingLowR (19)
9	cartwheelLHandStart1Reps (26)	19	cartwheelRHandStart1Reps (8)
10	jogLeftCircle4StepsRstart (22)	20	jogLeftCircle4StepsRstart (22)

(*)NoS is the Number of Samples.

input directly from the sensor images. While training or recognizing, the input images of two networks are re-sized to fix their sensor sizes.

Table B.2: Networks parameters.

Network	NoL(*)	Numuber of nodes in each layer
$depthNet_{L_1}$	7	[(1),(2×2),(4×4),(8×8),(16×16),(32×32),(64×64)]
$depthNet_{L_2}$	8	[(1),(2×2),(4×4),(8×8),(16×16),(32×32),(64×64),(128×128)]

(*) NoL is the Number of Layers.

We experiment the accuracy of this depth architecture by changing the window size $wzise$, in Eq. (B.5), from 20 to 140 frames on the set of five actions No.1 to No.5 as showed in Table B.1. We randomly two-thirds samples of the action sequences for training and used the remaining one-thirds for testing.

We compare the results from two depth architecture networks, the *multilevel-networks* combined from $depthNet_{L_1}$ and $depthNet_{L_2}$, while the *single-network* has only $depthNet_{L_1}$. The results are showed in the Figure B.3. In general, the accuracy results from the *multilevel-networks* as showed in red line are higher than from the *single-network* as show in blue line. The *multilevel-networks* gets the highest accuracy 94.71 percent at $wsize = 80$. The accuracy of both depth architectures always get higher than 88 percent. The results reduce to around 93.5 percent when the window size $wsize$ is larger than 100 pixel because we fit the size of input sensors are 128×128.

In the first experiment, we create the confusion matrix for *multilevel-networks* (trained from action No.1 to No.5) with window size $wsize = 20$, which is showed in the Figure B.1. The actions *kickRFront1Reps*(No.1) and *punchRFront1Reps*(No.2) often confuse to each others. At the beginning and ending of these actions, there are some similar positions which create this confusion.

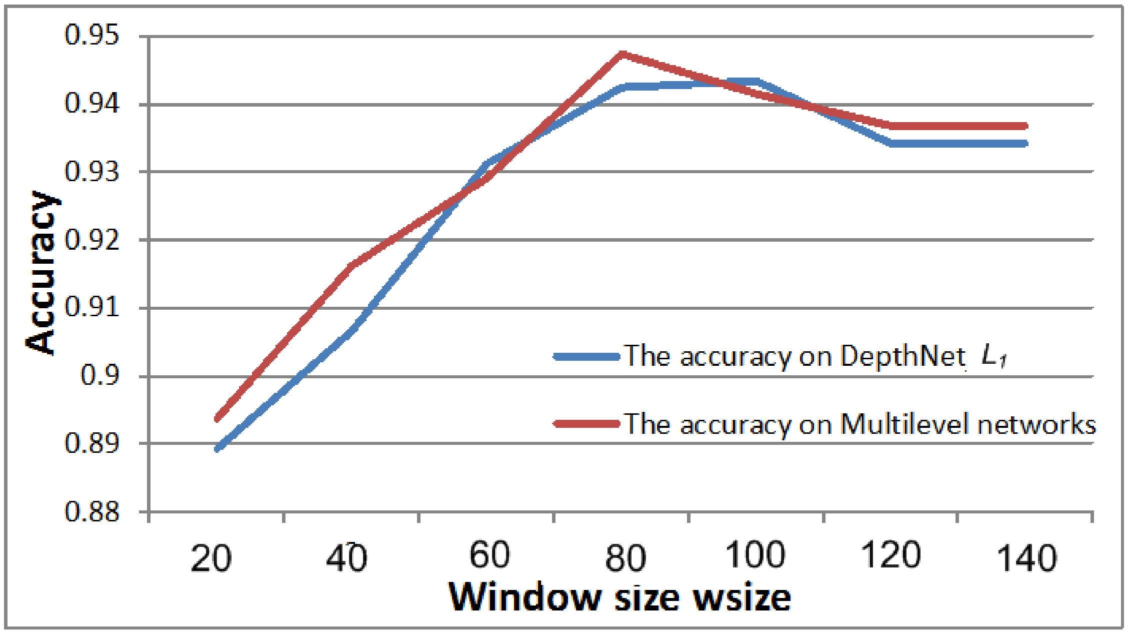


Figure B.3: The accuracy of two depth architecture networks with the changing of window size *wsize* (Eq.(B.5)).

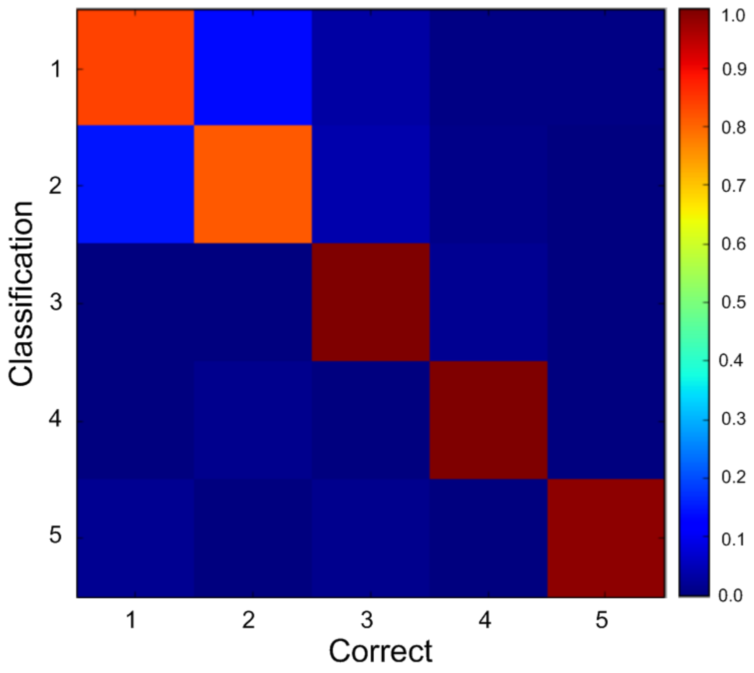


Figure B.4: Confusion matrix of *multilevel-networks* for five actions No.1-5 with $wsize = 20$.

B.4.2 Experiment 2: The combining depth architecture networks

We experiment with another depth architecture network combined from $depthNet_{L_1}$, $depthNet_{L_{2,1}}$ and $depthNet_{L_{2,2}}$ which have the same seven layers, the size of input sensors are 64×64 and the window size $wzise = 80$. At the layer one, $depthNet_{L_1}$ is trained from the set of ten actions No.1 to No.10 as showed in Table B.1. At the layer two, $depthNet_{L_{2,1}}$ is trained five actions No.1 to No.5 and No.6 to No.10 for $depthNet_{L_{2,2}}$.

The depth architecture structure for the second experiment is showed in the Figure B.3. At the layer one, the sequences of boolean images input to the $depthNet_{L_1}$ which is trained from ten actions (No.1 to No.10) to get the top five candidates. The greater number of candidates appear in the trained action lists decides which deep learning network (trained from five actions No.1 to No.5 or No.6 to No.10) in layer two will be used for the next step. The input image for the result network are filtered from the original image by five candidates actions. The highest value in its vector output decides the winner category.

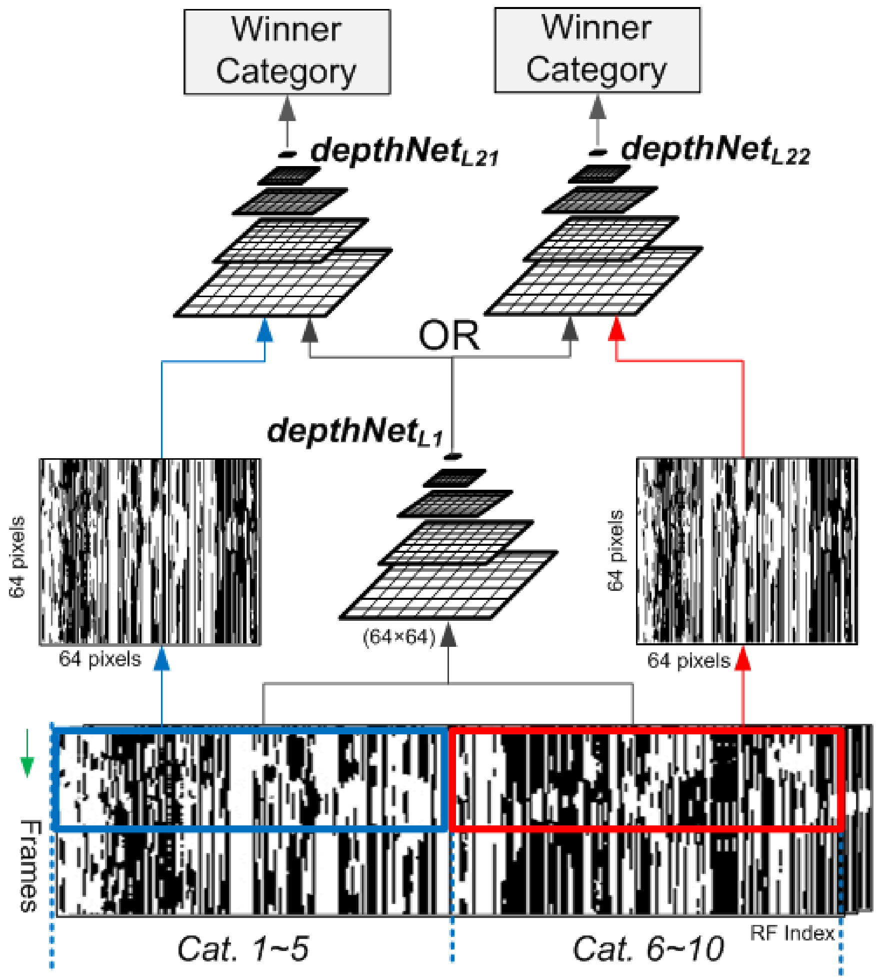


Figure B.5: Depth architecture for 10 actions.

To evaluate this architecture, we collect the set of ten semantic testing actions which have similar meaning to the original set, e.g. we train the action *punchRFront1Reps*(2) and test the semantic action *punchRSide1Reps*(12). All the pairs of original and semantic testing actions are showed in the Table B.1. The accuracy from the original set is 91.47 percent while the semantic testing set gets 75.26 percent. The Figure B.4. shows more detail of each action accuracy from both sets. In the semantic testing set, the accuracy of actions *punchRSide1Reps*(12), *sneak2StepsRStart*(16) and *throwSittingLowR*(18) get nearly equal to their correlated original actions. Especially, the accuracy from action *kickRSide2Reps*(11) gets 99.41 percent accuracy higher than the original action *kickR-Front1Reps*(1) which gets 79.70 percent.

The deep learning is used to create our architecture networks helps the speed of processing can reach to 1.296 seconds for each frame. Before the actions have finished, it already gave the results. For further work, we will focus on training more actions by grouping the set of actions into different kinds of blocks and could manage them by the semantic network. Using different kinds of action blocks, we only can function the depth architecture networks, but also easy to manage our program memory.

The deep learning networks require programming memory. The higher layer it has the large number of combining subsets of actions for training. To overcome this issue, instead of grouping all actions in one network, we train actions in different blocks then combine them in one depth architecture.

The confusion matrix on testing data-set for ten actions No.1 to No.10 is showed in the Figure B.4. The actions *sitDownChair(No.5)* and *standUpLieFloor(No.7)* have the highest confusion to each others due to their same local skeleton positions. We could manage the length of input sequences of images to reduce this local decisions in further works.

In this work, we group the sets of actions into blocks. In future works, we could train more actions in different blocks and manage them by semantic network. It could have two kinds of blocks. The *simi-blocks* which contain similar actions (e.g. *walking block* contains *walking left*, *walking right*,...) and the *diff-blocks* which contain different actions (e.g. *diff-block*$_i$ contains actions *runing*, *walking*, *jumping*, ect.) The *diff-blocks* could overlap to each others. So that, in recognizing, it uses more than one *diff-blocks* to recognize the main action category, then use *simi-blocks* to focus in detail. We use the semantic network to manage the actions between blocks. If the ambiguous between actions higher than a threshold, they should be retrained again in another blocks.

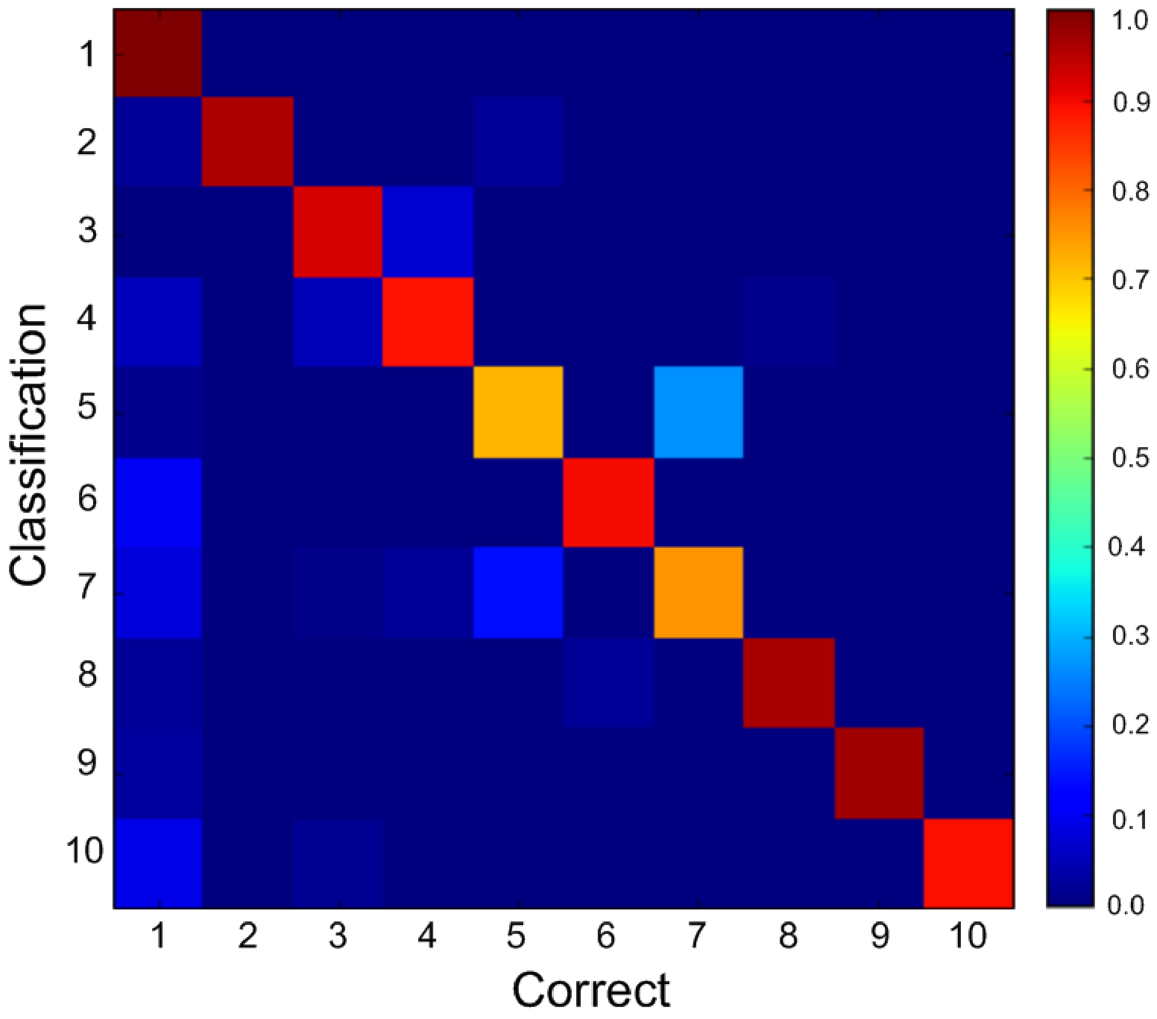


Figure B.6: Confusion matrix 10 actions.

In this first experiment, we propose a depth architecture networks which trained from semantic action features for real-time action recognition. The action features are extracted based on the velocity feature of body joints, relational feature concept and statistic method. We convert the extracted features into sequences of images which are feed into multilevel networks in depth architecture. These networks not only can learn the spatial and temporal from extracted features, but also can focus on objects more detail to recognize actions in real-time. Beside of the advantages, it still lefts some considering issues: 1) The information in action features will loss if we convert then re-size them into the fixed images size; 2) When the number of action increases, the combination of set of actions will dramatically increase. It lets the deep learning network's memory does not capacity for the huge of sequences of images. We provide some discussion and solutions for our future works to over these issues in Appendix C.

Appendix C

Dealing with the Large Number of Actions

As mention in the Chapter 5 and Appendix B, beside of the advantages of using action feature for real-time action recognition, it still lefts some considering issues: 1) The information in action features will loss if we convert then re-size them into the fixed images size; 2) When the number of action increases, the combination of set of actions will dramatically increase. It lets the deep learning network's memory does not capacity for the huge of sequences of images. We provide in this appendix some discussion and solutions for these issues.

We introduce in the deep architecture networks to learn the extracted action features from 3D MOCAP data for real-time action recognition. We apply our semantic annotation approach of the human motion capture data to automatically extract a set of semantic action features. These features are convert to sequences of images, then feed to the deep architecture networks for real-time action recognition. The whole process is showed in Figure C.1.

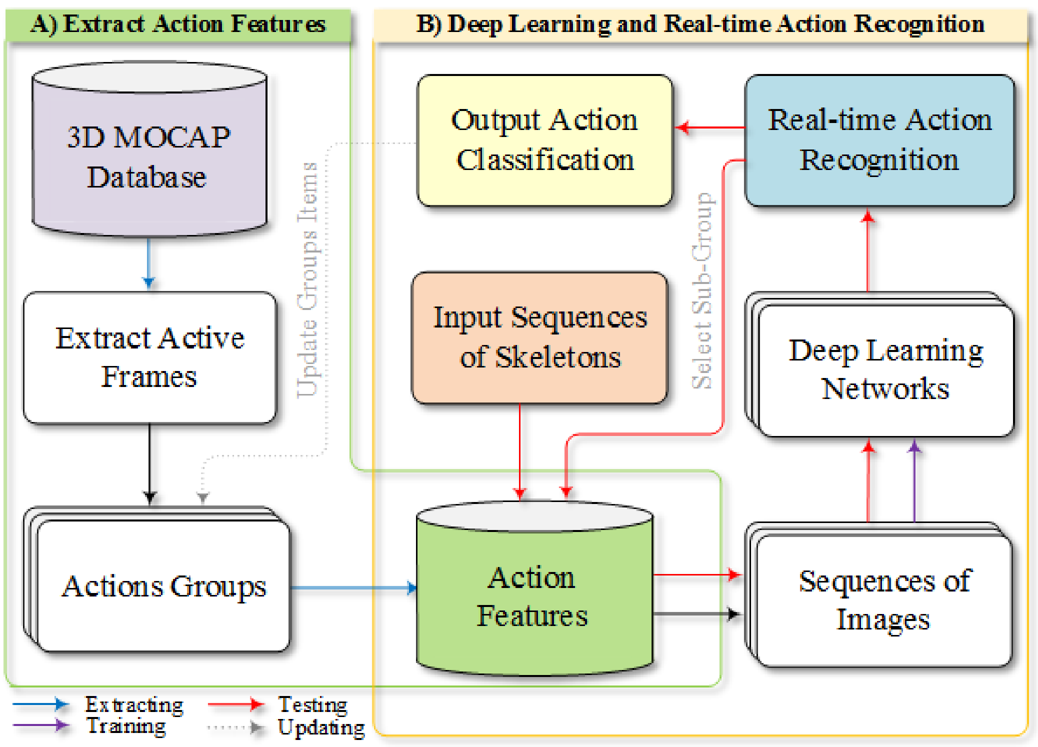


Figure C.1: Flow chart of the proposed approach.

Step 2: Real-time Action Recognition.

1. *Deep learning networks*: For each action group, we divide its action features into sub-sets, then create one deep learning network for each sub-set. These sub-sets of action features are used to convert the 3D moving joint into sequences of images which are used to train the networks.

2. *Real-time Action Recognition*: We create the a deep architecture model which is combined from many deep learning networks of action groups. The input of moving joints are converted into sequences of images which are parallel feed to each deep learning networks. The recognition process goes from groups to smaller ones to get the winning action category in real-time.

The structure is given as following: In section C.1, we extract the action features from the 3D MOCAP data. In section C.2, we create the deep architecture by combining deep learning networks. We convert extracted action features into sequences of images to train these networks for real-time action recognition.

C.1 Automatic Extraction Action Features

For real-time processing and action recognition, we first divide set of all actions G into NG smaller groups to extract the action features.

$$G = \{G_i\}_{i=1}^{NG} \tag{C.1}$$

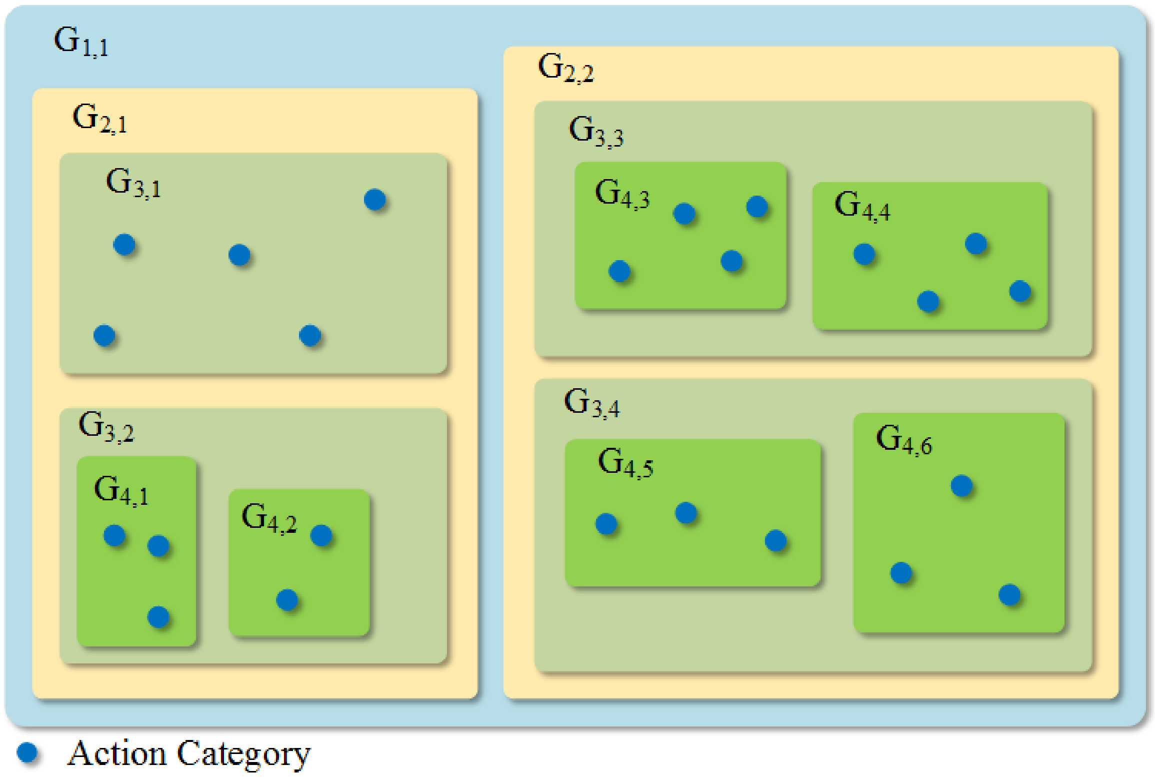


Figure C.2: Groups and sub-groups of actions.

We denote group G_i is the father group of G_j: $G_i = fa(G_j)$ iff $\nexists G_k : (G_k \supset G_i) \wedge (G_j \supset G_k)$.

One group can be divided into smaller groups if it satisfies: 1) Its sub-groups are not overlap to each others; and 2) A group contains only action categories or smaller sub-groups.

1. If $G_k = fa(G_i)$ and $G_k = fa(G_j)$ then $G_i \cap G_j = \varnothing$.
2. If $\exists Action_l \in G_i$, $1 \leq l \leq N^{ACT}$ then $\nexists G_j : G_i = fa(G_j)$.

We denote N_{cat} is the number of categories in a group and N_g is the number of sub-groups in a group. The number of items in group G_i is:

$$Items(G_i) = \begin{cases} N_{Cat}(G_i) \ if \ \exists Action_l : Action_l \in G_i, \\ N_g(G_i) \quad otherwise. \end{cases} \tag{C.2}$$

where, $1 \leq l \leq N^{ACT}$.

The Figure C.2 shows one example of all action categories $G_{1,1}$ which is divided into smaller groups $G_{2,1}$, $G_{2,2}$,$G_{2,3}$ and $G_{2,4}$. The group $G_{3,1}$ contains only five action categories, meanwhile the group $G_{3,2}$ contains two smaller sub-groups $G_{4,1}$ and $G_{4,2}$.

We extract action features for every group $\mathcal{G} \subset G$ by using our method as proposed in chapter 5. After defining set of GRFs as basic action units, we apply them to all body joints and create the bag of features. We consider these features as words in the documents, then using beyond TF-IDF weighting [34] to calculate the words weighting.

We select the set of top-N words, which have highest weighting from each category, are considered as action features:

$$\mathbf{\Phi}_{\mathcal{G}} = \{F_{\mathcal{G},j}\}_{j=1}^{N_{\mathcal{G}}^{RF}} \tag{C.3}$$

where $N_{\mathcal{G}}^{RF}$ is the number of relational features.

For parallel and real-time process, we separate the action features of each group $\mathcal{G}$, $\mathbf{\Phi}_{\mathcal{G}}$ into different sub-sets:

$$FSet_{\mathcal{G}} = \{F_{\mathcal{G},i}\}_{i=1}^{NS_{\mathcal{G}}} \tag{C.4}$$

where $NS_{\mathcal{G}}$ is the number sub-sets of group $\mathcal{G}$.

C.2 Deep Architecture Networks and Real-time Action Recognition

The deep learning networks which are structured from hierarchical layers can learn the spatial-temporal features from sequences of vision data in unsupervised manner. The nodes in each layer will combine input signals, groups them by the Markov chains networks then send output to the higher nodes. The top node covers all the signals from lower layers and classify the input to different categories. Layer by layer, the network learns sequences of input information. By this way, it learns the signals from each image as spatial features and the order of those appearance as temporal features.

We create the deep architecture by combining from the hierarchical deep learning sub-networks. Each sub-network is called a *Node* which presents for each action group. Each *Node* contains set of deep learning networks which are correlated to the sub-sets of its action features. The sequences of images are created by projecting the 3D MOCAP data into each sub-set of action features. The deep learning networks in groups are trained from these sequences of images. After training, the deep architecture can recognition action in real-time.

C.2.1 Depth Architecture Networks

For each action group G_i, $1 \leq i \leq NG$, we create set of deep learning networks $DLNet_{i,k}$, $1 \leq k \leq NS_{G_i}$ which are correlated to its sub-sets of action features $F_{G_i,k}$. We get the sequences of images by projected the sequences of input skeletons to all sets of RFs $F_{G_i,k}$, $1 \leq k \leq NS_{G_i}$. These sets of sequences of images are used to train the correlated deep learning networks $DLNet_{i,k}$. The illustration of this process is described in the Figure C.3. We create one *Node* G_i by combining all deep learning networks which are correlated to its sub-sets of action features as described in the Figure C.4.

The deep architecture is created in a hierarchical structure by combining from all *Node* G_i, $1 \leq i \leq NG$ as described in the Figure C.4. The *Node* G_i is lower than *Node* G_j if and only if $G_i = fa(G_j)$. The root node (e.g. *Node* $G_{1,1}$) contains all action categories. The leaf-*Nodes* (e.g. *Node* $G_{4,1}$, *Node* $G_{4,2}$, $G_{4,3}$ and $G_{4,4}$) are contain only the action categories while other *Nodes* are connected to their father *Nodes.* The sequences of skeletons is the parallel input to all *Node* on the networks. The sequences of skeleton are parallel input to each *Node.* The ouput of each *Node* decide which sub-group contains winner action categories candidates. The output from the leaf-*Nodes* give the winner action category.

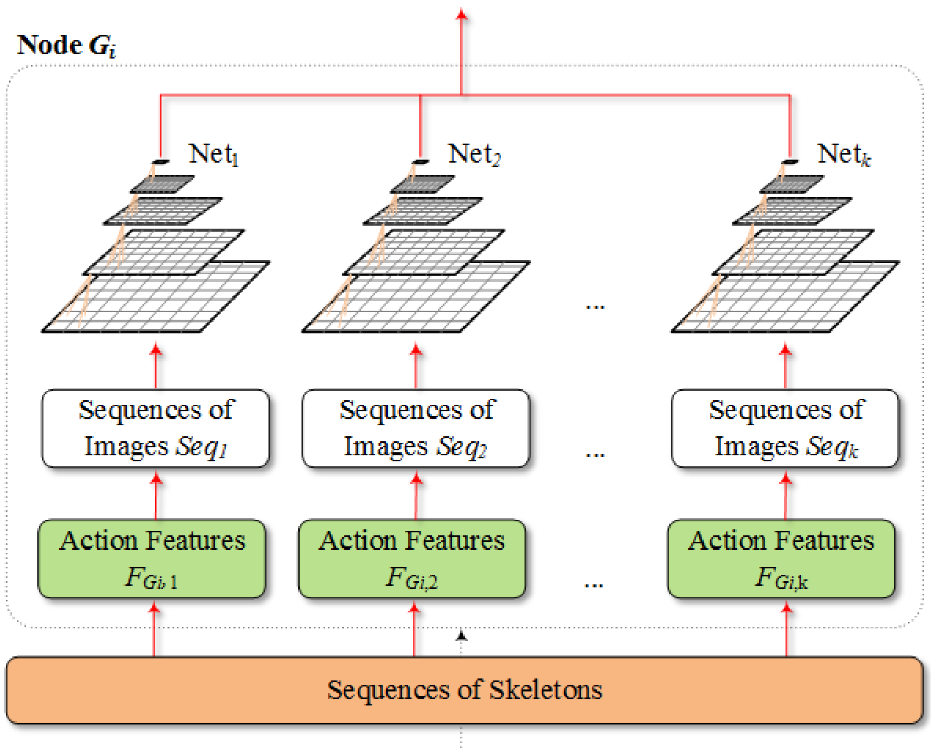

Node G_i
Net$_1$
Net$_2$
Net$_k$
...
Sequences of Images Seq_1
Sequences of Images Seq_2
Sequences of Images Seq_k
Action Features $F_{Gi,1}$
Action Features $F_{Gi,2}$
Action Features $F_{Gi,k}$
Sequences of Skeletons

Figure C.3: The structure of node group G_i.

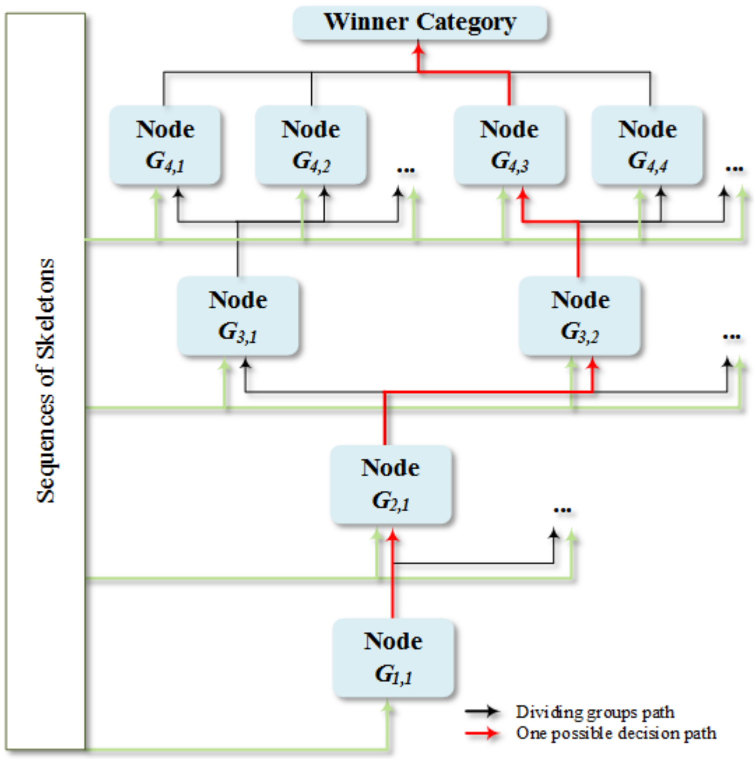


Figure C.4: Deep architecture network.

C.2.2 Sequences of Images

At each time slot t, the projection result from the set of NJ body joints into the set of action features $F_{\mathcal{G},i}$, $1 \le i \le NS_{\mathcal{G}}$, $\mathcal{G} \subset G$ is a vector:

$$v_{F_{\mathcal{G},i}}(t) = v_1 v_2 ... v_{|F_{\mathcal{G},i}|} \tag{C.5}$$

where $|F_{\mathcal{G},i}|$ is the number of RFs in $F_{\mathcal{G},i}$.

Each action sample $\mathcal{S} \in Action_l, Action_l \subset \mathcal{G}$ is a set of moving body joints in 3D. The projection result from $\mathcal{S}$ into the set of action fatures $F_{\mathcal{G},i}$, $1 \le i \le NS_{\mathcal{G}}$ is a matrix:

$$M(NF(\mathcal{S}), |F_{\mathcal{G},i}|) = \left[v_{F_{\mathcal{G},i}}(t)\right]_{t=1}^{NF(\mathcal{S})} \tag{C.6}$$

where $NF(\mathcal{S})$ is the number frames of samples $\mathcal{S}$.

An image Img is a sliding window on the matrix M. The sequence of images from samples $\mathcal{S}$ projected on the set of action features $F_{\mathcal{G},i}$ is:

$$Seq_{F_{\mathcal{G},i}}(\mathcal{S}) = \{Img_j(wsize, |F_{\mathcal{G},i}|)\}_{j=1}^{l(\mathcal{S})} \tag{C.7}$$

where $wsize$ is the number of processing frames and $l(\mathcal{S}) = NF(\mathcal{S}) - wsize + 1$.

We get all training sequences of images from all samples $\mathcal{S} \in Action_l$, all $Action_l \subset \mathcal{G}$ and all action groups $\mathcal{G} \subset G$.

C.2.3 Real-time Action Recognition

The depth architecture networks can recognize actions in real-time after training:

At each time slot $t_j (t_j > wsize)$, we get sets of images $\left\{Img_j\left(wsize, |F_{G_{i,k}}|\right)\right\}$ for all group G_i, $1 \le i \le NG$, and $1 \le k \le NS_{G_i}$ by projecting the set of NJ body joints into all sets of RFs $F_{G_{i,k}}$ from $t_{j-wsize+1}$ to t_j. These sets of images are parallel input to the deep learning networks.

- *Step 1:* We start with *Node* G_i which contains all action categories. With an input image $Img_j\left(wsize, |F_{G_{i,k}}|\right)$, $1 \le i \le NG$, $1 \le k \le NS_{G_i}$, the deep learning network $DNet_{i,k}$ gives vector output:

$$outSet_{G_i,k} = o_{1,k} o_{2,k} o_{3,k} ... o_{Items(G_i),k} \tag{C.8}$$

  the value $o_{h,k}$, $1 \le h \le Items(G_i)$ is the value result of action $Action_h$ in group G_i.

- *Step 2:* The vector output of *Node* G_i is the summary values all output vectors from all deep learning network $DNet_{i,k}$, $1 \le i \le NG$, $1 \le k \le NS_{G_i}$:

$$outNode_{G_i} = n_1 n_2 n_3 ... n_{NS_{G_i}} \tag{C.9}$$

  where $n_i = \sum_{j=1}^{Items(G_i)} o_{i,j}$, $1 \le i \le NS_{G_i}$.

- *Step 3:* The decision path j, $1 \le j \le NS_{G_i}$ is chosen, if n_j is the highest value in the vector $outNode_{G_i}$:

– If the processing *Node* G_i is not a leaf-*Node*, the processing continues from *Step 1* with *Node* G_j and $G_i = fa(G_j)$ until it reaches one leaf-*Node*.

– *Otherwise*, the winner category is the $Action_j$.

The number of testing items in action category $Action_l$, $1 \leq l \leq N^{ACT}$ is the summary number of processing frames of its testing samples:

$$numTestingItems_l = \sum_{\mathcal{S} \in Action_l} NF(\mathcal{S}) - wsize \tag{C.10}$$

The accuracy of action category $Action_l$,$1 \leq l \leq N^{ACT}$ is calculated as:

$$Accuracy_l = \frac{correctAnswers_l}{numTestingItems_l} \tag{C.11}$$

where $correctAnswers_l$ is the number of correct answers (the winner category output are $Action_l$).

In this appendix, we propose a depth architecture which trained from semantic action features for real-time action recognition. The action features are extracted based on the velocity feature of body joints, relational feature concept and statistic method. We convert these features into sequences of images which are feed into hierarchical deep learning networks in depth architecture. These networks not only can learn the spatial and temporal from extracted features but also can recognize actions in real-time. The experiment results shows that our propose method not only gets high accuracy in real-time action recognition, but also can recognition from the middle of the actions. We could apply this model to the human tracking systems and other practical applications in future works.