Motion Capture Data Processing, Retrieval and Recognition

Zhao Wang

Thesis

Oct, 2017

Copyright Statement

Abstract

Character animation plays an essential role in the area of featured film and computer games. Manually creating character animation by animators is both tedious and inefficient, where motion capture techniques (MoCap) have been developed and become the most popular method for creating realistic character animation products. Commercial MoCap systems are expensive and the capturing process itself usually requires an indoor studio environment. Procedural animation creation is often lacking extensive user control during the generation progress. Therefore, efficiently and effectively reusing MoCap data can brings significant benefits, which has motivated wider research in terms of machine learning based MoCap data processing.

A typical work flow of MoCap data reusing can be divided into 3 stages: data capture, data management and data reusing. There are still many challenges at each stage. For instance, the data capture and management often suffer from data quality problems. The efficient and effective retrieval method is also demanding due to the large amount of data being used. In addition, classification and understanding of actions are the fundamental basis of data reusing. This thesis proposes to use machine learning on MoCap data for reusing purposes, where a frame work of motion capture data processing is designed. The modular design of this framework enables motion data refinement, retrieval and recognition.

The first part of this thesis introduces various methods used in existing motion capture processing approaches in literature and a brief introduction of relevant machine learning methods used in this framework. In

general, the frameworks related to refinement, retrieval, recognition are discussed.

A motion refinement algorithm based on dictionary learning will then be presented, where kinematical structural and temporal information are exploited. The designed optimization method and data preprocessing technique can ensure a smooth property for the recovered result. After that, a motion refinement algorithm based on matrix completion is presented, where the low-rank property and spatio-temporal information is exploited. Such model does not require preparing data for training. The designed optimization method outperforms existing approaches in regard to both effectiveness and efficiency.

A motion retrieval method based on multi-view feature selection is also proposed, where the intrinsic relations between visual words in each motion feature subspace are discovered as a means of improving the retrieval performance. A provisional trace-ratio objective function and an iterative optimization method are also included.

A non-negative matrix factorization based motion data clustering method is proposed for recognition purposes, which aims to deal with large scale unsupervised/semi-supervised problems. In addition, deep learning models are used for motion data recognition, e.g. 2D gait recognition and 3D MoCap recognition.

To sum up, the research on motion data refinement, retrieval and recognition are presented in this thesis with an aim to tackle the major challenges in motion reusing. The proposed motion refinement methods aim to provide high quality clean motion data for downstream applications. The designed multi-view feature selection algorithm aims to improve the motion retrieval performance. The proposed motion recognition methods are equally essential for motion understanding. A collection of publications by the author of this thesis are noted in publications section.

Acknowledgements

First of all, I would like to express my gratitude to my supervisors Dr. Xiaosong Yang and Prof. Jiang J. Zhang for their continued support and advices. Their guidance and help, from research to life, are worth my lifelong gratitudes. I am really grateful to meet such supervisors as them, who are always patient to listen my thoughts and ideas. This work is the result from our countless meetings, especially with Dr. Yang, who encourages me throughout this four years. Also, I would like to thank Shuang Liu, for the collaboration and daily discussion.

Part of the inspirations leading to this work also arise from discussions with Dr. Yinfu Feng, Dr. Tian Qi. Their critical comments, kindly mind and resources sharing are always there when I need them most.

Memorable friendships have been established during my 4 years study in NCCA. Thanks to Prof. Jian Chang, Prof. Lihua You, Dr. Zhidong Xiao, Prof Ruofeng Tong, Prof. Jun Xiao, Prof. Yike Guo, Dr. Chao Wu, Dr. Meili Wang and Dr. Hongchun Yu for their help and support. Thanks to Kun Qian, Shuang Liu, Tao Jiang, Li Wang, Shaojun Bian, Yipeng Qin, Wenshu Zhang, Shujie Deng, Min Jiang, Shihui Guo, Yanran Li, Dan Song, Rongqiang Qian, Chao Yan and other friends for the amazing time. Thanks to Jan Lewis, Sunny Choi, Jane Whitaker and all gradute school staffs, who handled all the administrative tasks and supported all my visiting travels.

Gratitude also goes to the Sino UK project, BU PhD Fundings, Santander PGR mobility Award, AniNex project for PhD studentship and financial support of my activities.

Finally, I am greatly indebted to my family, including my parents and parents-in-law, my wife and our daughter.

Declaration

This thesis has been created by myself and has not been submitted in any previous application for any degree. The work in this thesis has been undertaken by myself except where otherwise stated.

Contents

List of Figures

List of Tables

List of Abbreviations

AR Augmented Reality .

BN Batch Normalization.

BP backward propagation.

CNN Convolutional Neural Network.

DTW Dynamic time warping.

EEG Electroencephalography.

EMG Electromyography.

GAN Generative adversarial network.

GMM Gaussian mixture model.

HCI Human Computer Interaction.

IMU Inertial measurement unit.

LSTM Long Short Term Memory networks.

LTI Linear time invariant.

MoCap Motion Capture.

NMF Non-negative Matrix Factorization.

RMSE Root mean square error.

RNN Recurrent Neural Network.

SGD stochastic gradient descent .

ToF Time-of-Flight.

TrNN Truncated Nuclear Norm.

VR Virtual Reality .

List of Publications

Relevant publications

1. Liu Shuang, **Wang Zhao**, Yang Xiaosong, and Zhang Jian J. Realtime Dynamic 3D Facial Reconstruction for Monocular Video In-The-Wild, The IEEE International Conference on Computer Vision (ICCV), 2017.

2. **Wang Zhao**, Liu Shuang, Hu Wenyu, Tong Ruofeng, Yang Xiaosong and Zhang Jian J. Face Alignment Refinement via Exploiting Low-Rank property and Temporal Stability, Proceedings of the 30th International Conference on Computer Animation and Social Agents (CASA). ACM, 2017.

3. Hu Wenyu, **Wang Zhao**, Liu Shuang, Yang Xiaosong, Yu Gaohang and Zhang Jian J. Motion Capture Data Completion via Truncated Nuclear Norm Regularization, IEEE Signal Processing letters, 2018, 25(2): 258-262

4. **Wang Zhao**, Feng yinfu, Liu Shuang, Xiao Jun, Yang Xiaosong and Zhang, Jian J. A 3D human motion refinement method based on sparse motion bases selection, Proceedings of the 29th International Conference on Computer Animation and Social Agents (CASA). ACM, 2016: 53-60..

5. **Wang Zhao**, Feng Yinfu, Qi Tian ,Yang Xiaosong and Zhang Jian J. Adaptive multi-view feature selection for human motion retrieval. Signal Processing, 120:691–701, 2016.

6. **Wang Zhao**, Liu Shuang, Qian Rongqiang, Jiang Tao, Yang Xiaosong and Zhang, Jian J. Human motion data refinement unitizing structural sparsity and spatial-temporal information, 2016 IEEE 13th International Conference on Signal Processing (ICSP), IEEE, 2016: 975-982.

7. Qian Rongqiang, Zhang Bailing, Yue Yong, **Wang, Zhao** and Coenen Frans. Robust Chinese Traffic Sign Detection and Recognition with Deep Convolutional Neural Network. in Proceedings of

2015 11th International Conference on Natural Computation (ICNC),IEEE, 791–796, 2015.

Contributions in co-authored publications For paper 1, the author has discussed with Dr Liu and contributed to the 3D facial reconstruction model design. In addition, the author has also designed a NMF model, which is used for facial expression clustering step. For paper 3, the author has discussed with Dr Hu and contributed to the low-rank refinement model design and optimization. In addition, the author has also conducted the experiment and results. For paper 7, the author has discussed with Dr Qian and contributed to the CNN model design and architecture optimization.

Other irrelevant publications

1. Liu Shuang, Yang Xiaosong, **Wang Zhao**, Xiao Zhidong and Zhang Jian J. Real Time Online Facial Expression Transfer with Single Video Camera, Computer Animation and Virtual Worlds, 2016, 27(3-4): 301-310.

2. Rong Ke, Hu Wansu, **Wang Zhao**, Ren Qun, Yang Xiaosong and Deng Zhikun. The determinant of network effects of platform ecosystems, 2016 European Academy of Management (EURAM).

3. Liu Yin, Yang Xiaosong ,Cao Yang **Wang Zhao** ,Chen Biaosong, Zhang Jianjun and Zhang, Hongwu. Dehydration of core/shell fruits, Computers & Graphics ,47:68–77, 2015.

Chapter 1

Introduction

1.1 Background

Motion design is essential for character animation production in the film and game industry. According to a recent market research report[1], the total value of global animation industry was US $254 billion in 2017 and is projected to reach US $270 billion by 2020. Existing methods of 3D character motion design can be categorized into three main fields: **artistic 3D animation**, **motion capture technique** and **procedural animation**.

1. **Artistic motion authoring** Artistic motion authoring approaches craft character poses and motions by adjusting character limbs iteratively with a variety of techniques, such as key frame animation, inverse and forward kinematics (IK/FK) and multiple targets morphing. These tedious operations on local body parts cause problems on the overall integrity of the design and low efficiency. As a result, the quality of result animation depend highly on the professional skills of appropriate artists.

2. **Motion capture technique (MoCap)** MoCap techniques pro-

[1]Global Animation, VFX & Games Industry: Strategies, Trends & Opportunities, 2018,https://www.researchandmarkets.com/reports/4449895/global-animation-vfx-and-games-industry

vide an effective solution to save animators from the laborious work of manually adjusting character models. Live motion is recorded directly from actors and then mapped on to a 3D character model, where the resulting animation looks real and natural. Recent advances in actor performance capture technologies have facilitated the capture of body motion at real-time framerates. Motion capture has traditionally focused on the skeletal posture of actors, with commercial solutions using optical marker tracking, multi-view cameras, and single-view camera configurations. In addition, recent work has also enabled the reconstruction of body shape as well as skeletal detail. Hybrid methods are gaining popularity as well, combining depth sensors, inertial measurement units (IMU), and pressure sensors in order to improve the performance compared to purely vision-based approaches. However, limitations to MoCap are evident: the motion is captured under specific environments with specific characters. Artifacts can occur when a character of different proportions or different environment is involved. Hence, it is difficult for animators to adjust the MoCap result to meet special requirements from artistic design .

3. **Procedural animation** Procedural animation approaches are based on physical simulation. which use computational models to create and control the motion for simulating virtual characters. Procedural animation can achieve visual realism, which is adaptive for different settings of character properties and robust to external perturbation. In general, these kinds of methods can be divided into four categories: *trajectory optimization*, m*odel based controller design*, *direct policy learning*, *reinforcement learning and deep reinforce learning for control*.

   Although physical simulation based methods can provide high quality results, such kind of methods are computational expensive and time consuming [Xia et al. 2017]. In addition, the lack of user control during motion generating process is not user-friendly for animators.

In sum, due to the fast growth of entertainment markets such as game and films in the last decade, the demand for character animation production is increasing year by year. Moreover, the recent explosive growth of virtual reality (VR) and augment reality (IR)industry is also in need of huge amounts of 3D character animation sources. Therefore, an intuitive and effective character motion design method with flexible user control is in great demand, which forms the motivation and wider relevance behind this project. Compared with procedural animation methods, the MoCap method could generate high quality result animations with lower costs. In addition, the large amount validate existing MoCap data in turn providing a solid base for developing data reusing techniques. As such, considering between the effectiveness and efficiency,the MoCap based method is chosen in this project.

1.2 Main challenges

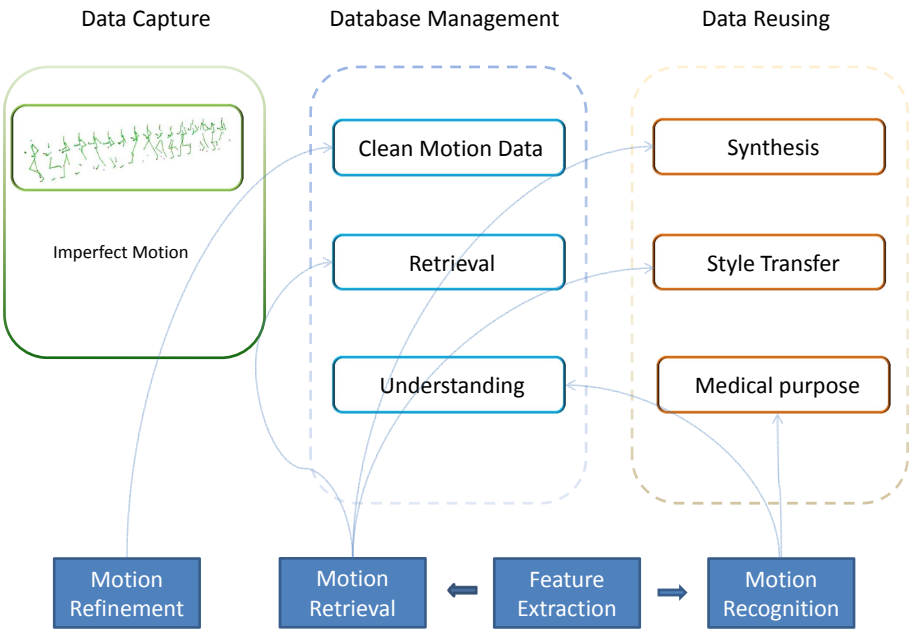


Figure 1.1: *The working flow of 3D motion data reusing.*

As shown in Fig. 1.1, the general work flow of motion data reuse can be divided into 3 stages: data capture, management and reusing itself.

Generally, difficulties occur at every stage of 3D MoCap data reusing for motion design purpose. Major challenges still remain across all motion data reusing stages.

1. **Data Quality problem** The problem of data quality influences both the data acquisition stage and the data management stage. State-of-art optical based MoCap technology has already been widely used in many areas, such as film industry, computer games, sports, medical rehabilitation and so on. However, even with professional MoCap system, the recorded motion often contains noises and missing values, which requires tedious post-processing work to refine the data. On the other hand, existing MoCap datasets also contains noises and missing values, which brings difficulties for building large database. The refinement of MoCap data is a challenging task due to its high dimensionality, time and frequency variance.

2. **Large Amount Data Management** Recently, widely used MoCap techniques has encouraged the explosive growth of motion data. A huge mass of human motion data has been accumulated in these years. Managing and accessing this data is a difficult task however, which is important for data reusing. Hence, it is essential to develop an effective and efficiency retrieval techniques, where the complexity of human motion and the huge amount of data brings additional challenges.

3. **Motion Understanding** The applications of motion data reusing for generation purpose such as stylized motion generation demand high requirement in terms of motion understanding. Other applications such as motion editing also have similar requirements. The complex nature of this data, different actors and multiple modalities cause difficulties for understanding model design. A possible solution is to design and develop an effective motion recognition techniques to overcome these difficulties. Besides, motion data recognition techniques are also essential for managing the motion data in a semantical level.

1.3 Research Aims

The aim of this research is therefore to solve these key technical challenges in motion data reusing. The major tasks here range from motion data refinement, motion management and motion understanding, all of which have significant influences at different stages.

Motion data refinement aims to provide high quality level motion data that benefits many downstream applications. For example, this will help enhance the sense of immersion for featured films, augmented reality (AR) and virtual reality (VR). Hence, it needs to design motion refinement techniques, e.g filling missing value and denoising, in order to refine imperfect motion data. On the other hand, designing the motion enhancement methods to improve the motion data quality recorded by low-cost MoCap systems is also helpful. Motion management proposes to effectively manage the recorded motion data, which requires to restoration, retrieve and recognition of data. Besides, the feature extraction plays an essential role in the motion retrieval and motion recognition process, which motivate the author towards machine learning methods to design powerful features for the desired purpose.

1.4 Research Objectives

In order to achieve the aim, the following objectives need to be accomplished:

- **Literature Review**: review and investigating current literature on 3D motion processing or the techniques used on similar multimedia contents.
- **Motion Refinement**: design and develop motion refinement algorithms that are able to accurately and effectively deal with outliers or missing marks appearing in 3D motion recording. Such kinds of techniques can be used in the motion capture stage to address the data quality problem. In addition, providing clean data is also

meaningful for downstream motion data applications.

- **Motion Retrieval**: design and develop effective and efficient algorithms on motion retrieval, which address the problem of large amount motion management. In addition, it is also essential for downstream reusing applications, such as motion editing and motion synthesis.
- **Motion Recognition** design and develop effective and efficient algorithms on motion recognition, which is essential for motion management, understanding and editing.

1.5 Contributions

There are several major contributions towards different tasks in this PhD research:

- Propose a novel ℓ_1 partial dictionary learning based motion refinement method, which aims to deal with missing marker and outlier problems in motion capture. (Related Publications: [Wang et al. 2016b, 2017b]).
- Propose a low-rank matrix completion based motion refinement method, which does not require pre-training and perform well for the long term motion capture data. (Related Publications: [Wang et al. 2017b; Hu et al. 2017a])
- Propose a manifold learning based multi-view feature selection model to extract features for solving the motion retrieval problem. The scalability with large motion dataset, capability for further multi-modality data and insensitivity with the algorithm parameter, make the proposed method applicable for real-world applications. (Related Publications: [Wang et al. 2014c]).
- Propose a NMF based unsupervised motion data clustering method, which can be used for motion recognition and data preprocessing of motion synthesis.

- Propose a multi-task learning CNN model to solve motion recognition problems on 2D gaits; develop a CNN model for motion recognition on 3D MoCap data. (Related Publications: [Qian et al. 2015]).

1.6 Structure of the following chapters

The following parts of this thesis comprises 6 more chapters:

- **Chapter 2** Literature review on related research topics, including motion refinement, motion retrieval and motion recognition, related machine learning techniques such as non-negative factorization (NMF), multi-modality learning, and deep learning on motion data, etc.
- **Chapter 3** The analysis of widely used motion refinement methods and presentation of a dictionary learning based method, which handles the missing marker problem.
- **Chapter 4** Present a low-rank estimation methods for motion refinement, which does not require pre-training. The designed iterative optimization can reduce the computational cost while dealing with long term motion sequences.
- **Chapter 5** Present a motion retrieval method based on multi-modality feature selection. The new feature representation can improve retrieval performance. Scalability with large motion dataset, capability for further multi-modality data.
- **Chapter 6** Present a multi-task convolutional neural network model for 2D gait recognition. Different attributes of human gait are studied to extract a discriminative feature for recognition purpose. Present a NMF based motion clustering method and a CNN based 3D motion recognition method.
- **Chapter 7** Conclusion and future plan.

Chapter 2

Literature Review

In this chapter, firstly, the background information of motion capture is introduced. Then, the related works of data-driven based motion data processing works, including data refinement, retrieval, recognition and synthesis are reviewed. After that, the related machine learning techniques that used in this PhD project are reviewed, including the multi-feature fusion, non-negative factorization and the deep learning model.

Mathematical Notations In order to clearly present the models and algorithms in this thesis, some important notations used in the rest of this thesis are provided here. The capital letter,e.g., X, represents matric or dataset, $X_{i,:}$ is the i-th row of X and $X_{:,j}$ is the j-th column of X. For any vector x, several vector norms are defined as

$$\|x\|_0 = \Sigma_{x_i \neq 0} |x_i|^0 \qquad \|x\|_1 = \Sigma_{i=1} |x_i|$$

For the matrix $X \in R^{p \times q}$, the squared Forbenius norm (ℓ_2-norm) and the ℓ_1-norm are defined as

$$\|X\|_F^2 = \Sigma_{ij} X_{ij}^2 \qquad \|X\|_1 = \Sigma_{ij} |X_{ij}|$$

In addition, $X \circ Y$ is denoted as the hadamard production of X and Y,

i.e. $(X \circ Y)_{i,j} = X_{i,j}Y_{i,j}$, and $\langle X, Y \rangle = tr(X^T Y)$ wherein $tr()$ is the matrix trace operation.

2.1 Background of motion data

2.1.1 Motion data acquisition techniques

MoCap technique is originally developed for film industry. The high quality motion data have been applied to generate character animation, facial animation and special digital effects in recent films e.g. *Avatar*, *The Avengers*, *Transformers*, *Captain America*, and *Warcraft*. The MoCap data is usually represented as 3D skeleton data, also known as stick-man image, where an example is shown in Fig 2.1.

Figure 2.1: *Example of MOCAP data: skeleton representation.*

The commonly used physical sensors for human motion capture include pressure sensors, magnetometer sensors, inertial sensors, acoustic sensors, and optical sensors. Generally, those MoCap systems could be classified into three categories: 1) optical based methods, 2) wearable sensors based methods and 3) depth sensor based methods.

(a)  (b) 

Figure 2.2: *The example of popular MOCAP techniques a) optical based (left actor) and wearable sensors (right actor), b) depth sensor*

The most popular commercial MOCAP systems are optical based, such as *Motion Analysis* and *Vicon*. Stereo vision techniques are used

Name of product	Description	Link to official website
ZED Stereo Camera	Depth sensor based on passive stereo vision	https://www.stereolabs.com/
OptiTrack Mo-Cap	Multi-camera system for motion tracking	http://www.optitrack.com/
Vicon	Passive optical motion capture	http://www.vicon.com/
Motion Analysis	Passive optical motion capture	https://www.motionanalysis.com/
Kinect v1/v2	Depth sensor based on ToF camera	http;//www.microsoft.com/
Creative Senz3D	Infrared sensor	http://us.creative.com/
Infineon 3D Image Sensor	Depth sensor based on ToF camera	http://www.infineon.com/
Noitom Perception Neuron	Wearable sensors based on IMU	https://www.noitom.com/
3-Space MoCap	Wearable sensors based on IMU	https://yostlabs.com/
Trigno EMG	EMG and IMU	http://www.delsys.com/

Table 2.1: *Example of motion capture hardware*

for triangulating the 3D position of a subject, where special markers are attached on to the actor to facilitate the data acquisition. Multiple cameras are used in the optical based MOCAP systems and actors are required to wear the tight cloth with retroreflective markers (Fig 2.2 (a), left character). However, there is a high environment requirement of optical based MOCAP system. For instance, the *Motion Analysis* system must be built in an indoor environment and the active area is also limited.

A typical wearable sensor based MoCap system is shown on the right actor of Fig 2.2 (a). Such kind of sensors is originally developed for biomechanics analysis. Inertial measurement unit (IMU) is contained in the sensor, which uses a combination of accelerometers and gyroscopes, sometimes also magnetometers to acquire the 3D position. Compare with the optical based MoCap systems, it ensures a wide range of operating conditions. However, the stability , e.g. the drift problem, is the bottleneck of the system. In addition, it also requires to define the skeleton model manually to get the whole body motion.

Depth sensor, e.g. *Microsoft Kinect* and other structured-light sensors provide a low cost solution to capture the 3D body information. The motion data acquired by depth sensors could be used in many areas like HCI. However, due to the limitations of the hardware, low data quality is the key problem of such kind of systems. Although many efforts have been done on improving the data quality, it's still not able

to meet the quality requirement of professional animation production. A list of popular MoCap hardware is listed in the table 2.1.

Besides, many researchers have made effort on MoCap using monocular 2D color image data to recover 3D pose, which is a very challenging problem due to the ambiguity of posture, occlusion and deformation of body. Dantone et al. have trained a joint regressors that using two layers of random forest, where a classifier is applied for detection then the joint position is obtained [Dantone et al. 2014]. The convolutional neural networks (CNNs) have achieved great successes in many computer vision areas, while researchers also apply CNNs to estimate 3D human pose. For instance, a convolution pose machine (CPM) is proposed by [Wei et al. 2016], which is implicit spatial models to estimate poses by a single image.

The result of image based MoCap approaches are often influenced by lighting conditions, environmental changing, self-occlusion and posture ambiguity. In order to improve the robustness, hybrid sensors methods are proposed. For example, Zhang et al. propose a system that combines 3 depth camera and a pair of foot pressure sensors, which reconstructs both the pose and kinetic information at same time [Zhang et al. 2014e]. Another framework employs a color camera and 5 IMUs, where the camera data is used to coordinate IMU data's offsets [von Marcard et al. 2016].

2.1.2 Motion Dataset

The **KTH** presented by [Schuldt et al. 2004] and **Weizmann** presented by [Blank et al. 2005] are famous video-based human action datasets in the early stage. Both datasets were captured in a controlled environment where the occlusion and clutter in the scene are limited. Although **KTH** and **Weizmann** have been used extensively, it limits the further improvements for future research on complex real-world applications with these particular datasets [Weinland et al. 2011].

After that, several papers have been published, including Hollywood2

by [Marszalek et al. 2009], UCF50 by [Liu et al. 2009]. These **video based** datasets collect video clips from various TV shows, movies and user contributed videos such as *YouTube*. These datasets have richer sets of activities and more complex environments, more faithfully representing the real-world scenarios. Despite of these monocular datasets, many multi-view datasets have also been developed, including the CMU motion of body (MoBo) [Gross and Shi 2001; Weinland et al. 2007]. These datasets increased the amount of information for analysis and introduced baselines for quantitative evaluation.

With the fast development of 3D motion capture techniques, several MoCap datasets were published, such as the CMU Motion Capture Dataset by the CMU Graphics Lab and the HDM05 dataset by [Müller et al. 2007a]. On the other hand, with the advent of depth sensor, e.g. Microsoft Kinect, the 3D depth motion dataset have been published, such as Msr-Action 3D [Li et al. 2010b], and Msr-DailyActivity 3D [Wang et al. 2012a]. Compare with the database that recorded from optical based MoCap system, these datasets could provide a rich depth representation of the scene. However, the captured result are only from a single viewpoint, which may cause occlusion in the scene. Additionally, using multiple depth camera may cause interference due to the ToF technique. The NTU RGB+D dataset is presented recently, which not only includes the depth map, but also the 3D scanning results of actors and the multi-view camera recordings by [Shahroudy et al. 2016]. Although the motion skeleton data contains too much noise, it is still a suitable choice to evaluate the algorithms.

Datasets	Samples	Classes	Subjects	Modality	Reference
Msr-Action 3D	567	20	10	D+Skel	[Li et al. 2010b]
Berkeley MHAD	660	11	12	RGB+D+Skel	[Ofli et al. 2013]
HDM05	2061	130	5	Skel	[Müller et al. 2007a]
CMU	2605	23	~	RGB+Skel	`http://mocap.cs.cmu.edu/`
Human3.6M	3.6M[1]	17	11	RGB+D+Skel	[Ionescu et al. 2014]
NTU RGB+D	56880	60	40	RGB+D+IR+Skel	[Shahroudy et al. 2016]

Table 2.2: *3D Skeleton Motion Datasets. Modality: D stands for depth image, RGB stands for color image, Skel stands for 3D skeleton joint position and IR stands for infrared image*

[1]Human3.6M contains 3.6M poses and 3D scanning model of 11 actors

A short summary of MoCap datasets used in this PhD research is listed in the table 2.2. The detail information of each dataset is shown below:

- **Msr-Action 3D** The Msr-Action 3D dataset provides both skeleton and depth data, where the skeleton contains 20 joints. The dataset provides 20 actions that performed by 10 subjects, each action is performed 2-3 times. Both the 3D world coordinates and screen coordinates with depth of the detected skeleton joints are provided. T
- **Berkeley MHAD** The Berkley MHAD dataset provides skeletion, depth and multi-view RGB camera videos, where the skeleton contains 30 joints that processed from 43 LED markers. The dataset provides 11 actions that performed by 12 actors, each action is performed 5 times.
- **CMU** The CMU motion dataset provides the skeleton data, where the skeleton contains 31 joints. The dataset provides 23 actions.
- **HDM05** The HDM05 motion dataset provides the skeleton data, where the skeleton contains 31 joints. The dataset provides 130 actions that performed by 5 actors. Since many of original classs are close to each other, e.g. *punchFront1Reps* and *punchFront1Reps*, the similar action classes are combined and finally 25 action classes are generated for 2605 motion clips in this project.
- **Human3.6m** The Human3.6m dataset provides the RGB image data from 4 views, the skeleton pose data, the depth data and the actor scanning data. The whole dataset consists of 3.6 million 3D human pose and corresponding images performed by 11 actors for 15 scenarios.
- **NTU RGB+D** The NTU RGB+D dataset provides the RGB videos, depth map sequences, 3D skeletal data and the infrared videos, where the skeleton contains 25 joints. The dataset consists of 60 action classes that performed by 40 actors

Basically, this PhD research project will concentrate on the motion

data processing rather than the data acquisition techniques. In the following part, the related work of motion data refinement, retrieval and recognition will be reviewed as well as the involved machine learning methods.

2.2 Motion refinement

In this section, the related work of motion data refinement will be reviewed.

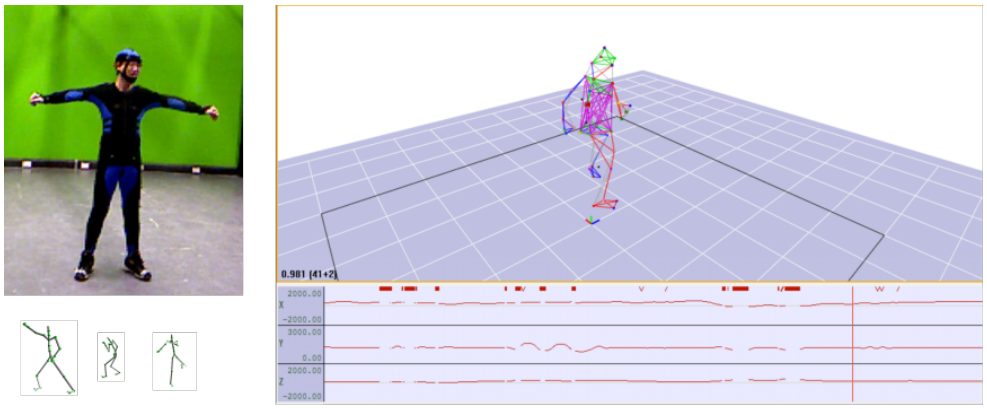


Figure 2.3: *Examples of MOCAP result recorded by* Motion Analysis System*, where the blanks on the time line denote the occurring of missing marker problem.*

Even with professional MoCap system, the recording motion data still suffers from the missing marker and noisy problems. For instance, a capturing result by *MotionAnalysis System* in the studio of NCCA is shown in Fig 2.3, where the blanks on the time line denotes the occurring of marker missing problem. Motion refinement is an essential preprocessing step for all those MOCAP data based applications. Many researchers have done a great deal of effort on the topic of human motion refinement which aims to reduce the noise and fill the missing values in the imperfect motion data. Generally, the existing approaches could be divided into three categories:

- signal processing based methods;
- data driven methods;

- low-rank matrix completion methods.

2.2.1 Signal processing based methods

In earlier studies, the classical signal processing methods are adopted for the motion data, such as Gaussian filter, discrete cosine transform(DCT), Fourier transform ,wavelet transform, Kalman filter and linear dynamic systems (LDS) [Bruderlin and Williams 1995; Yamane and Nakamura 2003; Hsieh and Kuo 2008; Li et al. 2010a, 2009]. The clean motion could be reconstructed from the perspective of signal filtering. For example, a B-spline wavelet-based agent is proposed by Hsieh and Kuo to remove the impulsive noise embedded in the noisy rigid body motion data [Hsieh and Kuo 2008]. In [Yamane and Nakamura 2003], it presents a linear time-invariant filter (LTI) approach that using dynamic filter to convert a physically inconsistent motion into a consistent one. In addition, the dimension reduction techniques used in signal processing area could also be applied to motion data, such as PCA [Tangkuampien and Suter 2006]. Later on, the manifold learning methods, which has been achieved great success in computer vision area, has also been applied to motion denoising [Wang and Carreira-Perpinán 2010]. These kinds of methods usually just require low computational cost and are efficient to handle simple and short term missing marker problem. However, each DOF(degree of freedom) of joint is processed independently, where the underlying structure correlation between human joints are usually ignored. Hence, these approaches could meet difficulties while dealing with complex motions.

2.2.2 Data driven methods

The data driven methods have also attracted much attention, such as example based learning [Lou and Chai 2010] and dictionary learning [Xiao et al. 2011, 2015]. For instance, Lou and Chai have presented an example based data driven method that using multi-channel singular spectrum analysis (M-SSA) to learn a series of filters [Lou and Chai 2010] . The

learned filter bases would then be used along with statistics techniques to filter noisy motion data. Another example is [Xiao et al. 2011], they propose an approach that casting the predicting missing marker problem as finding spare representation of imperfect pose. They succussed in introducing ℓ_1 sparse representation to solve predicting missing marker of motion data. A further improvement framework has been presented in 2015 where ℓ_2 penetration is added to filter the gaussian noise [Xiao et al. 2015]. However,for these data driven approaches, their performance are heavily rely on the training data selection. A large amount of labeling and clean data is required. In addition, the performance could decline significantly if the type of imperfect motion is not contained in the training set. Moreover, the kinematic characteristics of human motion are not considered in their dictionary learning and pose reconstruction process, which may cause the result motion "unsmooth".

2.2.3 Low-rank matrix completion methods

Lait et al. noticed the low-rank property of motion matrix has not been exploited explicitly [Lai et al. 2011] . They reformulate the human motion refinement into a low-rank matrix optimization where singular value thresholding (SVT) is applied to solve the objective function. After that, Feng et al. have proposed a motion data refinement via a matrix completion method using both the low-rank structure and temporal stability properties of the motion data [Feng et al. 2014b]. The key advantage of the low-rank based methods is that it overcomes the out-of-sample problem since no training data is required. However, the low-rank matrix completion method may fail when many data entries are badly corrupted,e.g. large amount of missing markers. In addition, the refined result by low rank based methods may not guarantee the smoothness of recovered motion.

2.2.4 Motion refinement summary

To sum up, motion refinement is essential for providing high quality motion data, which is a crucial part of motion data reusing. Existing signal processing based methods can not handle the complex motion. The performance of dictionary learning based approaches is limited by the learning database. Low-rank matrix completion methods do not require pre-training. However, the computational cost and smoothness of refined result need to be improved.

2.3 Motion retrieval

With the rise of some novel motion capture systems and technologies like depth-based MOCAP [Zhang 2012; Shum et al. 2013], it brings explosive growth of motion data. A huge mass of human motion data have been accumulated in these years, such as *HDM*05[1] dataset, *CMU*[2] dataset and *NTU RGB+D*[3] dataset. Therefore, an efficient and effective human motion management method plays an important role in managing and accessing these available data. In this section, the basis techniques of motion retrieval approach are reviewed.

Due to the high complexity and diversity of human motion, human motion retrieval is a very challenging task in computer animation and multimedia analysis communities. Similar to other multimedia retrieval tasks like image retrieval and document retrieval, feature representation is the corner-stone of human motion retrieval algorithm and system. A compact and discriminative motion feature representation can not only significantly improves the performance of the retrieval algorithm but also dramatically reduces the computational cost of the algorithm. As we known, there is a big gap between the low-level visual feature and the high-level semantic meaning, so single low-level visual feature is usually unable to fully characterize all aspects of the motion data. In other

[1] http://resources.mpi-inf.mpg.de/HDM05/

[2] http://mocap.cs.cmu.edu/

[3] http://rose1.ntu.edu.sg/Datasets/actionRecognition.asp

words, it would be beneficial to fuse multiple visual features for motion data representation.

With the explosive increasing of multimedia data, extensive research effort has been dedicated to multimedia analysis, such as image and video annotation and retrieval [Wang and Hua 2011; Wang et al. 2012b]. To reveal and exploit the geometric structure of multimedia data, many graph-based models have been proposed in recent years. They can be used as geometric image descriptors [Zhang et al. 2013b,c, 2014b] to enhance image categorization. Besides, these methods can be used as image high-order potential descriptors of superpixels [Zhang et al. 2014c,a; Ofli et al. 2012]. Further, graph-based descriptors can be used as a general image aesthetic descriptors to improve image aesthetics ranking, photo retargeting and cropping [Zhang et al. 2013d, 2014c].

Since human motion data is a special kind of multimedia data, many traditional multimedia retrieval approaches can be employed to process the motion data. In general, the main motion retrieval procedure includes three steps: 1) visual feature extraction; 2) training a search/ranking model; 3) human motion searching or ranking. To speed up the searching procedure, some data indexing techniques such as kd-tree and R-tree are used in approximate searching [Böhm et al. 2001; Keogh et al. 2004; Krüger et al. 2010].

The success of human motion retrieval heavily relies on the motion feature representation. Therefore, many researchers have focused their attentions on construction and learning efficient motion feature representation for content-based human motion retrieval.

As we know, text labels [Yoshitaka and Ichikawa 1999] such as *jump* or *kick* have been used to search motion clips at the beginning. However, it requires the user to label all motion data in database in advance, which is very tedious and time-consuming. Moreover, due to the high ambiguity and subjectivity of keyword, it is difficult to choose a correct and meaningful labels for a given human motion sequence. In fact, metric designing for the similarity measuring between two identities is still an open problem in text mining.

To overcome the shortcomings of label-based methods, the content-based methods that extract numeric features from motion data and use them to measure the similarity between two human motion have been proposed [Müller and Röder 2006; Chen et al. 2011]. Based on the used feature of the algorithm, we could divide the exiting work into two categories: frame/pose based methods and sequence/clip based methods.

2.3.1 Frame/pose based methods

To describe the local motion properly, a number of frame/pose based motion features have been developed. Chen et al. presented a geometric pose feature (GPF) which is effective in encoding pose similarity [Chen et al. 2011] . Raptis et al. have proposed an approach that using joint angles as feature to recognize dance actions [Raptis et al. 2011]. Xia et al have shown a histogram of 3D joints descriptors, after generating a dictionary, the temporal motion model is created via Hidden Markov Model (HMM) [Xia et al. 2012]. Following a similar paradigm, Kapsouras and Nikolaidis have used joints orientation angles and angles forward differences as local features to create a code book and generate a Bag of Visual Word (BOVW) to represent actions [Kapsouras and Nikolaidis 2014]. A Gaussian Mixture Model (GMM) was applied by Qi et al. to represent character poses, wherein the motion sequence is encoded, and then DTW and a string matching algorithm are applied for motion comparison [Qi et al. 2014b]. Barnachon et al. have used histograms of action poses to represent the action and employed DTW for comparing and recognizing actions [Barnachon et al. 2014]. Besides, these two approaches [Qi et al. 2014b; Barnachon et al. 2014] have shown the potential for online action recognition. Moreover, sparse representation can also be applied in motion data retrieval [Zhou et al. 2014]. Generally, for pose/frame based features, it requires an integration step such as BOVW [Kapsouras and Nikolaidis 2014], DTW [Qi et al. 2014b] or manifold embedding [Zhang et al. 2013b; Wang et al. 2014a] to represent the whole motion clip using the frame/pose based features.

2.3.2 Sequence/clip based methods

The sequence/clip based motion feature utilizes some global properties of the motion sequence, e.g. moving path or trajectories. Muller et al. [Müller and Röder 2006; Müller et al. 2009] presented the motion template (MT) that the motions of the same class can be represented by an explicit interpretable matrix using a set of Boolean geometric feature. To overcome the tolerate temporal variance problem in MT training process, dynamic time warping (DTW) was employed in their work. However, it still requires the training data in same class to have same number of cycles, otherwise MT will not be well trained. Moreover, both the training and comparing process of MT is time-consuming. Gowayyed et al. [Gowayyed et al. 2013] proposed a trajectory feature named histograms of oriented displacements (HOD). Each 3D trajectory is represented by the HOD of its 2D projection to xy, xz and yz planes. A temporal pyramid method, where the trajectory is computed using the whole, halves and then quarters of the motion sequence, is applied to preserve the motion temporal information. Compared with previous trajectory features [Wu et al. 2008; Yang et al. 2010], HOD provides a fixed-length feature representation for variable length motion sequences, which benefits further analysis. [Xiao et al. 2013] proposed a probability graph model for motion retrieval. However their work requires selecting representative frames from motion sequence before generating the probability graph.

2.3.3 Sketch based retrieval

Recently, a new technique called sketch-based searching, has been introduced in motion retrieval. Chao et al. [Chao et al. 2012] used a hand drawing sketch as query to describe the body/joint trajectories. For both the input query and the motion clips in database, spherical harmonic is used to encode the joint/body trajectories for motion representation. [Choi et al. 2012] proposed a sketch based motion retrieval framework that using skeleton stick figure as input query, where the motion clips

in the database are converted to sequences of stick figures. Then the retrieval work is realized by matching stick figures. However, it requires special skills and experiences for drawing stick figure, which brings in some new difficulties for the user.

2.3.4 Motion retrieval summary

To sum up, motion retrieval is a fundamental part of motion management. Existing methods can be categorized into frame/pose based and sequence/clip based category. In this thesis, a multiple visual features fusing model for motion representation is presented to improve the retrieval performance.

2.4 Motion Recognition

Previously, human motion recognition techniques are widely used in many areas, but most of the existing frameworks [Poppe 2010] in the state of art focused on the individual motions rather than interactions. A upper body detector is used in [Patron-Perez et al. 2010] to find and track people, where the head orientations and the Histogram of Oriented Gradient (HOG) feature are computed to train a structured Support Vector Machine (SVM) for motion recognition. In their later work [Patron-Perez et al. 2012], it is improved for different aspects. A hierarchical model is proposed by [Kong and Jia 2012] for human interaction recognition, where the global and local patch features of each person are extracted and Conditional Random Field (CRF)is applied for class prediction. This model is enhanced in their later work [Kong et al. 2012, 2014]. Basically, the occlusion and moving objects in the background which have overlapping motion patterns are the primary cause of misclassification for these monocular video approaches.

Although these difficulties are scarcely possible to be overcome in monocular videos, they can be easier solved with depth information. Hence, motion recognition from depth video became a hot topic in the

past few years with the popularity of depth cameras, where many effective depth features are designed for human motion recognition. In [Li et al. 2010b], the 3D silhouette points from projection maps of three directions are generated, and then an action graph is used to encode the bag of 3D points for recognition, where the 3D information in depth video is directly exploited. In the work of [Wang et al. 2012a], the local occupancy patterns (LOP) feature based on the 3D point cloud around each joint is calculated, and the fourier temporal pyramid is used to mine discriminative actionlets for classification. [Yang et al. 2012a] extract the histogram of oriented gradients (HOG) feature from the three orthogonal projections of the depth map as an global RGB-D feature. [Ni et al. 2013] divide the spatio-temporal interest points (STIP) into several depth-layered channels from depth map, and extend the original motion history images (MHI) from 2D to 3D to take the full advantage of depth information. The 3DMHI feature is fast to compute, appreciate to different kinds of motion. However, the depth images always contain a lot of noise from both the clothing of the performer and the background, where features that directly calculated from depth map is not robust enough to describe the human actions. A large depth video motion dataset captured by Kinect is presented by [Shahroudy et al. 2016], where a long-short term memory recurrent neural network (LSTM) model is applied for motion classification.

With the advance of skeleton recovery algorithms [Shotton et al. 2013b] from depth video, the skeleton represented motion features can also be employed for motion recognition. The trajectory of skeleton joints (known as MOCAP data) can be directly used by some numeric methods as a motion feature, e.g. PCA [Liu and McMillan 2006], wPCA [Forbes and Fiume 2005] and wavelet transform [Beaudoin et al. 2007]. As another category of motion feature, the geometric relations [Müller et al. 2005; Chen et al. 2009] and relative distances [Tang et al. 2008; Tang and Leung 2012] from joints are designed to explore structure information from motions. In addition, some researchers tried to summarize the high level semantic features [Qi et al. 2013, 2014a], where Gaussian mixture model (GMM) is used to predict the probability to each motion

class, and a semantic feature is constructed from those probabilities.

A key issue of motion capture data recognition is that the time serious motion sequences could have various length due to the velocity, action style and number of repetitions. The varying of sequence length may limit the recognition ability of many frameworks. In order to overcome this problem, histograms, pyramidal model or dynamic time warping (DTW) [Sakoe and Chiba 1978] are used in many approaches to align the varying length sequence to a reference one.

2.4.1 Feature representation

plays an important role in motion recognition. Traditional feature representation methods for 3D motion data can be categorized into joint based methods and dynamic based methods.

Joint based motion feature representation methods This kind of methods attempt to capture the relations of body joints' positions, where they can be further divided into four subcategories by considering their focus:

1. Spatial temporal information
2. Geometric relation
3. Key poses
4. Mining active/discriminative subset of joints

Methods of the first subcategory attempt to exploit the spatial information of motion data by measuring the pairwise distances of the 3D joints, the distance/difference between preceding frame, and the current frame and the initial/neutral pose are employed [Ellis et al. 2013; Kerola et al. 2014; Hammond et al. 2011]. The means of covariance matrix of the skeletal sequence is used in the work of [Hussein et al. 2013]. Furthermore, CNNs are also applied to capture the correlation among joints' locations [Wu and Shao 2014; Ijjina and Mohan 2014].

Methods of the second subcategory attempt to use geometric relations among different body part to represent the 3D pose. For instance, joints are considered as one associated with a quadruple of joints in the work of [Müller et al. 2005; Evangelidis et al. 2014]. A more complex approach is conducted in [Vemulapalli et al. 2014] that using the the relative 3D geometry between body parts, where the relative geometry between body parts are defined as the rigid-transformation (translation and rotation). In the work of [Wang et al. 2012a], dynamic time warping (DTW) is employed to translate action sequences that have variant length into a nominal curve. Then Fourier temporal pyramid (FTP) and linear One-versus-All SVM classifier are applied for action recognition task. Integral invariants are used by [Shao and Li 2015] to describe motion trajectories robustly and effectively. The integral invariants representation is not sensitivity to the noise. In their work, two motion trajectories will be recognized as equivalent if and only if a group transformation to map one trajectory onto the other one is existed. Nearest neighbor and DTW are applied for classification purposes.

Methods of the third subcategory attempt to represent the motion sequence in terms of key-poses, where a codebook or dictionary of these key-poses are learned from the dataset. For instance, a ℓ_1 norm histogram of motion words is used in [Ofli et al. 2012] to represent action sequences. A Gaussian mixture model (GMM) is trained by [Qi et al. 2013] for each motion class from its key frame. Then the learned GMM and sparse coding are applied for recognition. A dictionary of body poses is trained by [Lillo et al. 2014] using K-means clustering, where the whole body is divided into 4 partial regions: *right arm*, *right leg*, *left arm* and *left leg*. Moreover, it trains a hierarchical model to represent the complex motion using a mixture of basic actions.

Methods of the fourth subcategory focused on the active subsets of joints rather than the whole body. Generally, same kind of actions involve similar subsets of body part even different individual has personal action style. For instance, [Eweiwi et al. 2014] use partial least squares (PLS) [Barker and Rayens 2003] to weight the importance of joints while a kernal-PLS SVM [Rosipal and Trejo 2001] is then employed for action

classification. The contribution of each joints to the accuracy of an action recognition is determined via a genetic algorithm by [Climent-Pérez et al. 2012], then the representative set of joints is selected. Joints are grouped into 5 body-sets in the work of [Wang et al. 2013a] while contrast mining algorithm is applied to identity the co-occurring body-sets. A dictionary is learned from such co-occurring body-sets and then a bag-of-words approach and 1-versus-1 intersection kernel SVMs are then used to classify the pose sequences. [Wei et al. 2013] adopt a multiple kernel learning (MKL) method [Damoulas and Girolami 2008] to determine the mos representative body parts. A multi-part modeling method is presented by [Seidenari et al. 2013] where each joint is represented in a local reference coordinates that defined at the proceeding joint in the chain. Then most informative body parts are learned and a modified nearest-neighbor (NN) classifies is applied for classification.

Dynamic based motion feature representation methods The dynamic based 3D skeleton action recognition methods focus on modelling the dynamics rather than the joints/subsets of joints in the skeleton data, where techniques such as linear dynamical systems (LDS) and hidden Markov models can be used.

Similar to the frameworks that mining subset of joints, [Chaudhry et al. 2013] divide the skeleton into several body parts then compute a shape context feature representation for each body part, where the motion sequence is represented as a set of time series of features. Then, LDS is applied to model each time series of feature while the corresponding system parameters is learned. After that,the estimated parameters are used to represent the motion sequence and MKL [Damoulas and Girolami 2008] is used to study a set of optimal weights. In [Slama et al. 2015], motion sequence is described as a collection of time series corresponding to joints' 3D location, where an autoregressive moving average model (ARMA) is adopted to each motion sequence to capture the dynamics using observability matrix. Hence, the comparison of any two ARMA model represented motion sequences can be performed on the Grassmann manifold, where all motion sequence are projected on control tangents

(CTs) space that form a local tangent bundle (LTB) representation. A linear SVM is then applied for classification. [Li et al. 2012] capture captures dynamic properties of short tracklets by using Hankelets to describe trajectories of the 3D joints. [Li et al. 2011]'s work shows that 3D joints' trajectories of same kind of activity live in a dynamic subspace angle (DSA) subspace, where the discriminant function for canonical correlations (DCCs) are used to increase the separation between classes, and Hankel matrices are used to capture the temporal information.

[Lehrmann et al. 2014] present a hidden Markov approach called dynamic forest model (DFM) for human motion, which uses a set of autoregressive trees [Meek et al. 2002]. [Raman and Maybank 2015] design a hierarchical Dirichlet process-hidden Markov model (HDP-HMM) for action recognition, where the non-parametric HDP-HMM allows the inference of hidden states automatically from training examples. Another related dynamic based action recognition work is proposed by [Zanfir et al. 2013], where the body pose is defined as a continuous and differentiable function of joints' locations over time.

2.4.2 Summary of motion recognition

To summarize, human motion recognition is still a challenge task due to its diversity and complexity. In this thesis, the designed unsupervised NMF model and deep learning techniques are used to generate representative and discriminative features that deal with these difficulties.

2.5 Motion Synthesis

Motion capture is an effective way of realistically obtain digital representation of dynamic human body, which can often be expensive, time consuming and require professional actors. Therefore, it is an alternative approach that considers reusing MoCap data to synthesize new motions. Data driven methods and physical simulation methods are two popular solutions for motion synthesis. The physical-based simulation methods

compute the driving torques of skeleton through the force and torque given by environmental constraints. However, the lack of user control and high computational cost are the weakness of physical based simulation approaches. Therefore, the following part of this section will focus on the data-driven approaches, which is reusing existing motion sequences for editing and synthesis. Existing data driven based approaches can be divided into following categories: motion interpolation, motion editing, statistical model, motion graph and stylized motion synthesis.

2.5.1 Motion Interpolation

Motion interpolation based methods interpolate existing motion sequences create a new peace of motion sequence. Such kind of methods usually map exiting motion data which is in time domain into a subspace that suitable for interpolation. For instance, a geostatistical method is proposed by [Mukai and Kuriyama 2005] that is used to control the process of motion blending. [Kovar and Gleicher 2004] presented a weighted interpolation method that using a continuous motion sequence space. [Wang et al. 2015] translate the problem of motion planning between two substantially different poses to a task of boundary value solving, where an energy graph is taken into account.

2.5.2 Motion Editing

Motion editing frameworks provide an intuitive way to synthesis new motion via editing existing motion sequence. For example, [Gleicher 2001] present a trajectory control method that modify the original motion data by key frame editing. Such kind of method is easy to use and it is intuitive for user to edit the action. However, it can be very time-consuming if the motion sequence to be edited is long. A similar approach has been proposed by [Huang and Kallmann 2016], which is used for planning of whole-body motion of virtual demonstrators. [Kim et al. 2016] propose a precomputed spatial map by taking object interaction into account, which aims to rearget human motion to virtual avatar in real time.

2.5.3 Motion Generation based on Statistical Model

Statistical models and machine learning models can also be applied to generate motion sequence. Early framework presented by [Tanco and Hilton 2000] employs clustering based hidden Markov models to generate motion sequence between two key frames. [Pullen and Bregler 2000] decompose the motion sequences in the frequency domain and then generate the joint angle and global movement translation. A statistical linear dynamic system modeling is applied to motion synthesis by [Chai and Hodgins 2007], where the user constrained motion generation problem is regarded as maximum a posteriori probability. [Lau et al. 2009] use a Bayesian dynamic model to generate motion sequences that the output result has simlar spatio-temporal variation. A Gauss process based motion synthesis method is proposed by [Min and Chai 2012], where the movement is decomposed into finite structural variations and continuous style variations. After that, the decomposed components are encoded with a concatenation of morphable functional models.

2.5.4 Motion Graph

Given a corpus of MoCap data, motion graph based methods divide the data into several fragments and construct a directed graph to encapsulates connections among the database. New motions sequence is generated by building a works on the graph to reassemble the poses. Unlike previously reviewed methods, motion graph based method can be applied to the whole body motion creation [Arikan and Forsyth 2002; Kovar et al. 2002], but also partial body such as limb [Ikemoto and Forsyth 2004]. The major limitation of the motion graph based methods is the results are restricted by the motion sequences in the database.

2.5.5 Stylized Motion Synthesis

Human motion can semantically be represented as a combination of two factors: content and style. The content of motion generally refers to the nature of the action such like walking and running, while the style denotes the particular way that motion is performed. The stylistic of human motion can provide an expression of particular personality, mood and/or role, which is essential for storytelling and bringing characters to life. Hence, the style or emotion of the motion may convey more meaning than underlying motion content itself.

Motion can be produced by cutting and pasting pre-recorded examples while the results is able to achieve realism similar to that of pure motion capture play back. Although snippet concatenation can produce novel content by arbitrarily complex new movements, it is restricted to synthesize only the subset of input database. Therefore, data driven motion synthesis has attracted great attentions, where it aims to make use of large motion capture datasets to allow animators produce convincing character movements from high level parameters.

Considering the difference of the concentration on motion content and style, traditional stylized motion synthesis can generally be classified into two categories: generation based methods and style translation based method.

For generation based methods, both the content and the style are involved in style motion synthesis, which are treated as two hidden factors. Then a statistical model is used to solve this problem. For example, a hidden Markov model (HMM) is used in the work presented by [Brand and Hertzmann 2000], where the motion content is regarded as hidden states and the motion style is treated as state transition probabilities. A multi-factor form of the Gaussian process latent variable model is proposed by [Wang et al. 2007a] for identify the style differences of human body motion. [Min et al. 2010] firstly construct a motion database that contains large number of pre-registered action data. After that, a multilinear model is designed to characterize the motion style and content,

which facilitates downstream applications such as motion style synthesis and editing. [Ma et al. 2010] present a method that simultaneously model style and variation of MoCap data, where several joint groups are used to represent the whole body skeleton and latent parameters are introduced to represent the variation of each group. The Bayesian network is used to identify the relationships between the style and the latent parameters of joint group variation.

The style translation methods focus on characterizing the differences between motion of different styles, where the content is still retained. In other word, it aims to generate a new motion sequence that has a specified style but the same content of the input. Hsu et al. [2005] model the differences of motion style with a LTI system. The input motion can be converted to other styles once the parameters of LTI system is trained. Similarly, a GMM based method is designed by [Ikemoto et al. 2009] to model the kinematics and dynamic of motion for converting styles. In the work presented by [Xia et al. 2015], a candidate motion sequence, which is closet to the input, is retrieved from the database firstly. Then a local mixtures of autoregressive (MAR) model is learned for motion style transfer. Despite of the LTI used in Hsu et al. [2005], the local MARs can represent the nonlinear relationships better and handle unlabeled input.

Building motion style database and generating vivid motion are still challenging tasks. Recent development in machine learning, especially deep learning, have shown great potential for this area. It is valuable to exploit the applications of various CNN based architectures, including GAN on stylized motion generation.

2.6 Multi-modality feature selection

Previously, many researchers tend to use a single type of visual feature as MoCap data representation in the past years, which leads to under-performing retrieval performance [Müller and Röder 2006; Chen et al. 2011]. In contrast, recent studies have shown that using multi-view fea-

tures can dramatically boost the algorithms' performance in computer vision and machine learning [Xia et al. 2010; Feng et al. 2013b; Tang and Leung 2012]. The core problem in these methods is how to exploit these multi-view features into a high dimensional representation, which would bring in the risk of dimensional explosion as well as the redundancy of informationn. Because each visual feature characterize different aspects of human motion and has dissimilar discriminative power with repect to a specific class of motion, it is wise to jointly take these multi-view features into account and exploit the embedded complementary information.

The traditional feature selection methods such as Laplacian Score [He et al. 2005] and Feature Ranking [Zhao and Liu 2007] are developed to select features that preserve the data property or data manifold structure. However, these learning methods are designed for the single-view feature data scenario, and they do not consider the correlations and complementary information between different features when they applied to deal with multi-modality data that comprise multiple features in feature representation. Thus, some multi-modality feature selection methods are developed for various applications like multimedia data ranking, annotation, recognition and retrieval [Xia et al. 2010; Yang et al. 2011; Feng et al. 2013b; Yang et al. 2013; Wang et al. 2014b; Han et al. 2012, 2014], where the information and correlations between different features are exploited. To integrate multimodel feature, Zhang et al. [Zhang et al. 2013a] presented the feature correlation hypergraph (FCH), where shared entropy is proposed to capture the high order correlation among multi-modality features. Multimodel graph learning learning methods can also be implemented to video annotation [Wang et al. 2009a] and image ranking [Wang et al. 2012c]. Moreover, some metrics [Wang et al. 2009b; Yu et al. 2012] have been developed specially for multi-modality feature based methods, where they are more accurate to precisely measure the similarities and dissimilarities rather than the traditional Euclidean distance metric when using multi-modality features.

However, as motion data belongs to a special kind of multimedia data, few multi-modality feature based researches are reported on this topic. Some of previous multi-modality feature selection approaches [Feng et al.

2013b] can handle multiple real data applications, but no enough experiments on motion data have been taken and shown. Chen et al. [Chen et al. 2011] used feature selection method to boost their proposed geometric pose descriptor. However, their work focuses on the task of pose recovery from the monocular image. Gao et al. [Gao et al. 2014] developed a multi-modality action recognition approach based on collaborative representation [Zhang et al. 2011]. Tang and Leung [Tang and Leung 2012] have proposed to select suitable kinds of features for each query and then split them. Their work [Gao et al. 2014; Tang and Leung 2012] did not consider correlations between the original features, where some prior researches [Xia et al. 2010; Feng et al. 2013b; Wang et al. 2014b] have already shown that leveraging shared information is beneficial to improve the recognition accuracy. Therefore, a multi-modality feature selection method for motion data retrieval is developed in this project, which takes the discriminating power of each feature synergistically.

2.7 Non-negative Matrix Factorization (NMF)

Non-negative matrix factorization (NMF) is initiated by [Paatero and Tapper 1994; Paatero 1997; Lee and Seung 1999, 2001], which incorporates the non-negative constraint in the factorized matrices. Thus, it obtains part-based representation while enhancing the interpretability of the issue correspondingly. The part-based representation is indeed on the basis of physiological and psychological evidence: perception of the whole is based on perception of its part, which is one of the core concepts in recognition [Ullman et al. 1996]. In addition, the nonnegativity constraint in NMF will naturally lead to sort of spareness [Lee and Seung 1999]. Moreover, the learned basis of NMF commonly correspond with certain semantic interpretation. For instance, the learned basis images are parts of faces in face recognition, such as eyes, nose, mouth and cheeks [Lee and Seung 1999]. Therefore, NMF is an imperative tool in MoCap data processing due to the enhanced semantic interpretability

under the nonnegativity and the ensuing sparsity.

2.7.1 Brief review of NMF

For a given non-negative data matrix $X = [x_1, x_2, \ldots, x_n] \in \Re^{d\times n}$, each column of X is a data sample. The classical NMF aims to find two non-negative matrices $Z = [Z_1, Z_2, \ldots, Z_r] \in \Re^{d\times k}$ and $G = [G_1, G_2, \ldots, G_n] \in \Re^{k\times n}$ such that

$$X \approx ZG,\ X \in \Re^{d\times n},\ Z \in \Re^{d\times k},\ G \in \Re^{k\times n}. \tag{2.1}$$

The product result of ZG is determined as a NMF of X.

NMF leads to learn a linear part-based feature representation,which is highly consistent with the human motion data that parts form a whole [Lee and Seung 1999]. In other word, NMF has a good feature representation ability in human motion data representation. Additionally, if $d \ll n$, G could be regarded as a low-dimensional representation of input X, which means that NMF is also a suitable dimensionality reduction method of motion data. Moreover, the development of motion capture clearly shows a tendency of multi-modality fusion, e.g. data collected from multiple sensors rather than the simple position information. The latent components of motion are jointly estimated, where NMF is an effective multi-view technique that allows advanced inference.

Standard NMF algorithm In order to find an approximate NMF of X, a proper cost function should be designed firstly. According to [Lee and Seung 2001], there are two most popular cost functions for NMF, where Eq. 2.2 is the square Euclidean distance (SED) based and another one Eq 2.3 is the generalized Kullback–Leibler divergence based (GKLD).

$$\|X - ZG\|_F^2 = \sum_{i=1}^{d}\sum_{j=1}^{n} \|X_{ij} - (ZG)_{ij}\|^2 \tag{2.2}$$

$$D(X \mid ZG) = \sum_{i=1}^{d} \sum_{j=1}^{n} (X_{ij} \log \frac{X_{ij}}{(ZG)_{ij}} - X_{ij} + (ZG)_{ij}) \tag{2.3}$$

[Lee and Seung 2001] have presented a standard iterative updating rule to find the global minimization of NMF.

Apart from these two cost functions shown in Eq. 2.2 and Eq. 2.3, researchers also developed other cost functions that adopt in some tasks [Cichocki et al. 2006; Xue et al. 2007; Cichocki et al. 2011; Feng et al. 2015]. Additionally, many efforts have been done on improving the updating algorithms [Zhou et al. 2012; Li et al. 2014].

2.7.2 Improved NMF algorithms

Basically, the standard NMF is not able to promise a unique solution under the only non-negative constraint. It is imperative to introduce additional auxiliary constraints on Z and/or G as regularization terms, which will also incorporate prior knowledge and reflect the characteristics of the issues more comprehensively [Wang and Zhang 2013]. In the following, three kinds of constrained NMF will be reviewed, including

1. Sparse NMF
2. Orthogonal NMF
3. NMF on manifold

Sparse NMF Although the standard NMF is able to provide the sparseness of its components, the control of this sparsity can be achieved by imposing additional constraints in addition to the non-negativity constraints. The enforced local-based representation resulted by sparseness constraint is helpful to improve the uniqueness of the decomposition. One thing need to be addressed is which factor Z or G is selected where the sparseness constraint will be imposed [Hoyer 2002, 2004].

A typical work of Sparse NMF is non-negative sparse coding (NSC) [Hoyer 2002], which employs SED and ℓ_1 norm of G in the objective function. In the NMFSC approach [Hoyer 2004], parseness measure is used for enforcing sparseness by means of nonlinear projection at each iteration. After that, the active set based algorithms, e.g. Least Angle Regression and Selection (LARS), is applied to solve the least-squares minimization of SED under the nonnegativity and sparseness constraints [Van Benthem and Keenan 2004; Kim and Park 2007; Morup et al. 2008]. Apart from these active set-based solution methods, a stable and efficient NSC approach (SENSC) is presented by [Li and Zhang 2009] , which provides quicker convergence and much more stability than NSC. Besides, sparsity can also be achieved by incorporating some structure information instead of using additional constraints on the factors. For instance, the Convex NMF is presented [Ding et al. 2010].

Orthogonal NMF Orthogonal NMF is defined as the orthogonality constraint is applied on the factor matrices, either Z or G, which is firstly introduced by [Li et al. 2001]. The purpose of using Orthogonal NMF is to minimize the redundancy between different bases [Li et al. 2001; Ding et al. 2006]. The orthogonality can be viewed as a special case of Sparse NMF while the optimization procedure is different from each other. In addition, the solution of orthogonal NMF leads to a unique sparse area.

$$\begin{aligned} E(Z,G) &= \|X - ZG\|^2 \\ s.t. Z \geq 0,\ Z^T Z = I,\ G \geq 0,\ GG^T &= I \end{aligned} \tag{2.4}$$

As shown in Eq. 2.4, the constraint of $Z^T Z$ enforce the orthogonality of columns of Z while $GG^T = I$ enforce the rows of G to be orthogonal. The orthogonality can also be imposed on both Z and G or individually.

According to [Ding et al. 2005; Li and Ding 2006], orthogonal NMF is identical to clustering since one factor matrix could be corresponding to the cluster centres and the other represent the indicators. For instance, orthogonal NMF is applied to clustering task in [Ding et al. 2006; Choi

2008; Li et al. 2010d] because of such characteristics.

Besides, a lower computational complexity NMF updating approach is presented by [Choi 2008] and further improved by [Yoo and Choi 2010], where Stiefel manifold with the nonnegativity constraint is used to solve the NMF with the orthogonality constraint. In addition, a modified multiplicative update rule is provided in [Li et al. 2010d].

NMF on manifold In practice, many real-world data are often sampled from a low dimensional intrinsic manifold embedded in a high-dimensional ambient space. Existing research has shown that the learning performance could be significantly enhanced while the intrinsic geometrical information is preserved. To this end, numerous manifold learning methods have been developed, including Locally Linear Embedding (LLE) [Roweis and Saul 2000], ISOMAP [Balasubramanian and Schwartz 2002], Laplacian Eigenmaps [Belkin and Niyogi 2003],etc. The key difference of those models is what local topological property to be considered. In other words, the essential things is how to describe and preserve the local relationship between a data sample point and its neighboring sample points.

[Cai et al. 2008, 2011] proposed the graph regularized NMF (GRNMF), where the geometrical information is combined in the original NMF objective function as the additional regularization term. In those works, a nearest neighborhood graph on a scatter of data points is construced and then a weight matrix W on the graph is defined. The local invariance assumption is made that the points in the mapped low-dimensional space should be close enough with one another if they are neighbors in the original high dimensional space. The objective function is shown in Eq. 2.5

$$E = \|X - ZG\|^2 + \lambda Tr(GLG^T) \tag{2.5}$$

where $L = D - W$, D is a diagonal matrix that $D_{jj} = \sum_l W_{jl}$ and $\lambda \geq 0$ is the regularization parameter that controls the smoothness.

Additionally, a neighborhood preserving non-negative matrix factorization (NPNMF) method is proposed by [Gu and Zhou 2009] that employs local linear embedded graph rather than the $k-$ nearest neighbor graph used in [Cai et al. 2008]. A multiple manifold learning NMF (MM-NMF) is presented by [Shen and Si 2010] where a ℓ_1 graph is used to model the geometric structure of multiple manifold.

2.7.3 Structured NMF

Despite of aforementioned methods that introduce some additional constraints, the structured NMF usually modifies the regular factorization formulation directly to enforce some structures or characteristics while solving NMF learning problem. It can be written as:

$$X \approx F(ZG) \tag{2.6}$$

Weighted NMF Weighted NMF is a typical example of structured NMF, where the model can be written as:

$$W \otimes X \approx W \otimes (ZG) \tag{2.7}$$

With the pre-defined weights W, Weighted NMF aims to approximate a low-rank matrix which is close to the input matrix X. It can be applied for matrix completion if the input matrix has missing value. To solve Weighted NMF, it can introduce the weight matrix in the standard multiplicative update rules. For instance, the Mult-WNMF algorithm has been proposed by [Mao and Saul 2004]. Moreover, EM-WNMF, a superior method that employ the EM algorithm, is presented by [Zhang et al. 2006].

To sum up, NMF is a powerful unsupervised learning method. Less NMF research on MoCap data processing has been reported. The linear part-based, nonnegativity and sparsity feature representation property of NMF makes it highly consistent with the MoCap data. A specialized NMF model is designed in this PhD research for motion understanding.

2.8 Deep learning model: Convolutional Neural Networks

The feature representation is an essential element for machine vision tasks. A compact and discriminate feature representation could significantly improve the performance. However, for a given specific task, it usually requires specify domain of knowledge and lots of time to design such a kind of feature. Moreover, the manually designed features usually can not be robust enough for general case. In other word, less dependent on human ingenuity and prior knowledge compensation are essential for feature's applicability. Thus, a learned feature representation is generally preferred rather than hand-coded features. A remarkable review on feature representation learning algorithms has been done recently [Bengio et al. 2013], which covers advances in probabilistic models, auto-encoders, manifold learning and deep learning.

The development of deep learning was initiated by [Hinton and Salakhutdinov 2006; Hinton et al. 2006] in 2006 and it immediately addressed huge attentions [Poultney et al. 2006; Bengio et al. 2007; Lee et al. 2008a]. One of the most used deep learning models is the the convolutional neural network (CNN), which is a bio-inspired hierarchical multi-layered neural network able to learn visual patterns directly from the raw data. Recently, the frameworks developed based on CNNs have been shown to yield excellent performances on speech recognition [Abdel-Hamid et al. 2014; Mao et al. 2014; Sainath et al. 2015], brain electroencephalogram (EEG) signal recognition [Cecotti and Graser 2011], natural language sentences recognition [Hu et al. 2014], visual classification tasks [Krizhevsky et al. 2012; Krause et al. 2014; Simonyan and Zisserman 2014a; Zhang et al. 2014d], visual detection tasks [Weinzaepfel et al. 2013; Girshick et al. 2014] and other computer vision tasks [Fan et al. 2010; Farabet et al. 2013; Sun et al. 2014; Taigman et al. 2014]. In the following part of this section, it focuses on the review of recent deep learning approaches that concerned with 3D skeleton human motion data prediction, recognition, editing and synthesis.

Motion Prediction As discussed in section 2.1, MoCap is a well studied problem where commercial optical based multi-camera systems can provide impressive results. However, it is still a open problem for motion capture from a single monocular camera. Earlier approaches have considered the linear motion models [Choo and Fleet 2001; Jepson et al. 2003] and non-linear models [Urtasun et al. 2006]. Recently, a sequential architecture composed of CNN named as convolutional pose machine is proposed by [Wei et al. 2016] for pose estimation [Bogo et al. 2016] present a SMPL [Loper et al. 2015] based two-stage learning model to predict the 3D skeleton from static image and parametric 3D shape that fit the skeleton. A RNN based online 3D motion detection method is designed by [Li et al. 2016]. In contrast,[Tung et al. 2017] design a self-learning model that couples 3D skeleton and mesh estimation in one framework.

Motion Synthesis There are many existing frameworks that have been designed for motion synthesis. For instance, [Taylor and Hinton 2009] and [Taylor et al. 2011] have implemented conditional Restriced Boltzmann Machines (cRBM) for gait synthesis. A recurrent temporal RBM approach has been presented by Mittelman et al. [2014] for motion reconstruction. A special RNN model called Encoder-Recurrent-Decoder (ERD) network is used by Fragkiadaki et al. [2015] to create smooth interpolated movements. A convolutional auto-encoder representation of MoCap data has been proposed by [Holden et al. 2015]. A further improvment has been developed by [Holden et al. 2016], where the motion sequence can be produced at once, in parallel, without performing any integration process.

Motion Recognition Besides of the most frameworks that reviewed in Sec 2.4 which mainly focus on hand-crafted features, the very recent deep learning based methods have shown the great power in tackling 3D motion recognition task.

A hierarchical bidirectional RNN model is used by Du et al. [2015] to model the structural and temporal dynamics of the skeleton sequences.

[Shahroudy et al. 2016] try to model the long-term temporal correlation of the features for each body part to improve the recognition performance of RNN. A LSTM based action recognition approach is proposed by [Veeriah et al. 2015] which utilizes special gating schemes for learning representations from long input action sequences. [Zhu et al. 2016] present a co-occurrence LSTM based motion feature learning framework, where a mixed-norm regularization and in-depth dropout method are adopted. In order to handle the inaccurate/noisy 3D skeleton data, a trust gating mechanism is introduced to LSTM model by [Liu et al. 2016a]. A further global context-aware attention LSTM is proposed by [Liu et al. 2017a], where an attention representation is generated to the network model to optimize the classification performance.

In contrast to these RNN/LSTM based approaches mentioned above, the CNN model is also applied to MoCap data recognition. [Hou et al. 2016] use three orthogonal canvases to encode the spatio-temporal information of a skeleton sequence into color texture images, then apply a CNN model to extract features for action recognition. Similarly, [Wang et al. 2016a] encode the dynamic information in the skeleton sequences into multiple texture images then map these images into HSV space to form the Joint Trajectory Maps (JTMs). [Li et al. 2017] encode the pair-wise of skeleton joints into texture maps called Joint Distance Maps(JDMs), which is then used as the input of CNN models. [Liu et al. 2017b] propose to translate the skeleton sequence to a series of visual and motion enhanced color images. A sequence based view-invariant transform is applied to enhance the robustness and multi-stream CNN is used for recognition. The part-body based feature vectors are used by [Ke et al. 2017a] to represent the motion sequence. Then the features are transformed into images to feed into CNN. In their another work [Ke et al. 2017b], the motion sequence is represented as a clip with several gray images. [Kim and Reiter 2017] propose to use the Temporal CNN [Lea et al. 2016] for 3D action recognition, where a Res-TCN is designed by factoring out the deeper layers into additive residual terms. [Huang et al. 2017] present a work that trains a deep learning architecture on Lie group, named as LieNet, to handle the problem of skeleton-based

motion sequence recognition.

To sum up, deep learning methods have achieved great performance in many computer vision area and is applicable in motion data processing. Existing approaches on motion prediction, synthesis and recognition have achieved many successes. However, the complexity of human motion and variation of term and actors still cause challenges in motion understanding. A CNN model is designed for motion recognition purpose in this project.

Chapter 3

Motion Refinement: Dictionary Learning

In order to acquire accurate motion data, many MOCAP techniques such as mechanical, magnetic, optical and depth-based have been developed, where the most popular commercial MOCAP system are optical-based. However, even with those professional commercial optical based MOCAP systems, e.g. *Motion Analysis*[1] and *Vicon*[2], the captured results still suffer from the problem of missing marker. For instance, if some marker is occluded by objects or the actor's body, they become invisible from the camera, which caused the appearance of missing marker in the captured result. The most popular missing marker filling methods used in the commercial motion capture system are linear/non-linear interpolation methods. However, those methods are only efficient for simple and short-term noise cases. The refined result could become unrealistic while handling complex or long term missing cases.

As discussed in section 2.2, a lot of methods have been developed in the literature to improve the performance of motion refinement. However, it is still a challenge task due to the complexity and diversity of human motion.

Inspired by the success of sparse coding on motion data [Feng et al.

[1]http://www.motionanalysis.com/
[2]http://www.vicon.com/

2014a; Xiao et al. 2011, 2015], a sparse coding based motion refinement method is designed to tackle the problem of filling missing markers. Both the temporal information and kinematic structure of human motion data are accounted while designing the objective function, to ensure the visual quality of refined result. The methods presented in this chapter have appeared in [Wang et al. 2017b, 2016b].

3.1 Background of Sparse Representation

K-SVD dictionary learning K-SVD [Aharon et al. 2006; Bruckstein et al. 2009] dictionary learning is adopted in this design for sparse repsentation. K-SVD is a classical method for solving dictionary learning problem that fit the model adaptively and finally achieve sparsity representation. Generally, the sparse representation problem could be viewed as generalization of the vector quantization (VQ), where the dictionary of VQ codewords is typically learned via K-Means algorithms. Assume a dictionary D, the closest sparse representation y_i, i.e. using ℓ_2 distance, could be defined as:

$$y_i = Dx_i \tag{3.1}$$

where $x_i = e_j$ is the coefficient which is a vector from the trivial basis,with all zero entries except a one in the j-th position. K-means could be treated as a special case of spare coding that there is only one non-zero element in the coefficient x_i . The mean square error (MSE) is defined as

$$E = \sum_{i}^{K} e_i^2 = \|Y - DX\|_F^2 \tag{3.2}$$

The objective function of VQ is then written as:

$$\begin{aligned} &\min_{D,X} \|Y - DX\|_F^2 \\ &s.t. \forall i, x_i = e_k, \ for\ some\ k \end{aligned} \tag{3.3}$$

While using K-SVD rather than K-means, the objective function is rewritten as:

$$\begin{aligned} &\min_{D,X} \|Y - DX\|_F^2 \\ &s.t.\ \forall i,\ \|x_i\| \leq \epsilon \end{aligned} \tag{3.4}$$

The operations of solving sparse coefficient X and updating the dictionary D are taken iteratively. For the sparse coefficient solving problem, many existing methods could be applied, such as Orthogonal Matching Pursuit (OMP) [Donoho et al. 2012] , Basis Pursuit (BP) [Chen et al. 2001], FOCUSS [Gorodnitsky and Rao 1997], LLC [Wang et al. 2010] and so on.

For K-SVD, the dictionary updating is done column by column. Assume that the sparse coefficient X and the dictionary D is fixed, the $k-$th column of dictionary d_k is going to be updated. Let's denote the x_T^k as the $k-$th row of X, which will be used to multiply with d_k, the objective function of K-SVD could then be rewritten as

$$\begin{aligned} \|Y - DX\|_F^2 &= \|Y - \sum_{j=1}^{K} d_j x_T^j\|_F^2 \\ &= \|(Y - \sum_{j \neq k} d_j x_T^j) - d_k x_T^k\|_F^2 \\ &= \|E_k - d_k x_T^k\|_F^2 \end{aligned} \tag{3.5}$$

where the DX is composed as a summation of k rank-1 matrix components. Assume $k-1$ components are fixed, the left component is the one to be updated. E_k is the error caused by the other components of D excepted d_k.

If SVD is applied directly in this step to update the d_k and x_T^k, the closest to E_k rank-1 result matrix could be identified. However, in this case, the sparsity of updated x_T^k can not be prompted. In other word, the position and value of non-zero elements of x_T^k are not same with the previous one when d_k is not updated. Hence, to solve this problem, only non-zero elements are involved during SVD. Only non-zero elements of x_T^k and their production results with d_k in E_k, denoted as E_R^k, are kept.

Then, the SVD operation is applied to E_R^k to update d_k.

To sum up, K-SVD could guarantee the training error monotone decreasing, however, the size and sparsity of the dictionary should be set reasonably. Besides, in my experiment, the larger dictionary size could promote the performance, which is different from the sparsity. .

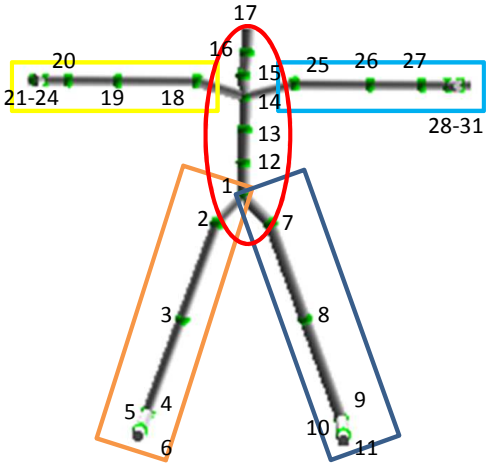
3.2 Data Preprocessing

To better explore the spatial-temporal information of the motion data and enhance the robustness of designed model, three preprocessing steps are designed, that are:

1. Normalization and coordinate translation;
2. Human partial model;
3. Grouping operation.

Normalization and Coordinate translation The motion capture data are recorded under the real-world global coordinate. Even visual-similar motion could have dramatically numerical diversity due to the pose translation and rotation. Thus, a local coordinate transformation would be applied to the raw data which aims to remove the effect of pose translation and rotation. Additionally, the torso is a rigid part in various kinds of motion. Therefore, each pose frame will be translated to the local coordinate representation respect to the root marker, i.e. marker 1 for CMU motion data. Then, the local pose frames will be rotated to ensure that the rigid plane, which is consisting of 3 markers, i.e.markers 2 (right femur), 7(left femur), and 14 (upper neck) for CMU motion data, parallels to the XY plane.

Poselet: Human Partial Model Due to its kinematic characteristics, the motion data intrinsically contains hierarchical spatial-temporal information. Although entire pose model representation are frequently



Partlet	Subset of Markers
Torso(TO)	1,12,13,14,15,16,17
Left Arm(LA)	14,25,26,27,28,29,30,31
Right Arm (RA)	14,18,19,20,21,22,23,24
Left Leg (LL)	1,7,8,9,10,11
Right Leg (RL)	1,2,3,4,5,6

Figure 3.1: *Partial model for CMU motion data. The markers 1, 2, 7, and 14 are the root, the right and left femur markers, and the upper neck marker, respectively, which are used for local coordinate translation.*

used Lai et al. [2011]; Feng et al. [2014b]; Li et al. [2010a]; Xiao et al. [2011], those global pose representations have two disadvantages: 1) the noisy body part will inevitably affect the clean body part, 2) the abundant similar local body part movement is hard to be exploited. Therefore, many researches have applied the partial human model [Feng et al. 2014a; Xiao et al. 2015; Guay et al. 2013; Peng et al. 2015; Huang et al. 2015] to approximate the spatial-temporal relationship while processing 3D motion data and achieved expressive performance.

In this work, the author has divided the whole body into 5 partlets [Feng et al. 2014a; Xiao et al. 2015], which is *Torso(TO), Left Arm (LA),Right Arm (RA),Left Leg(LL) and Right Arm(RL)*. An example of pose-let model for CMU motion data are shown in the Fig 3.1.

Therefore, for a given motion sequence $X = \{X_1, X_2, \ldots, X_S\}$ contains S pose frames, the submatrix X^i will be derived from X to represent each pose-let motion sequence, as $X^i = \{X_1^i, X_2^i, \ldots, X_S^i\} \in R^{d^i \times S}, i = 1, 2, \ldots, 5$.

Grouping Human motion data is a sequence of pose frames, where the continuous characteristics of motion, e.g. the trajectory, are embedded in

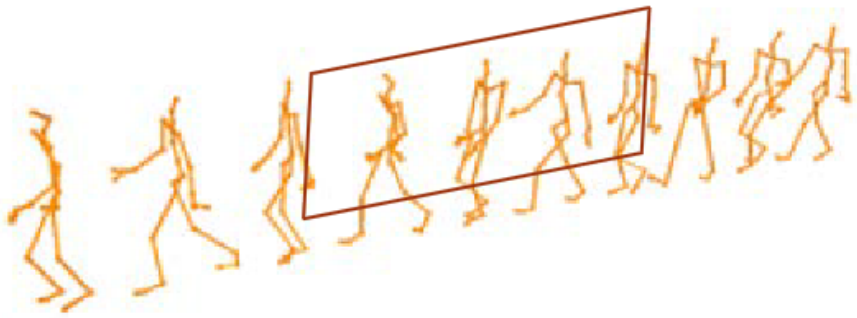


Figure 3.2: *Example of grouping operation.*

the local between-frame information. In other words, if it processes the refinement frame by frame, this information would be ignored. Hence, in this work, it uses a short clips of motion instead of single pose frame to better explore the embedded continuous between-frame information. For instance, considering a given pose-let sequence $X^i = \{X_1^i, X_2^i, \ldots, X_S^i\}$ stands for S pose frames with a lagging window of size M. It will generate $N = S - M + 1$ overlapping clips of motion while each clip contains M frames, that is $X(M)_j^i = [X_{(j-1)\times M+1}, \cdots, X_{j\times M}]$. In each clip, the M frames clip is reshaped into one vector Y_j^i, i.e. $R^{d_i \times M} \rightarrow R^{(d^i)\times 1}, d^i = M \times d_i$. Thus, the author finally get the groups of pose-let motion matrices $Y^i = \{Y_1^i, Y_2^i, \ldots, Y_N^i\} \in R^{d^i \times N}, N = S - M + 1, i = 1, 2, \ldots, 5$. An example of grouping operation is shown in Fig 3.2

3.3 Motion Dictionary Learning

Compare with other kinds of methods such as signal processing and matrix completion, data driven methods could achieve high quality performance even for complex motion. A new approach is proposed here to overcome the "smoothing" problem of refined result motion. For example, the character body may looks like shaking when the refined frames are combined together. Assume that the pose-let group set $Y^i = [Y_1^i, Y_2^i, \ldots, Y_N^i] \in R^{(d^i \times N)}, i = 1, 2, \cdots, 5$ stands for a piece of clean mo-

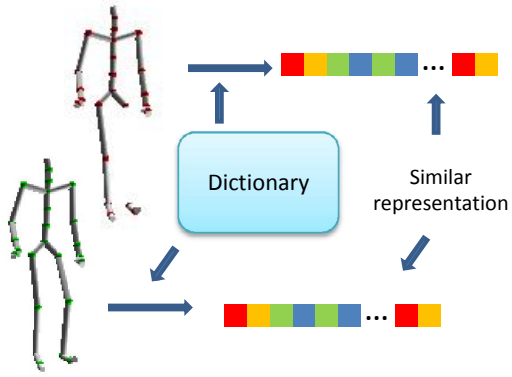


Figure 3.3: *Sparse representations of complete and incomplete frames. The difference between two representations should be minimized*

tion clip, where the partlet modeling and grouping operations have been processed. A conversation dictionary learning is to solve the following optimization problem

$$\begin{aligned} &\min_{W,D} \|Y^i - D^i W^i\|_F^2 \\ s.t. &W_j^i = [W_1^i, \ldots, W_N^i],\ \|W_j^i\|_0 \leq t_s, \\ &D^i = [D_1^i, \ldots, D_k^i],\ \|D_m^i\|_2 \leq 1, \\ &\forall\ i, 1 \leq j \leq N, 1 \leq m \leq K^i \end{aligned} \tag{3.6}$$

to extract the most suitable dictionary D^i for the sparse representation of ith pose-let training dataset Y^i, where W^i is the sparse coefficient matrix, K^i is the number of motion bases in the dictionary and t_s is the target sparsity. The columns of motion dictionary matrices D^i are the motion bases, which are normalized to unit ℓ_2 length for simplicity. However, equation 3.6 is a least square error function that is unstable with respect to the noise and outliers [Feng et al. 2013a]. The missed markers contained in the MOCAP data, may dominate the objective function 3.6 due to the square error. Hence, this objective function need to be enhanced.

As shown in Fig 3.3, it aims to minimize the difference of sparse representation between complete and incomplete pose, (i.e. minimize the reconstruction error of missing part). Let's denote the diagonal binary matrix $\Omega_j \in \{0,1\}^{d\times d}, j = 1, 2, \ldots S$ denotes the missing feature (i.e.

1 for corresponding marker miss) of the pose in j-th frame. Thus, for a given pose X_j, the missing part is $\Omega_i X_i$ while the observable part is $\overline{\Omega}_j X_j$. In order to match the pose-let training data, the missing marker would also be processed by the operations mentioned in previous section 3.2, the result markers are denoted as $\Omega^i, i = 1, 2, \cdots, 5$ corresponding to each pose-let group.

The ℓ_0 pseudo-norm in equation 3.6 is hard to solve, thus the author relaxes it to a ℓ_1 minimization. Recall the purpose of the proposed method shown in Fig 3.3, the objective function is then reformulated and the ideal dictionary would provide the sparse representation via satisfying

$$\begin{aligned} &\arg\min_{W_i} \|\overline{\Omega^i} Y^i - \overline{\Omega^i} D^i W^i\|_F^2 + \lambda \|W^i\|_1 \\ &s.t. D^i = [D_1^i, \ldots, D_k^i], \\ &\quad \|D_j^i\|_2 \leq 1, 1 \leq m \leq K^i \end{aligned} \tag{3.7}$$

A loss function L is defined for representing the reconstruction error of missing part:

$$L^i(Y_j^i, D^i, W_j^i) = \sum_{j=1}^{N} \frac{1}{2} \|\Omega_j^i Y_j^i - \Omega_j D^i W_j^i\|_F^2 \tag{3.8}$$

Thus,the dictionary learning objective function 3.6 is modified into

$$\begin{aligned} &\min_D \sum_j^N L^i(Y_j^i, D^i, W_j^i) \\ &s.t. \arg\min_{W_i} \|\overline{\Omega^i} Y^i - \overline{\Omega^i} D^i W^i\|_F^2 + \lambda \|W^i\|_1 \\ &\forall\ i, j = 1, 2, \ldots, N, \\ &D^i = [D_1^i, \ldots, D_k^i], \|D_m^i\|_2 \leq 1 \end{aligned} \tag{3.9}$$

For each pose-let sets $Y^i, i = 1, 2, \ldots, 5$, a corresponding D^i will be trained,respectively.

Generally, equation 3.9 is actually a nonconvex problem with respect to D^i and W^i jointly, which is difficult to find the global minimum.

However, equation 3.9 is convex with the three variable separately. In the following, the variables would be optimized alternatively. For the fixed W^i, equation 3.9 is equivalent to

$$\begin{aligned} &\min_{D^i} \frac{1}{2}\|\Omega^i Y^i - \Omega^i D^i W^i\|_F^2 \\ &s.t. D^i = [D_1^i, \ldots, D_k^i], \|D_m^i\|_2 \leq 1 \end{aligned} \tag{3.10}$$

which is a least squares problem with quadratic constraints. It could be solved by using gradient descent with iterative projection or be solved by using a Lagrange dual [Lee et al. 2006] much more efficiently. Once the motion dictionary D^i is updated, it turns to optimize W^i in Eq. 3.9:

$$\min_{W^i} \|\overline{\Omega^i} Y^i - \overline{\Omega^i} D^i W^i\|_F^2 + \lambda \|W^i\|_1 \tag{3.11}$$

which is a classical ℓ_1 minimization problem and it could be solved by using many existing methods [Yang and Zhang 2011; Donoho et al. 2012; Chen et al. 1998; Gorodnitsky and Rao 1997; Hale et al. 2008]

Therefore, W^i and D^i are updated iteratively until achieving convergence. The convergence conditions are set as:

$$\begin{aligned} &\frac{\|\Omega^i Y_{IT}^i - \Omega^i D_{IT}^i W_{IT}^i\|_F}{\|\Omega^i Y_{IT}^i\|_F} \leq \epsilon \\ &\frac{\|\Omega^i D_{IT}^i W_{IT}^i - \Omega^i D_{IT-1}^i W_{IT-1}^i\|_F}{\|\Omega^i D_{IT}^i W_{IT}^i\|_F} \leq \epsilon \end{aligned} \tag{3.12}$$

where IT denotes the iteration number, ϵ is the threshold. To simplify the problem, ϵ is set same during the training of all five motion dictionaries. The result five motion dictionaries $D^i, i = 1, 2, \cdots, 5$ will then be used for motion refinement. In practice, the training achieve convergence at around 30 iterations. Additionaly, the five motion dictionaries can also be trained in parallel.

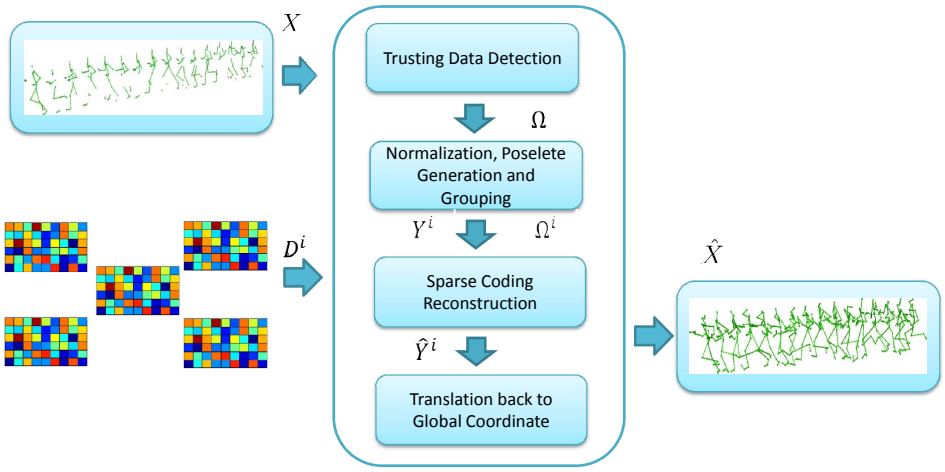


Figure 3.4: *Work flow of the proposed sparse basis selection method for human motion refinement*

3.4 Motion Recovery

Trust Data Detection As mentioned in the previous paragraph, apart from the dictionaries D and parameter λ, our approach also need to specify the missing marker Ω while processing the motion data. Here, the author employs a trust data detection (TDD) method [Feng et al. 2014b] to identify the missing data entries.

$$\begin{aligned} \Omega &= \text{TDD}(X, \phi) \\ s.t. \Omega &= [\Omega_1, \ldots, \Omega_S] \in \{0, 1\} \end{aligned} \tag{3.13}$$

where X is the given noisy motion sequence, Ω is the corresponding marker and ϕ is the threshold value which is set as $6cm$ in this work. The detail of TDD implementation is omitted, which is available in [Feng et al. 2014b].

Imperfect Motion Refinement For a given input imperfect motion sequence, we will firstly take the operations mentioned in section 3.2 to generate the five partial-group motion matrices, which denoted as $\{Y^i \in d^i \times N, i = 1, 2, \cdots, 5\}$. The corresponding missing mark Ω would be detected via TDD and also be translated to $\{\Omega^1, \cdots, \Omega^5\}$ using similar operations. In order to simplify the problem, the basis number K^i for

Algorithm 1 Sparse based motion refinement

Input: motion dictionary matrices D^i; the input imperfect motion sequence X_{global}; the length of moving window for grouping M; the regulation parameter λ; the threshold value ϕ.

Output: the refined motion sequence $\hat{X}_{global}$

1: **Coordinate translation**
change the unperfect motion sequence X_{global} into the local coordinate representation X_{local};

2: **Trust data detection**
generate the missing marker matric Ω via the TDD method shown in equation 3.13

3: **Poselet generation and grouping**
generate pose-lets group $\{Y^i, i = 1, 2, \cdots, 5\}$ with X_{local} and given window size M, generate the corresponding $\{\Omega^1, \cdots, \Omega^5\}$ via similar operations.

4: **Motion refinement**
with the motion dictionaries D^i, solve the sparse representation and rebuild the refined pose-let $\hat{Y}^i$ groups.

5: **Decompose groups and reconstruct the refined pose**
decompose the poslet groups $\hat{Y}^i$ and reconstruct the local pose frames $\hat{X}^i_{local}$ with the pose-lets of each frame.

6: **Form Motion Sequence and translate back to world coordinate**
form the refined local motion sequence $\hat{X}_{local}$ and translate it back into world coordinate representation motion sequence $\hat{X}_{global}$

each partial motion dictionary is all set as K, that is $\{D^i \in d^i \times K, i = 1, 2, \cdots, 5\}$. With the pre-trained five dictionary matrices $\{D^1, \cdots, D^5\}$, the reconstructed result groups $\{Y^i, i = 1, 2, \cdots, 5\}$ could be calculated by solving a $\ell_1 - norm$ minimization framework:

$$\arg\min_{W_i} \|\overline{\Omega}^i \circ Y^i - \overline{\Omega}^i \circ D^i W^i\|_F^2 + \lambda \|W^i\|_1 \tag{3.14}$$

Since the $\ell_1 - norm$ penalty in equation 3.14 on the coefficient W^i is not able to promise the smoothness of reconstructed result, a locality-constrained linear (LLC) coding method Wang et al. [2010] is used. Hence, the objective function is reformulated as

$$\arg\min_{W_i} \|\overline{\Omega}^i \circ Y^i - \overline{\Omega}^i \circ D^i W^i\|_F^2 + \lambda \|G^i \circ W^i\|_2 \tag{3.15}$$

where $G^i \in R^{K \times N}$ is the locality adaptor that each column gives the different freedom for each basis vector proportional to its similarity to the input descriptor $Y^i_{:,j}, j = 1, 2, \cdots, N$. Specifically,

$$\begin{aligned} G^i_{:,j} &= \exp\left(\frac{dist(\tilde{Y}^i_{:,j}, \tilde{D}^i)}{\sigma}\right) \\ \tilde{Y}^i &= \overline{\Omega}^i \circ Y^i, \;\; \tilde{D}^i = \overline{\Omega}^i \circ D^i W^i \end{aligned} \tag{3.16}$$

where $dist(Y^i_{:,j}, D^i) = [dist(Y^i_{:,j}, D^i_{:,1}), dist(Y^i_{:,j}, D^i_{:,2}), \cdots, dist(Y^i_{:,j}, D^i_{:,K})]^T$, and $dist(Y^i_{:,j}, D^i_{:,p})$ is the Euclidean distance between $Y^i_{:,j}$ and $D^i_{:,p}$. Each column of G^i is normalized to be between $(0, 1]$. Note that the LLC code in equation 3.15 is not sparse in the sense of ℓ_0 norm, but is sparse in the sense that the solution only has few significant values [Wang et al. 2010]. In practice, a threshold is applied to make those small coefficients be zero.

Solving the $\ell_1 - norm$ problem like equation 3.14 usually requires optimization procedures, e.g. Feature Sign algorithms [Yang et al. 2009], which is time consuming. Unlike equation 3.14, the solution of equation

3.15 can be derived analytically by:

$$\begin{aligned} \tilde{W^i_{:,j}} &= (C^i_j + \lambda\ diag(G^i)) \setminus 1 \\ W^i_{:,j} &= \tilde{W^i_{:,j}} / 1^T \tilde{W^i_{:,j}} \end{aligned} \tag{3.17}$$

where $C^i_j = (\tilde{D}^i - \tilde{Y}^i_{:,j}1^T)^T(\tilde{D}^i - \tilde{Y}^i_{:,j}1^T)$ denotes the data covariance matrix. Additionally,when the large size dictionary $D^i \in R^{d^i \times K}$ is used, a fast approximated method could be achieved by first performing a k-nearest neighbor ($k < d^i < K$) search and then solving a small constrained least square fitting problem, bearing computational complexity of $O(K + k^2)$ [Wang et al. 2010].

The overall work flow of proposed data-driven motion refinement method is shown in Fig 3.4. For a given input imperfect motion sequence, the corresponding five motion pose-let group matrices are generated via a using the preprocessing techniques that mentioned in previous section 3.2. At same time, the corresponding marker Ω will be detected and then translated to $\{\Omega^1, \cdots, \Omega^5\}$ via similar operations. With the pre-trained five dictionary matrices $\{D^1, \cdots, D^5\}$, the reconstructed pose-let groups $\{Y^i, i = 1, 2, \cdots, 5\}$ are calculated by solving equation 3.11. After the sparse reconstruction is finished, the pose-let group would be firstly used to recover the pose and then translated back to the world coordinate to reconstruct the final refined motion sequence $\hat{X}$.

3.5 Experimental Result and Discussions

Experimental Setup Due to the diversity of human motion, the quality of refinement could be affected by many factors, such as the action's category and the appearance frequency of missing marker. To evaluate the performance of the proposed method, the comparison is taken with other methods under various conditions. Four representative kinds of actions, i.e., *run*, *dance*, *boxing* and *basketball*, are chosen from CMU human motion database[3]. Two motion sequences from each category

[3]http://mocap.cs.cmu.edu/

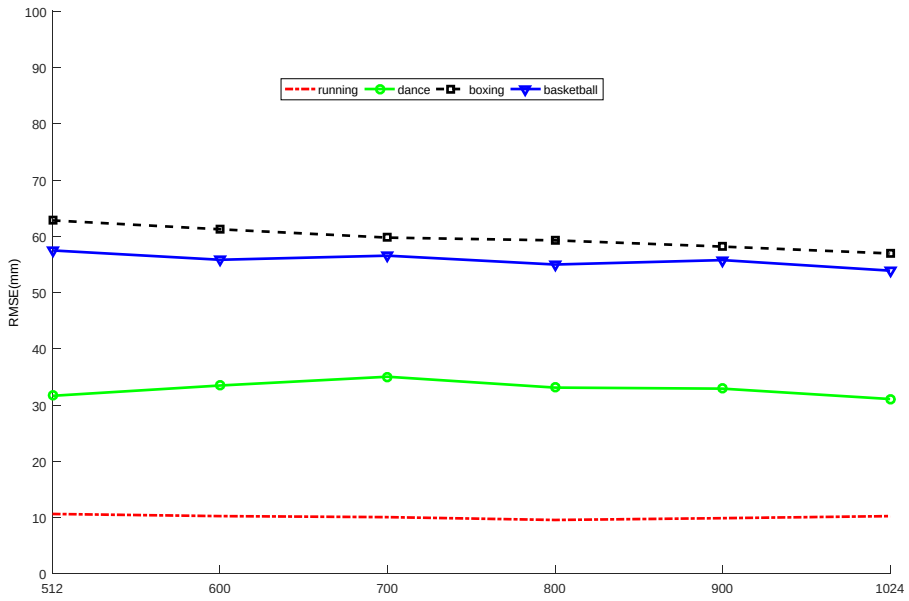


Figure 3.5: *Parameter tuning of dictionary size K while M is set as* 10

are randomly selected as testing set while others are used for training. Most of CMU motion data are very clear and would be directly used as the training data for our method. Similar testing data synthesis method is used that follow Feng et al. [2014b]; Xiao et al. [2011]. The noise with missing ratio from 10% to 30% with 10% interval. The size of the moving window M for grouping operation is tuned from $\{8, 10, 16, 24\}$. The parameter λ is tuned from $\{10^{-3}, 10^{-2}, 10^{-1}, 1, 10\}$ and finally set as 10. The dictionary size $K^i, i = 1, 2, \cdots, 5$ is set to be the same value and denoted as K. Finally M is set as 10 and K is set as 1024 due to the trade off between the effectiveness and efficiency, where the parameter tuning of K is shown in Fig 3.5. Besides, since D^i should be an over-complete dictionaries, K need to become larger when bigger M is used. In addition, M and K would also affect the computational cost, where the detail is discussed in section 3.5.

In the following, three exiting methods are implemented as comparison: dyanmmo [Li et al. 2009], a linear dynamic system(LDS) based method for predicting missing value; SRMMP [Xiao et al. 2011], a sparse coding based method for predicting missing markers; SVT [Lai

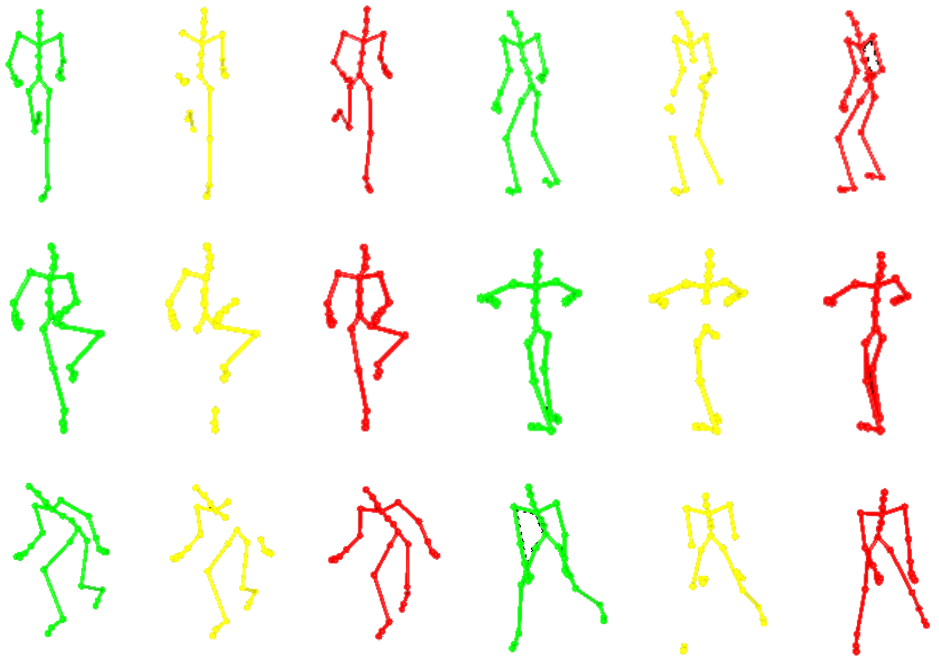


Figure 3.6: *Examples of recovery results for filling missing value: Original (green), Imperfact (yellow) and Refinement result (red)*

et al. 2011], a matrix completion based method. In order to make a fair comparison, the parameters for each algorithm are tuned by cross validation.

Experimental Results To evaluate the refinement performance, following the work [Feng et al. 2014a; Xiao et al. 2015; Li et al. 2009; Xiao et al. 2011; Baumann et al. 2011], the Root Mean Squared Error (RMSE) measurement is adopted:

$$\text{RMSE}(X_i, \hat{X}_i) = \sqrt{\frac{1}{n_e}\|X_i - \hat{X}_i\|^2} \tag{3.18}$$

where X_i is the original pose frame and $\hat{X}_i$ is the recovered one, n_e is the total number of missing markers in X_i. One motion sequence of each kind of motion is presented as an example.

The parameter are chosen via the cross validation. As shown in Fig 3.7, the result shows that our method outperform the competitors in

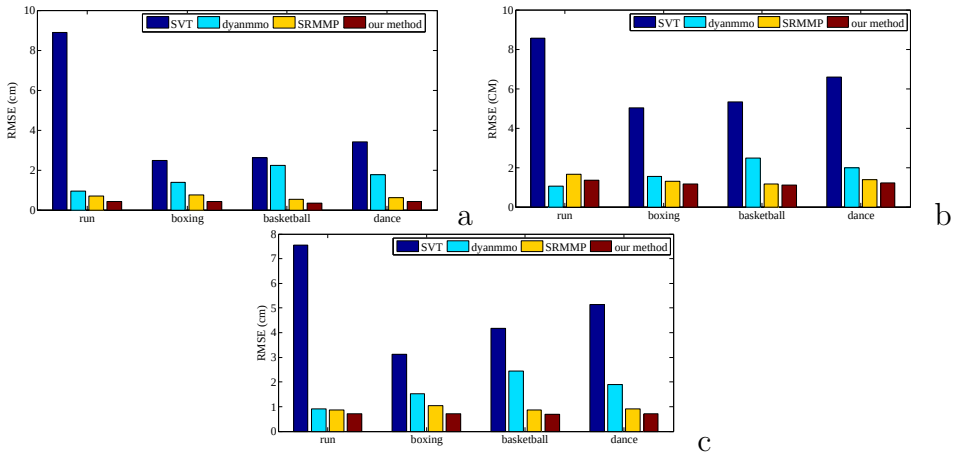
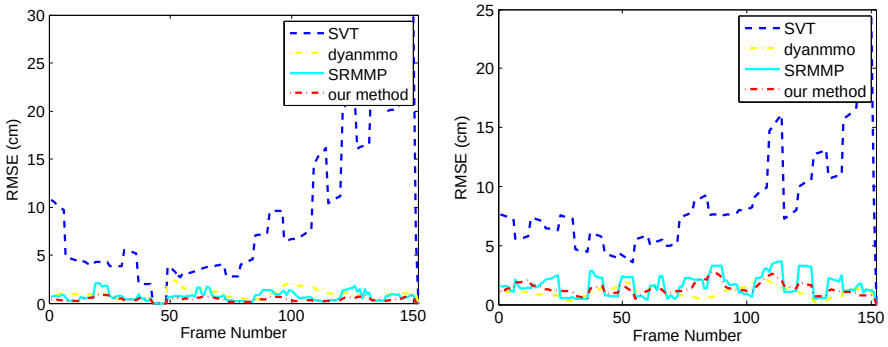
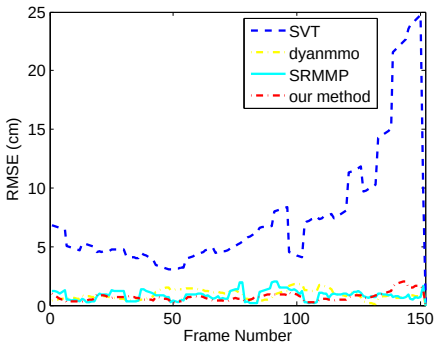


Figure 3.7: *The Comparisons of our method with other motion refining algorithms on four human motion sequences with missing rate 10%,20% and 30%. The average RMSE values of each frame(cm/frame) are reported.*

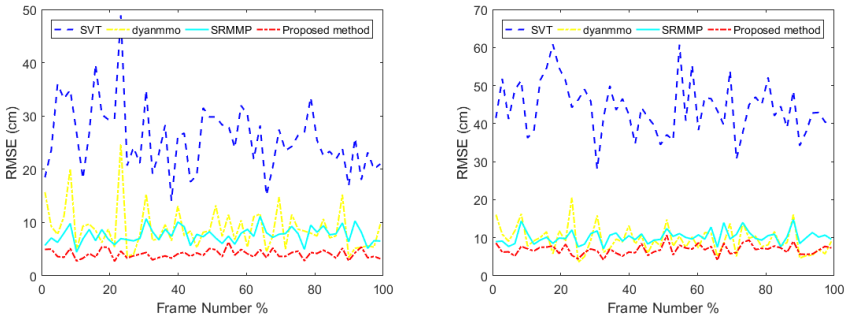


(a) (missing rate: 10%) (b) (missing rate: 20%)



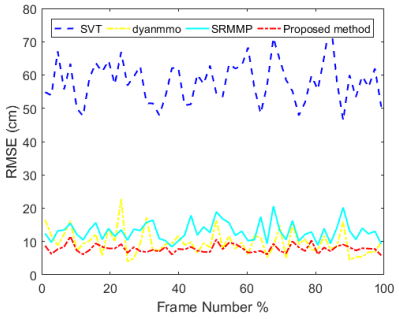
(c) (missing rate: 30%)

Figure 3.8: *Motion refinement comparisons of different algorithms on* **running** *with different missing rate.*



(a) (missing rate: 10%)

(b) (missing rate: 20%)

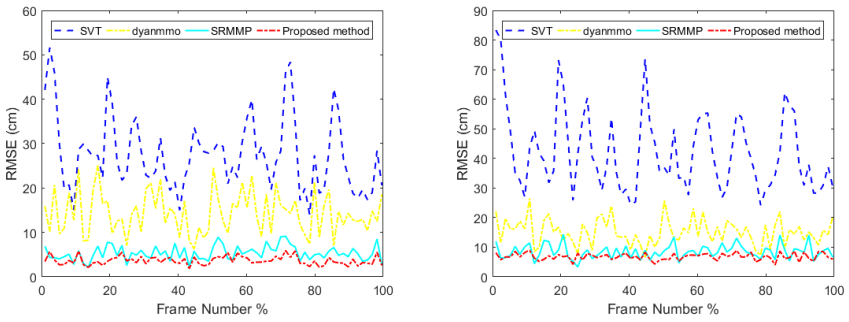


(c) (missing rate: 30%)

Figure 3.9: *Motion refinement comparisons of different algorithms on* **boxing** *with different missing rate.*

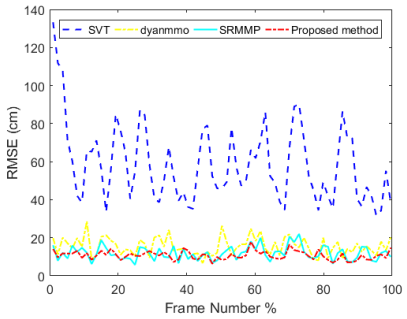
most cases, especially for the complicate motion and large amount of missing entries. It needs to be pointed out that the LDS method dyanmmo performs well in the simple and repetitive motion, e.g. running; our method perform better in other cases, especially for the complicate motion, e.g. basketball. In Fig 3.8, 3.9, 3.10 and 3.11, for each kind of action, the recovered result of one motion sequence is presented. The result shows that, in most of time, the variation of our algorithm's RMSE is smaller, which means that the performance of our method is more stable than the competitors. In other words, the refined motion from other methods are easy to cause the "shaking" artifacts.

Generally, our method could achieve a smooth and realistic result. The detail of the recovered motion could be seen in the demo video.



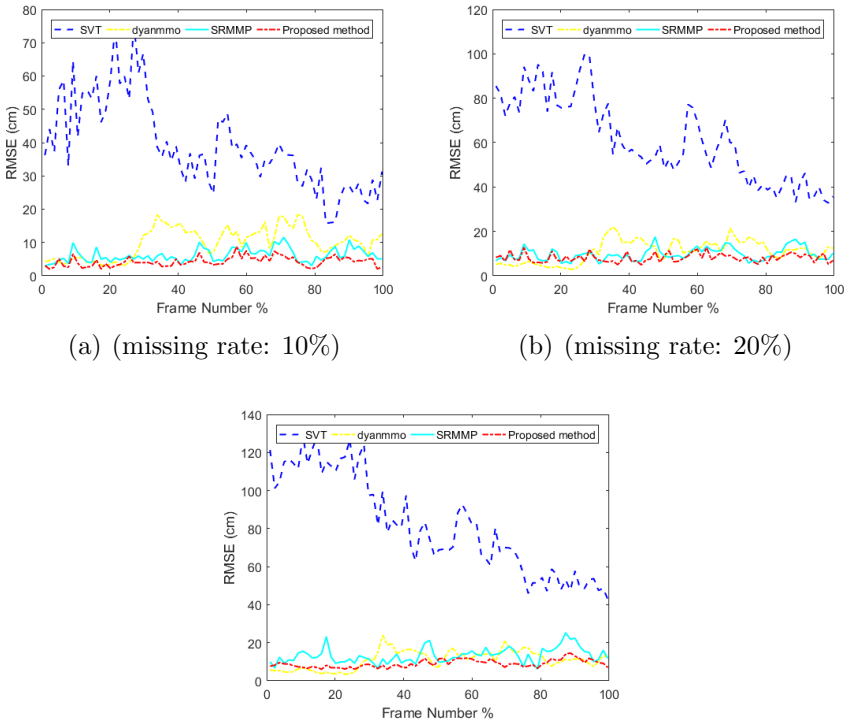
(a) (missing rate: 10%)

(b) (missing rate: 20%)



(c) (missing rate: 30%)

Figure 3.10: *Motion refinement comparisons of different algorithms on* **basketball** *with different missing rate.*



(a) (missing rate: 10%) (b) (missing rate: 20%)

(c) (missing rate: 30%)

Figure 3.11: *Motion refinement comparisons of different algorithms on* **dancing** *with different missing rate.*

Discussion

Computational complexity analysis. The computational cost of the proposed method are mainly from two steps operations:

- Learning dictionaries D^i: If fast K-SVD [Rubinstein et al. 2008] is applied to solve the calculation of D^i, the time complexity is $O(N_\Omega \times N_p \times [(t_s)^2 + 2d^i \times M] \times K^i)$, where N_p is the number of pose-let groups for training, N_Ω is number of missing markers for each pose frame, which is used for training data augmentation.

- Solving W for sparse representation: The dictionary learning just need to be implemented for one time. Hence, the computational cost for refining a motion sequence mainly comes from the sparse coefficient calculation. which is about $O(K + k^2)$ (k nearest neighbor searching, $k < d^i < K$) by using a fast LLC method Wang

et al. [2010].

In practice, it could assume an asymptotic behaviour that $t_s < K^i << (N_p \times N_\Omega)$ and $(d^i \times M) < K$, noticed that $K^i, i = 1, \cdots, 5$ is set to be the same value K. The first part for dictionary learning just need to be implemented for once time. Therefore, the computational cost for refining a motion sequence mainly comes from the second part. Hence, the overall computational cost is reasonable. In addition, the processing for each kind of pose-let is independent, therefore, those process could be taken in parallel in both training and refining stage, which could reduce the time consuming in practice.

Denoising In this work, the proposed method just focus on solving the missing marker problem. However, in practice, the MoCap data may also contains the noise, e.g. Gaussian noise. It's known that the ℓ_2 norm is effective for dealing with this kind of noises. In addition, in [Xiao et al. 2015], it has presented that the ℓ_2 denoising and ℓ_1 missing filling could be combined together. In other words, to refine a piece of imperfect motion data sequence, the ℓ_1 normalization, e.g. equation 3.11, could be applied firstly to fill the missing value, then the ℓ_2 normalization, e.g. using ℓ_2 norm instead of ℓ_1 norm in equation 3.11, could be applied to remove the Gaussian noise.

Limitations and future work. Although learning spatio-temporal dictionary has been addressed in literatures, the input data in this design is carefully organized. In this work, the global pose representation is replaced with the partial model and group operation is applied in each motion sequence using a lagged window to generate pose-lets-groups, which preserve the embedded spatio-temporal patterns of human motion.

Although the sampling of pose-lets is not addressed in this work, this problem can be handled by adding a data preprocessing step with DTW. The correlation among body part has been taken into account via allowing some adjacent joints belonging to multiple pose-lets. The foot-skating removal term could be added in the data post-precess step in the future.

One limitation of the proposed method is that it needs clean motion for training. Hence, for uncleaned training data, both the distribution of missing marker and noise will be considered for dictionary training in the further objective function design. Besides, an additional optimization step for an ℓ_2 normalization is required to deal with common gaussian noise. Thus, in the future work, it will try to combine the refining of noise and missing marker in one objective function. Moreover, the TDD method is based on the assumption that the motion is smooth in the feature space. In some extreme cases which contains sudden changes, it may not work well. Kicking is an example, where the sudden change of velocity, direction and acceleration affect the performance of TDD and refined result. An example demo video is available online [1].

3.6 Summary

To sum up, human motion refinement is an essential step for MOCAP data based applications. A locality sparse coding based motion refinement method is proposed in this chapter. The proposed method can be regarded as an extension of the sparse coding framework with penalties on the dictionary D and the sparse coefficient W. Both hierarchical characteristics and spatial temporal information of motion data are considered while designing the objective function. The LLC coding and grouping operation ensure the smooth property of the recovered result. In addition, the partial model enhance the robustness of designed method. The experimental result shows that the proposed method outperforms the state-of-art method in various cases.

[1] `https://www.youtube.com/watch?v=2E7QNMNmL3E&t=204s`

Chapter 4

Motion Refinement: Low-Rank Matrix Completion

In this chapter, the low-rank matrix completion based motion data refinement method is presented. The presented methods in this chapter have appeared in [Hu et al. 2017b,a]. where the author has contributed to the model building and algorithm design. Besides, the designed bilinear factorization model that presented in the first part of this chapter has also been used in face motion refinement that appeared in [Wang et al. 2017a].

4.1 Low-Rank based Data Refinement model

For a given MoCap data sequence, let's denote it as $X = [x_1.x_2, \cdots, x_n] \in \Re^{d\times n}$, where $x_i \in \Re^{d\times 1}$ is the ith frame, d is the number of features, e.g. location, velocity, acceleration, etc. and n is the number of samples. Inspired by Feng et al. [2014b], the proposed low-rank based refinement model is developed base on an assumption that the noise is sparse in the observed part, e.g. $Y + E = X$, where Y is the clean data and E is the corresponding noise/outlier. This is a weak assumption for enhancing the robustness of the model, but it is reasonable since the observable da-

ta from the professional MoCap systems usually contains a little amount of noise and outlier. Here, the proposed model is shown as follows:

$$\begin{aligned} &\min_{Y,E} \ rank(Y) + \alpha\|\Omega \circ E\|_0 + \frac{\beta}{2}\|YO\|_F^2 \\ &s.t. \ X = Y + E \end{aligned} \tag{4.1}$$

where $\circ$ is the dot product of matrix, $\Omega \in \{0.1\}^{d\times n}$ is a mask matrix that 1 stands for observable entry and 0 stands for missing entry. The tridiagonal symmetrical square matrix $O \in \Re^{n\times n}$ is used to ensure C^2 continuity on every landmark's trajectory, which defined as:

$$O = \begin{bmatrix} -1 & 1 & & & & \\ 1 & -2 & 1 & & & \\ & \ddots & \ddots & \ddots & & \\ & & \ddots & \ddots & \ddots & \\ & & & 1 & -2 & 1 \\ & & & & -1 & 1 \end{bmatrix} \tag{4.2}$$

Since Eq. 4.1 is NP-hard due to the discontinuous and non-convex nature of the rank function and ℓ_0 norm, the nuclear norm is employed to replace the rank function and relax the ℓ_0 norm to ℓ_1 norm. Therefore, it yields the following optimization model:

$$\begin{aligned} &\min_{Y,E} \ \|Y\|_* + \alpha\|\Omega \circ E\|_1 + \frac{\beta}{2}\|YO\|_F^2 \\ &s.t. \ X = Y + E \end{aligned} \tag{4.3}$$

In this work, bilinear factorization method is used to solve Y. For a given data matrix $A \in \Re^{d\times n}$, the bilinear factorization aims to find two low-rank matrices $U \in \Re^{d\times k}$ and $V \in \Re^{n\times k}$ that $A = UV^T$. Here k is the upper bound on the rank of A, i.e. $k \geq rank(A)$. The nuclear norm is adopted:

$$\|A\|_* = \min_{A=UV^T} \frac{1}{2} n(\|U\|_F^2 + \|V\|_F^2) \tag{4.4}$$

Since $n >> d$ in most cases, the author can choose to study the trans-

position version of Eq.4.3 to avoid large computing cost of SVD:

$$\begin{aligned} &\min_{Y,E} \ \|Y^T\|_* + \alpha\|\Omega^T \circ E^T\|_1 + \frac{\beta}{2}\|OY^T\|_F^2 \\ &s.t.\ X^T = Y^T + E^T \end{aligned} \tag{4.5}$$

where d is usually a fixed value, e.g. $3 \times Numberof Joints$, which can be omitted. Refer to Eq. 4.4, it can be reformulated as:

$$\begin{aligned} &\min_{U,V,E} \frac{1}{2}(\|U\|_F^2 + \|V\|_F^2) + \alpha\|\Omega^T \circ E\|_1 \\ &\qquad + \frac{\beta}{2}\|OUV\|_F^2 \\ &s.t.\ X^T = UV + E \end{aligned} \tag{4.6}$$

where $U \in \Re^{n\times k}$, $V \in \Re^{d\times k}$ and $E \in \Re^{n\times d}$.

4.1.1 Optimization

The augmented Lagrange multiplier (ALM) method is used to solve the objective function shown in Eq.4.6. Here, it introduces a slack matrix $M = Y = UV$ and an equivalent form of Eq.4.6 can be written as:

$$\begin{aligned} &\min_{U,V,E} \frac{1}{2}(\|U\|_F^2 + \|V\|_F^2) + \alpha\|\Omega^T \circ E\|_1 \\ &\qquad + \frac{\beta}{2}\|OM\|_F^2 \\ &s.t.\ X^T = UV + E, \quad M = UV. \end{aligned} \tag{4.7}$$

Then it leads to the augmented Lagrangian function:

$$\begin{aligned} \mathcal{L}(U,V,E,M,Y_1.Y_2) \ &= \frac{1}{2}(\|U\|_F^2 + \|V\|_F^2) \\ &+ \alpha\|\Omega^T \circ E\|_1 + \frac{\beta}{2}\|OM\|_F^2 + \langle Y_1, M - UV\rangle \\ &+ \frac{\lambda_1}{2}\|M - UV\|_F^2 + \langle Y_2, X^T - UV - E\rangle \\ &+ \frac{\lambda_2}{2}\|X^T - UV - E\|_F^2 \end{aligned} \tag{4.8}$$

where $Y_1, Y_2 \in \Re^{n\times d}$ are lagrangian multipliers, λ_1, λ_2 are regulation parameters. Eq.4.8 is solved by minimizing each variable alternatively while fixing the other variables. Therefore, the optimization is divided into four sub-problems of U, V, E, M and two multipliers Y_1, Y_2 updating:

Fix others, update U To solve the variable U , the author fixes the other variables and solve the following least-square problem:

$$\begin{aligned} U^{(p+1)} &= \arg\min_{U\in\Re^{n\times k}} \mathcal{L}(U, (V, E, M, Y_1, Y_2)^{(p)}) \\ &= \arg\min_{U} \frac{1}{2}\|U\|_F^2 + \frac{\lambda_1}{2}\|M^{(p)} - U(V^{(p)})^T + \frac{Y_1^{(p)}}{\lambda_1}\|_F^2 \\ &\quad + \frac{\lambda_2}{2}\|X^T - U(V^{(p)})^T - E^{(p)} + \frac{Y_2^{(p)}}{\lambda_2}\|_F^2 \\ &= Z_u V^{(p)}[I_k + (\lambda_1+\lambda_2)(V^{(p)})^T V^{(p)}]^{-1} \end{aligned} \tag{4.9}$$

where $Z_u := Y_1^{(p)} + Y_2^{(p)} + \lambda_1 M^{(p)} + \lambda_2(X^T - E^{(p)})$

Fix others, update V Similarly, the variable V is updated as follows:

$$\begin{aligned} V^{(p+1)} &= \arg\min_{V\in\Re^{n\times k}} \mathcal{L}(U^{(p+1)}, V, (E, M, Y_1, Y_2)^{(p)}) \\ &= Z_v U^{(p+1)}[I_k + (\lambda_1+\lambda_2)(U^{(p+1)})^T U^{(p+1)}]^{-1} \end{aligned} \tag{4.10}$$

where $Z_v := Y_1^{(p)} + Y_2^{(p)} + \lambda_1 M^{(p)} + \lambda_2(X^T - E^{(p)})$

Fix others, update E Then, the variable E is updated as follows:

$$\begin{aligned} E^{(p+1)} &= \arg\min_{V\in\Re^{n\times k}} \mathcal{L}(U^{(p+1)}, V^{(p+1)}, E, (M, Y_1, Y_2)^{(p)}) \\ &= \arg\min_{E} \alpha\|\Omega^T \circ E\|_1 + \frac{\lambda_2}{2}\|X^T - U^{(p+1)}(V^{(p+1)})^T - E^{(p)} + \frac{Y_2^{(p)}}{\lambda_2}\|_F^2 \\ &= \Omega^T \circ S_{\alpha/\lambda_2}(Z_e) + \hat{\Omega}^T \circ (Z_e) \end{aligned} \tag{4.11}$$

where $Z_e := X^T - U^{(p+1)}(V^{(p+1)})^T + \frac{Y_2^{(p)}}{\lambda_2}$. The operation S_τ is defined as

$$S_\tau = \arg\min_A \tau\|A\|_* + \frac{1}{2}\|A - B\|_F^2 \tag{4.12}$$

for any $\tau \geq 0$.

Fix others, update M After that, the variable M is updated as follows:

$$\begin{aligned} M^{(p+1)} &= \arg\min_M \frac{\beta}{2}\|OM\|_F^2 \\ &+ \frac{\lambda_1}{2}\|M - U^{(p+1)}(V^{(p+1)})^T + \frac{Y_1^{(p)}}{\lambda_1}\|_F^2 \\ &= (\lambda_1 I_n + \beta O^2)^{-1}(\lambda_1 U^{(p+1)}(V^{(p+1)})^T - Y_1^{(p)}) \end{aligned} \tag{4.13}$$

Update Lagrange multipliers Next, the Lagrange multipliers Y_1, Y_2 are updated as follows:

$$\begin{aligned} Y_1^{(p+1)} &\leftarrow Y_1^{(p)} + \lambda_1(X - U^{(p+1)}(V^{(p+1)})^T \\ &\quad - E^{(p+1)}) \\ Y_2^{(p+1)} &\leftarrow Y_2^{(p)} + \lambda_2(M^{(p+1)} - U^{(p+1)}(V^{(p+1)})^T) \end{aligned} \tag{4.14}$$

The ALM will converge to the optimal solution as proved in [Lin et al. 2010]. Till now,the updating rules of U, V, E, M, Y_1, Y_2 are shown. The optimization method is summarized in Algorithm 2. Besides, the outlier mask Ω is required for the proposed model, where a b-spline based trust detection (TDD) method [Feng et al. 2014b] is employed to fix this problem.

4.1.2 Rank estimation

Since a proper estimation to the rank of the MoCap data is essential for the bilinear factorization. Here, the concept of last significant jump (LSJ) rule that proposed in [Wang and Yin 2010; Wang and Su 2014] is employed to adaptively estimate the rank. Indeed, the basic principle of

Algorithm 2 Optimization of proposed bilinear factorization based method

Input: $X, \Omega, O, \alpha, \beta, \lambda_{\max}$

1: **Initialize:** Estimate the rank r and compute κ; p = 0; $Y_1^{(0)} = Y_2^{(0)} = X^T / \max(\|X^T\|_F^2, \|X^T\|_\infty)$; $V^{(0)} = rand(d, \kappa)$, $V^{(0)} = \frac{V^{(0)}}{\|V^{(0)}\|_F}$; $E^{(0)} = M^{(0)} = 0$; $\lambda_1 > 0, \lambda_2 > 0$

2: **while** not converge **do**

3: Update U^{p+1} using Eq.(4.9) ;

4: Update V^{p+1} using Eq.(4.10) ;

5: Update E^{p+1} using Eq.(4.11) ;

6: Update M^{p+1} using Eq.(4.13) ;

7: Update the multipliers Y_1, Y_2 using Eq.(4.14)

8: Update the regulation parameter with a positive scalar $\gamma > 1$:
$\lambda_1 \leftarrow \min(\gamma\lambda_1, \lambda_{\max})$,
$\lambda_2 \leftarrow \min(\gamma\lambda_2, \lambda_{\max})$;

9: $p = p + 1$;

10: **end while**

Output: $Y \leftarrow V^{(p)}(U^{(p)})^T$, $E \leftarrow (E^{(p)})^T$.

LSJ is to detect the support of a sparse vector consisting of eigenvalues whose cardinality is the rank of a low-rank matrix.

Since the given MoCap data X is incomplete, an initial guess for the missing markers is given by the linear interpolation method and thus $\tilde{X}$ is obtained. Next, the LSJ rule based on thresholding is applied estimate the rank of $\tilde{X}$. in details, let $\Lambda = (\lambda_1, \lambda_2, \cdots, \lambda_k)$, with $k = \min(3J, n)$ be the singular values of $\tilde{X}$ satisfying $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_k \geq 0$.. The LSJ rule is to look for the largest index k_0 such that $\|\lambda_{k_0} - \lambda_{k_0+1}\| > \epsilon$m and then define $r = \text{Rank} \simeq k_0$, where ϵ is a threshold value which is set as $\epsilon = 0.1$ in this approach. Finally, considering that X also contains noises and outliers, the rank estimated above may be a little conservative, and thus a parameter κ is specified that satisfying $\kappa \geq r$ as

$$\kappa = \min\{(1.1 \times r), 3J, n\} \tag{4.15}$$

where J is the number of joints.

4.1.3 Experimental Results

In this section, the result of comparison with TSMC [Feng et al. 2014b] will be shown, since they have already shown that TSMC performs more effectively than many existing approaches, such as linear interpolation, spline interpolation, Dynammo and SVT. Six motion sequences (walk, gymnastics, dance, acrobatics, basketball and boxing) from CMU mocap dataset are selected to evaluate the performance. Inspired by [Feng et al. 2014b], four classical situations are simulated to synthesize four different kinds of corrupted data, which are briefly listed as follows: Randomly corrupt data (rdcrupt), Randomly lose data (rdlose), Mixed corrupt data (mxcrupt) and Regularly lose data (rglose).

The parameters in the proposed method are set as follows for all experiments: $\alpha = 1.0$, $\beta = 100$, $\lambda_1 = \lambda_2 = 10^{-5}$, $\gamma = 1.4$, and $\lambda_{max} = 10^{10}$. To verify the refining accuracy, the root mean square error (RMSE) measurement is adopted:

$$\text{RMSE}(x_i, \tilde{x}_i) = \sqrt{\frac{1}{n_p}\|x_i - \tilde{x}_i\|^2} \tag{4.16}$$

where x_i and $\tilde{x}_i$ correspond to the imperfect and refined poses, respectively, and n_p is the total number of imperfect entries, i.e. the missing and noise entries.

For the four cases (rdcrupt, rdlose, mxcrupt and rglose), the RMSEs of TSMC (in red color) and the proposed method (in blue color) were shown in Fig 4.1 to Fig 4.4 respectively. The experimental results show that the proposed approach outperforms TSMC for most of the refined results in each case. The reason lies in that our approach exploits the true low-rank property of mocap data by using bilinear factorization, while TSMC depends on partial SVD (pSVD) which causes the information corresponding to the singular values under the pSVD specified threshold dropped.

Generally, an efficient MoCap data refinement method based on the matrix bilinear factorizaon and inverse discrete cosine transform is pre-

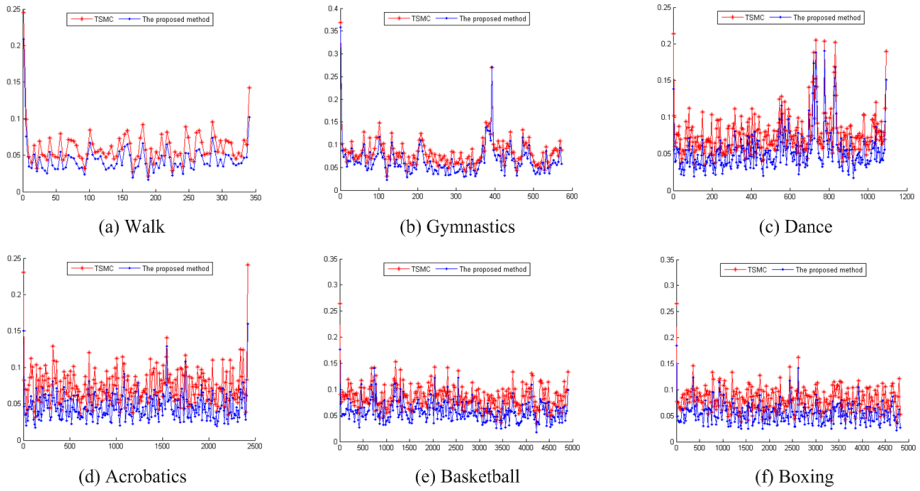


Figure 4.1: *Comparisons of refined results for the case of randomly corrupt data (rdcrupt): Gaussian noises ($\sigma = 2$) were randomly added on* 30% *data for each motion sequences, wherein x-label denotes the frame index and y-label the RMSE of each frame (cm/frame).*

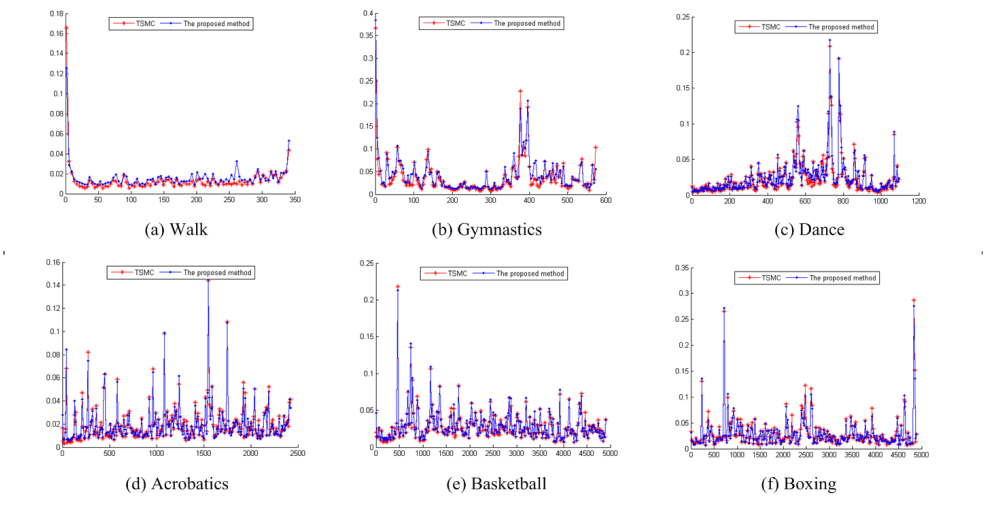


Figure 4.2: *Comparisons of refined results for the case of randomly lose data (rdloss):* 30% *data of each motion sequence were randomly missing, wherein x-label denotes the frame index and y-label the RMSE of each frame (cm/frame).*

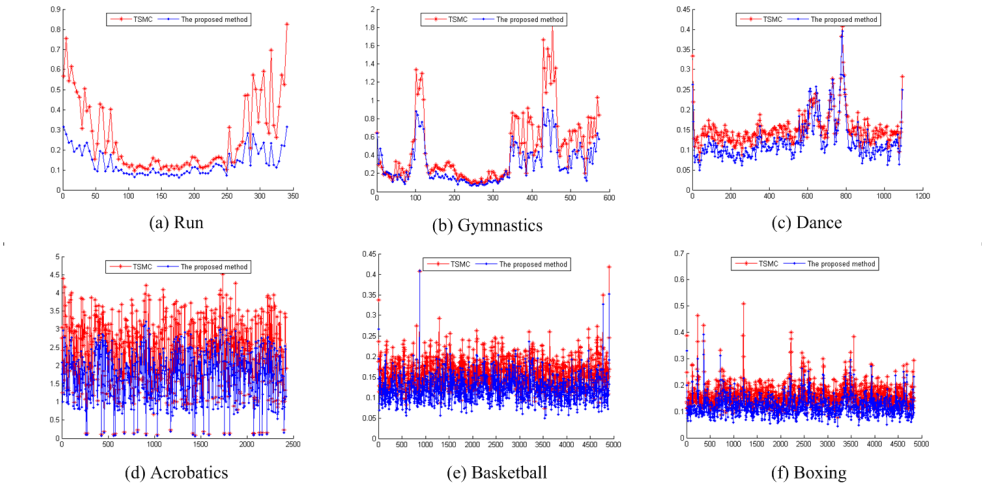


Figure 4.3: *Comparisons of refined results for the case of mixed corrupt data (mxcrupt):* 30% *data were randomly missing and then* 30% *of the remaining data were corrupted by Gaussian noise, wherein x-label denotes the frame index and y-label the RMSE of each frame (cm/frame).*

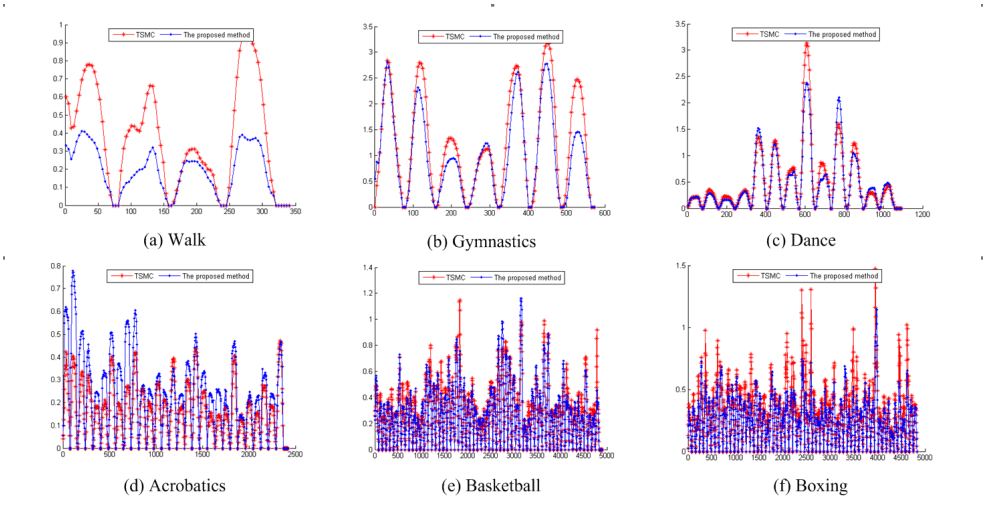


Figure 4.4: *Comparisons of refined results for the case of regularly lose data (rgloss):* 30% *data were randomly removed wherein the number of selected missing markers is fixed to be 10 and each missed 60 frames, wherein x-label denotes the frame index and y-label the RMSE of each frame (cm/frame).*

sented here. The proposed method outperforms TSMC. However, from the results in Fig 4.4, it finds that both of the methods have a sharp decline in performance for the case of regularly and continuous lose data, which needs to be explored further.

4.2 Improved approach based on Truncated Nuclear Norm (TrNN)

Recall that the low-rank matrix completion model to recover the MoCap data can be defined as:

$$\min_{X} \|X\|_* \quad s.t. P_\Omega(X) = P_\Omega(D) \tag{4.17}$$

where $\|\|\|_*$ denotes the nuclear norm of a matrix, $D \in \mathbb{R}^{m \times n}$ is the observed incomplete motion data, each column of D represnts the 3D coordinates of body markers in each frame, e.g. $m = 3 \times$ marker number , $n =$ frame number. X is the corresponding complete and clean motion data and P_Ω denotes the orthogonal projection of a matrix onto the subspace of matrices which has non-zero entries corresponding to the observed entries in Ω and 0 otherwise.

Due to the application of nuclear norm, existing matrix completion approaches have two major limitations. On one hand, the employed iterative solution method involves the expensive computational task of singular value decomposition (SVD) at each iteration [Lin et al. 2010; Cai et al. 2010; Kang et al. 2015; Parekh and Selesnick 2016], which becomes increasingly costly as the frame numbers of motion sequences grow. On the other hand, nuclear norm minimization makes all of the singular values simultaneously minimized, and thus the rank may not be well approximated in practice [Cao et al. 2017].

As shown in Fig 4.5, the information of motion sequences is commonly dominated by the top $r(\leq 30)$ singular values. Motivated by this observation, a novel MoCap data completion method by replacing the nuclear norm with a new matrix norm is proposed, called truncated nuclear nor-

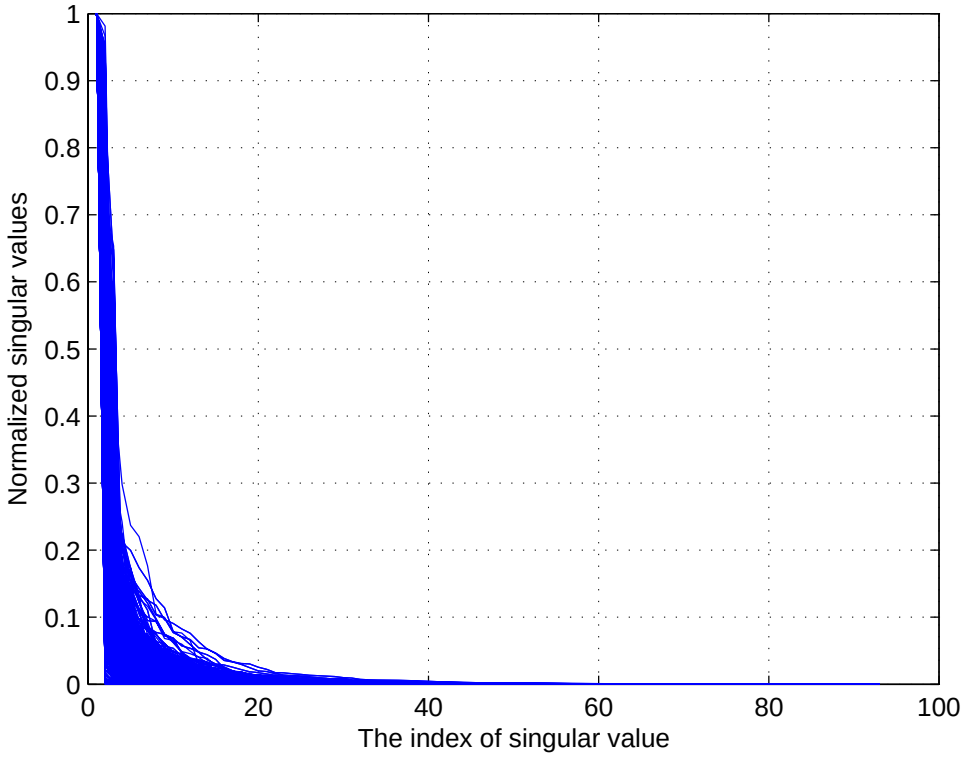


Figure 4.5: *Normalized singular values of motions collected from CMU MoCap database which totally includes 2605 motion sequences. All the singular values are normalized to* $[0, 1]$. *Its shown that the main information roughly lies in the first 30 largest singular values.*

m (TrNN) which is defined as the sum of the smallest $(\min(m, n) - r)$ singular values.

Till now, TrNN has been successfully applied in many fields, such as image inpainting [Hu et al. 2013b; Liu et al. 2016b; Zhang et al. 2012], multi-class classification [Hu et al. 2015], photometric stereo Oh et al. [2016, 2013a] and high dynamic range imaging [Oh et al. 2013b; Lee and Lam 2016]. This owes to that TrNN based minimization encourages the results to have a rank close to a specified value r, since TrNN is defined as the partial sum of the $\min(m, n) - r$ smallest singular values. It's noteworthy that the rank of motion matrices mainly concentrates on the first few singular values (see Fig. 4.5), meaning TrNN has strong potential to accurately constrain the rank of MoCap data, which is also the motivation of this work. To the best of my knowledge, it's the first time to use TrNN studying MoCap data completion.

In this section, an improved TrNN based algorithm is devised to solve mocap data completion problem, where each step admits closed-form solutions. The proposed approach outperforms conventional methods in not only computation efficiency but also completion accuracy.

4.2.1 Truncated nuclear norm regularization

Objective function

Given an incomplete motion matrix $D \in \mathbb{R}^{m\times n}$, the RPCA attempts to decompose D as the sum of the a low-rank matrix $X \in \mathbb{R}^{m\times n}$ concerning the latent ideal motion and a sparse error matrix $S \in \mathbb{R}^{m\times n}$. Further, by replacing the nuclear norm with the truncated nuclear norm (TrNN) [Zhang et al. 2012], the RPCA problem becomes

$$\min_{X,S} \|X\|_r + \lambda\|S\|_1, \text{ s.t. } \mathcal{P}_\Omega(X+S) = \mathcal{P}_\Omega(D), \tag{4.18}$$

where $\|X\|_r = \sum_{i=r+1}^{\min(m,n)}\sigma_i(X)$ for $0 \le r \le \text{rank}(X)$, $\sigma_i(X)$ denotes the ith singular value of X, and $\mathcal{P}_\Omega$ denotes the orthogonal projection of a matrix onto the subspace of matrices which has non-zero entries corresponding to the observed elements in Ω and 0 otherwise. For the optimal solution of Eq. (4.18), it can easily prove $S_{\Omega^c} = 0$ by contradiction, which leads to $\|S\|_1 = \|\mathcal{P}_\Omega(S)\|_1$, where Ω^c denotes the complement of Ω. Therefore, an equivalent but more easily solved form can be gained by removing the projection operations on X and S in constraints and assuming they take opposite values on Ω^c:

$$\min_{X,S} \|X\|_r + \lambda\|\mathcal{P}_\Omega(S)\|_1, \text{ s.t. } X+S = \mathcal{P}_\Omega(D), \tag{4.19}$$

Inspired by the work [Hu et al. 2013b; Liu et al. 2016b; Hong et al. 2016], the TrNN can be written as an equivalent matrix factorization

form

$$\begin{aligned} \|X\|_r = &\min_{L,R,U,V} \frac{1}{2}(\|L\|_F^2 + \|R\|_F^2) - tr(ULR^TV^T), \\ &\text{s.t. } X = LR^T, UU^T = VV^T = I_r, \end{aligned} \tag{4.20}$$

where $L \in \mathbb{R}^{m\times d}$, $R \in \mathbb{R}^{n\times d}$, $U \in \mathbb{R}^{r\times m}$ and $V \in \mathbb{R}^{r\times n}$ for any $r \leq \text{rank}(X) \leq d$, and I_r denotes the $r \times r$ identity matrix. The advantage of using Eq. 4.20 to define TrNN is that large-scale SVD computation is avoided when solving Eq. (4.19).

By combining Eqs. (4.20) and (4.19), and additionally taking into account the assumption of temporal C^2 continuity of human motion [Feng et al. 2014b], the overall objective function is

$$\begin{aligned} \min_{L,R,S,U,V} \quad &\frac{1}{2}(\|L\|_F^2 + \|R\|_F^2) - tr(ULR^TV^T) \\ &+ \lambda\|\mathcal{P}_\Omega(S)\|_1 + \frac{\mu}{2}\|LR^TO\|_F^2, \\ \text{s.t. } &LR^T + S = \mathcal{P}_\Omega(D), UU^T = VV^T = I_r, \end{aligned} \tag{4.21}$$

where $O \in \mathbb{R}^{n\times n}$ is a symmetrical matrix,

$$O = \begin{bmatrix} -1 & 1 & & & & & \\ 1 & -2 & 1 & & & & \\ & \ddots & \ddots & \ddots & & & \\ & & & 1 & -2 & 1 & \\ & & & & 1 & -1 \end{bmatrix}. \tag{4.22}$$

Note that Eq. (4.21) is a more general model, e.g. case $r = 0$ corresponds to the model variant studied in [Feng et al. 2014b], and if $\beta = 0$ and Ω is the whole space, it reduces to RPCA problem [Lin et al. 2010].

4.2.2 Solution to the optimization problem

In this section, it shows how to use the augmented Lagrange multiplier (ALM) method to efficiently solve problem (4.21). By introducing a slack

variable $M = LR^T$, the author equivalently reformulates Eq 4.21 as

$$\begin{aligned} \min_{L,R,S,M,U,V} \ & \frac{1}{2}(\|L\|_F^2 + \|R\|_F^2) - tr(ULR^TV^T) \\ & + \lambda\|\mathcal{P}_\Omega(S)\|_1 + \frac{\mu}{2}\|MO\|_F^2, \\ \text{s.t. } & M + S = \mathcal{P}_\Omega(D), M = LR^T, UU^T = VV^T = I_r. \end{aligned} \tag{4.23}$$

It is noteworthy that the introduction of slack matrix M decouples L, R and O, and also enables us to give simple closed-form solutions at the following alternative iterations. For problem (4.23), the author defines its partial augmented Lagrangian function $\mathcal{L} =: \mathcal{L}(L, R, S, M, U, V, Y_1, Y_2, \eta)$ as

$$\begin{aligned} \mathcal{L} = & \frac{1}{2}(\|L\|_F^2 + \|R\|_F^2) - tr(ULR^TV^T) + \lambda\|\mathcal{P}_\Omega(S)\|_1 \\ & + \frac{\mu}{2}\|MO\|_F^2 + \frac{\eta}{2}(\|M - LR^T\|_F^2 + \|\mathcal{P}_\Omega(D) - LR^T - S\|_F^2) \\ & + \langle Y_1, M - LR^T\rangle + \langle Y_2, \mathcal{P}_\Omega(D) - LR^T - S\rangle \\ & \text{s.t. } UU^T = VV^T = I_r, \end{aligned}$$

where $Y_1, Y_2 > 0$ are Lagrange multiplier matrices, $\eta > 0$ is used to penalize the linear equality constraints. Then, an alternating direction idea can be employed to consecutively update each variable separately while fixing the other variables in the order of (L, R, S, M, U, V) under the spirit of Gauss-Seidel iteration.

First, by simple computation, the L and R are updated as

$$\begin{cases} L^{k+1} = T_1^{k+1}R^k(T_2^{k+1})^{-1}, \\ R^{k+1} = (T_1^{k+1})^T L^{k+1}(T_3^{k+1})^{-1}, \\ S^{k+1} = \mathcal{P}_{\Omega^c}(T_s^{k+1}) + \mathcal{P}_\Omega\left(\mathcal{T}_{\lambda/\eta^k}(T_s^{k+1})\right), \end{cases} \tag{4.24}$$

where $T_1^{k+1} = (U^k)^TV^k + (Y_1^k + Y_2^k) + \eta^k(M^k + \mathcal{P}_\Omega(D) - S^k)$, $T_2^{k+1} = I_d + 2\eta^k(R^k)^TR^k$, $T_3^{k+1} = I_d + 2\eta^k(L^{k+1})^TL^{k+1}$, $T_s^{k+1} = \mathcal{P}_\Omega(D) - L^{k+1}(R^{k+1})^T + Y_2^k/\eta^k$ and $\mathcal{T}_\tau(\cdot)$ denotes the element-wise soft-thresholding operator [Lin et al. 2010; Daubechies et al. 2004].

Next, M can be roughly updated by $M^{k+1} = T_m^{k+1}(\eta^k I_n + \mu O^2)^{-1}$,

where $T_m^{k+1} = \eta^k L^{k+1}(R^{k+1})^T - Y_1^k$. However, for a large n, the above inversion calculation is much costly. Fortunately, the eigen decomposition of O defined in Eq. (4.22) has good properties. Namely, for $O = U\Lambda U^T$, it leads to

$$\Lambda = \text{diag}(\xi_1, \cdots, \xi_n),\ \xi_i = -2 + 2\cos((i-1)\pi/n),$$

and U^T and U composed of eigenvectors are actually n-by-n type-2 discrete cosine transform (DCT) and inverse DCT (IDCT) matrices, respectively [Garcia 2010]. As a consequence, the author can get a much more economical updating rule for M :

$$M^{k+1} = [(M^{k+1})^T]^T = \left[\text{IDCT}\left(\Gamma\ \text{DCT}\left((T_m^{k+1})^T\right)\right)\right]^T, \tag{4.25}$$

where Γ is a diagonal matrix with the diagonal elements

$$\Gamma_{ii} = [Y_1^k + \mu\xi_i^2]^{-1},\ i = 1, 2, \cdots, n.$$

Finally, U and V are updated based on the lemma similar to [Hong et al. 2016]. Besides, A different proof from [Hong et al. 2016] is provided in Lemma 1 at Appendix. Therefore, U and V are updated by

$$\begin{aligned} U^{k+1} &= \underset{U\in\mathbb{R}^{r\times m}}{\text{argmax}}\ tr(UL^{k+1}(R^{k+1})^T(V^k)^T) \text{ s.t. } UU^T = I_r \\ &= Q_u^{k+1}(P_u^{k+1})^T, \\ V^{k+1} &= \underset{V\in\mathbb{R}^{r\times n}}{\text{argmax}}\ tr(VR^{k+1}(L^{k+1})^T(U^{k+1})^T) \text{ s.t. } VV^T = I_r \\ &= Q_v^{k+1}(P_v^{k+1})^T, \end{aligned} \tag{4.26}$$

where P_u^{k+1}, Q_u^{k+1} and P_v^{k+1}, Q_v^{k+1} are given by the economy-size SVDs of $L^{k+1}(R^{k+1})^T(V^k)^T$ and $R^{k+1}(L^{k+1})^T(U^{k+1})^T$, respectively.

The whole algorithm is summarized in Algorithm 3. Compared to the previous TrNN based works [Oh et al. 2016] (that is based on a partial singular value thresholding (PSVT)) and [Hu et al. 2013b] (that is a two-loop iterative scheme), our method is a one-loop iterative scheme based on matrix factorization and each step has closed-form solutions. Due to including matrix multiplication, DCT and IDCT, and SVD

Algorithm 3 Mocap data completion via TrNN

Input: D, Ω, O, d, r, $\lambda, \mu, \max_\eta$.

1: **Initialize:** $R^1 = \text{randn}(m, n)$, $R^1 = \frac{R^1}{\|R^1\|_F}$, $S^1 = M^1 = 0$, $\rho > 0, \max_\eta = 10^6$, $Y_1^1 = Y_2^1 = \frac{D}{\max(\|D\|_2, \|D\|_\infty/\lambda)}$, $\eta^1 > 0$, and $k = 1$.
2: **while** not converge **do**
3: Update L^{k+1}, R^{k+1} and S^{k+1} using (4.24);
4: Update M^{k+1} using (4.25);
5: Update U^{k+1} and V^{k+1} using (4.26);
6: Update $Y_1^{k+1} = Y_1^k + \eta^k(M^{k+1} - L^{k+1}(R^{k+1})^T)$,
$Y_2^{k+1} = Y_2^k + \eta^k(\mathcal{P}_\Omega(D) - L^{k+1}(R^{k+1})^T - S^{k+1})$;
7: $\eta^{k+1} = \min\{\rho\eta^k, \max_\eta\}$;
8: $k = k + 1$;
9: **end while**

Output: $\tilde{X} = L^k(R^k)^T$.

for matrices of small size, the proposed algorithm has complexity of $\mathcal{O}(mnd + n\log(n))$.

4.2.3 Experimental results

In this section, the designed approach is applied to the task of corrupted data completion on both synthetic data and real motion capture data. All the experiments were implemented and timed on a PC with an Intel Core i5 CPU at 2.4GHz and with 4GB of memory, running Windows 7 and Matlab version 7.10. During the experiments, the parameters are set as d as $d = \min(\text{round}(1.2 \times \text{rank}(X)), m, n)$ and employ the convergence rate at the k-th iteration defined as

$$\vartheta = \max\{\frac{\|\mathcal{P}_\Omega(D) - L^k(R^k)^T - E^k\|_F}{\|\mathcal{P}_\Omega(D)\|_F}, \frac{\|L^k(R^k)^T - L^{k-1}(R^{k-1})^T\|_F}{\|L^k(R^k)^T\|_F}\}$$

, and run the iterations until $\vartheta < \varepsilon$.

Matrix completion on synthetic data

Let the raw data matrix be $D = X^* + S^*$, where X^* and S^* are respectively the low-rank and sparse components to be recovered. First of all, the author synthesizes the low-rank part X^* as the product of a $m \times r_0$ matrix and a $r_0 \times n$ matrix, whose entries are generated independently

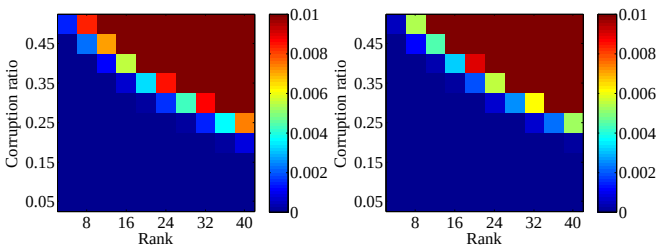
from standard Normal distribution. Hence, the rank of X is r_0. Next, for generating S^*, the author selects $m \times n \times cr$ entries randomly from X^*, whose values are corrupted by random noises from $U[-10, 10]$, where cr denotes the corruption ratio.

It fixes $m = 1000$ and $n = 200$, and sets $\lambda = \frac{1}{\sqrt{\max(m,n)}}$, $\mu = 0$, $r = r_0$, $\rho = 1.5$, $\eta^1 = 1.25/\|D\|_2$, $\varepsilon = 10^{-7}$ and Ω as the whole space. The propsed approach (TrNN) is compared with PSVT Oh et al. [2016] and IALM [Lin et al. 2010] by evaluating the normalized reconstruction error (NRE) $\frac{\|\tilde{X}-X^*\|_F}{\|X^*\|_F}$ and execution time over various settings of matrix rank and corruption ratio. The experiments are repeated by 30 times. PSVT and IALM are two representative approaches, because one is based on the truncated nuclear norm while the other on the traditional nuclear norm. Fig. 4.6 reports the NRE, and Fig. 4.7 reports the execution time. It's shown that TrNN and PSVT producing comparable results outperform IALM as the matrix rank r_0 and corruption ratio cr increase, which is due to that they take a *prior* rank information into account. In Fig. 4.7, it can find that TrNN performs more efficiently than PSVT and IALM. The reason is that PSVT needs compute full SVD at each iteration that is very costly for matrices of large size, while IALM converges incredibly slowly after ϑ reaches certain threshold.

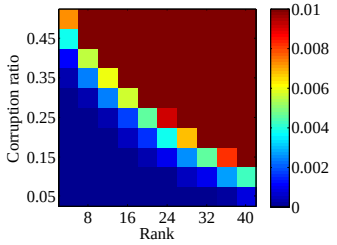
Mocap data completion

In order to test the capacity of the proposed approach in completing missing data, test motion sequences are selected from CMU MoCap database[1]. The CMU MoCap totally includes 2605 trials which are classified into 6 categories and 23 subcategories. Each motion can be represented by an $m \times n$ matrix X, where $m \equiv 93$ and n denotes the frame number (generally $m \ll n$). In Fig. 4.5, the author has verified the correctness of low-rank properties of all motion matrices. the completion results of six motion sequences (i.e. dance, walk, gymnastics,

[1] http://mocap.cs.cmu.edu/



(a) TrNN-Ours (b) PSVT Oh et al. [2016]



(c) IALM Lin et al. [2010]

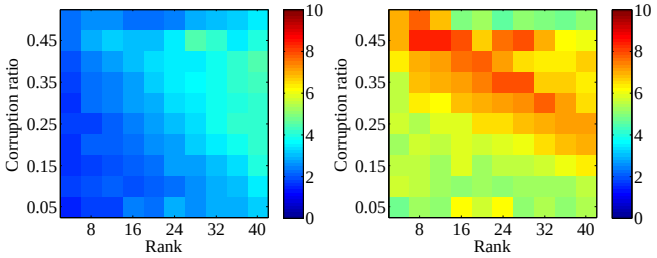
Figure 4.6: *Comparison of NRE for synthetic data by varying rank r_0 and support ratio of the sparse component cr. (a) the proposed approach (TrNN), (b) PSVT Oh et al. [2016], and (c) IALM Lin et al. [2010]. The color magnitude represents NRE.*

jump, score and boxing[2]) are presented in the following experiments.

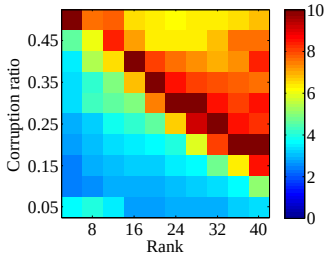
Like [Feng et al. 2014b], the author generates two kinds of incomplete mocap data: randomly missing data and continuously missing data. The first one is obtained by randomly removing mr markers from each frame, while the second one is continuously removing ml frames for 10 randomly selected missing markers in each frame, where mr and ml denote the randomly missing ratio and continuously missing length, respectively. The experiments are performed by fixing $\lambda = \frac{100}{\sqrt{\max(m,n)}}, \mu = 100, \rho = 1.4, \eta^1 = 10^{-5}$ and $\varepsilon = 10^{-4}$, and repeated by 20 times.

Recalling the definition of truncated nuclear norm in (4.20), it needs to estimate the rank rank(X) of the incomplete motion data at first. To this end, an initial guess for missing data is given by adopting the linear interpolation scheme along the temporal direction, and then detect the largest jump between adjacent singular values from SVD. Specifically, the estimated rank is set as the largest index where the jump is beyond a specified threshold, namely, $\text{rank}(X) \approx \text{argmax}\{i : |\sigma_i - \sigma_{i+1}| \geq 0.1\}$.

[2]The indices of the selected motions are 05_13, 12_02, 49_02, 13_13, 10_01 and 13_17, which consist of multiple types of action.



(a) TrNN-Ours (b) PSVT Oh et al. [2016]

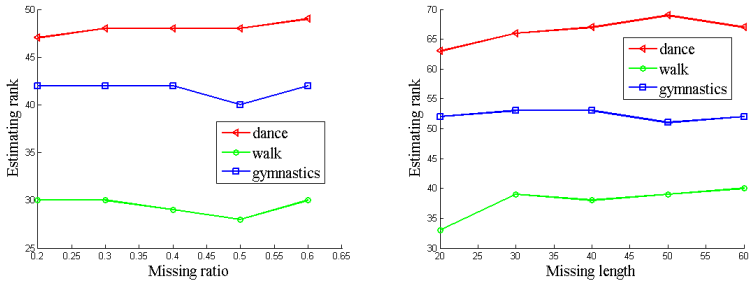


(c) IALM Lin et al. [2010]

Figure 4.7: *Comparison of execution time for synthetic data by varying rank r_0 and support ratio of the sparse component cr. (a) the proposed approach (TrNN), (b) PSVT Oh et al. [2016], and (c) IALM Lin et al. [2010]. The color magnitude represents execution time in seconds.*

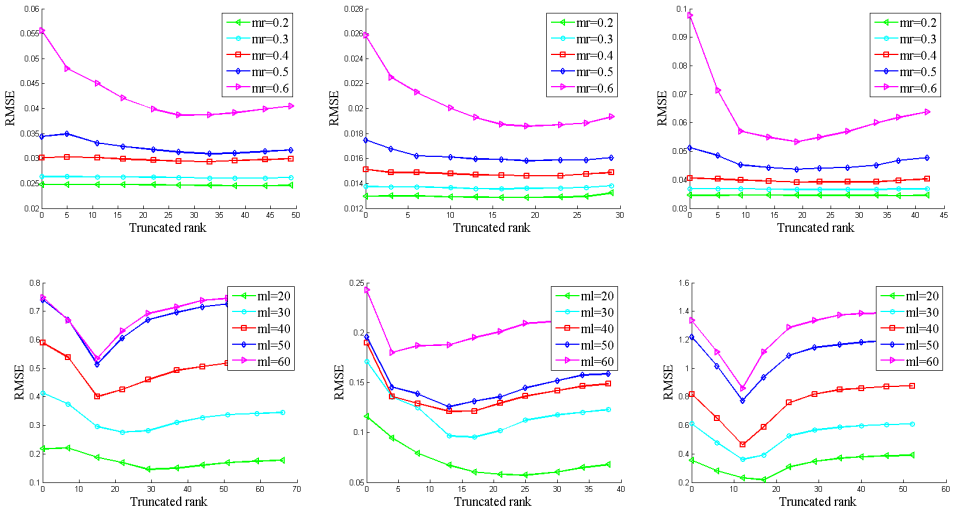
For both of the missing modes,the robustness of rank estimation is verified with respect to the missing ratio mr and missing length ml in Fig. 4.8. It observes that the estimated ranks keep much stable as mr and ml increase. The result is reasonable since human motion is highly articulated so that most of the missing information can be revived through interpolation from neighboring markers.

Next, the effect of truncated rank r to the motion completion results is illustrated, where the metric of Root Mean Squared Error (RMSE) is adopted, which is defined by RMSE $= \frac{\|(X-\tilde{X})|_{\Omega^c}\|_F}{\sqrt{|\Omega^c|}}$. As shown in Fig. 4.9, the truncated nuclear norm indeed improves the completion results a lot except for the randomly missing cases with mr below 0.4. This is due to the results of these cases are already good enough (because RMSE ≤ 0.03), and besides a small amount of randomly missing markers don't make the original motion structures badly damaged, so they can be recovered well from their (dense) neighboring markers. Most significantly, the author has observed that regardless of what values mr and ml take, the RMSE of all the cases approximately attain its minims when the truncated rank is $r = 15$. Thus, it always sets $r = 15$ in the



(a) Randomly missing (b) Continuously missing

Figure 4.8: *Robustness of rank estimation with respect to the randomly missing ratio mr and continuously missing length ml. (a) Estimated ranks for* $mr = [0.2 : 0.1 : 0.6]$*, (b) Estimated ranks for* $ml = [20 : 10 : 60]$*.*

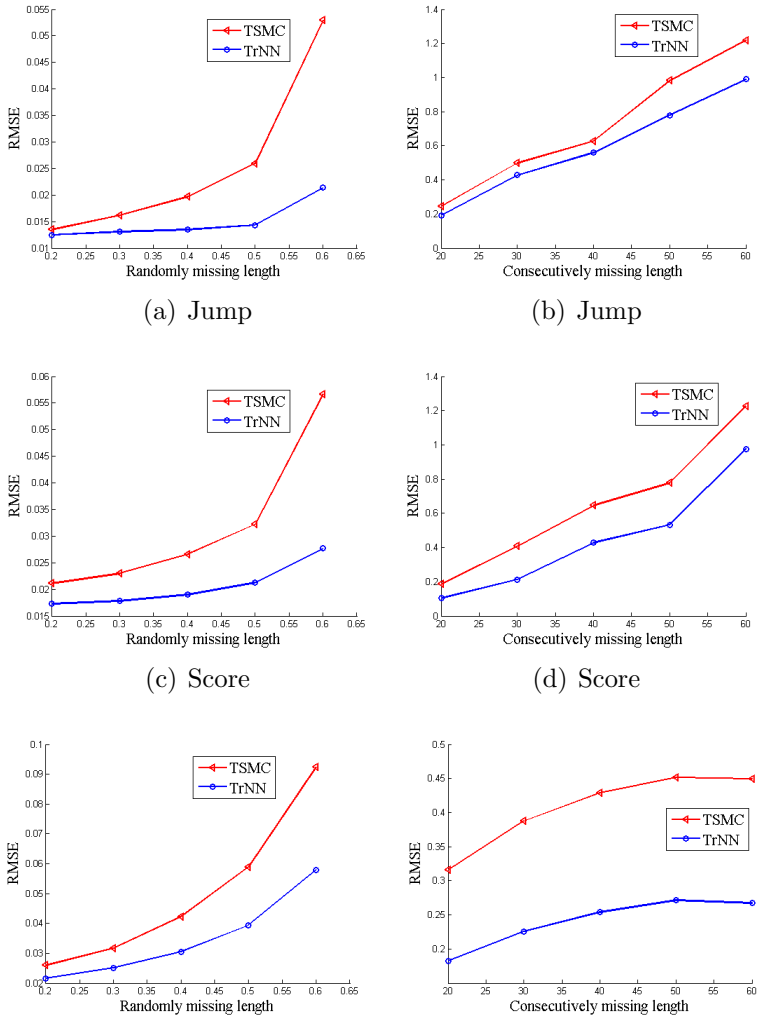


(a) Dance (b) Walk (c) Gymnastics

Figure 4.9: *Illustration of the effect of truncated rank to the completing results via RMSE. The* ***upper*** *row shows the RMSE for the randomly missing case with* $mr = [0.2 : 0.1 : 0.6]$*, while the* ***lower*** *row shows the RMSE for the continuously missing case with* $ml = [20 : 10 : 60]$*.*

following experiments.

Finally, the performance of proposed approach (also called TrNN as in Section 4.2.3) is evaluated by comparing with Feng et al.'s approach (TSMC) [Feng et al. 2014b]. Since [Feng et al. 2014b] have demonstrated TSMC's better performance than many other methods such as linear/spline interpolation, Dynammo [Li et al. 2009] and SVT [Lai et al. 2011], only comparison between TrNN and TSMC are shown. In Table 4.1, it reports the time efficiency completion results of 12 motions. The results have shown that TrNN outperforms TSMC and its advantage becomes even more obvious especially for the long term actions. An



(a) Jump (b) Jump

(c) Score (d) Score

(e) Boxing (f) Boxing

Figure 4.10: *Comparison of completion results between the proposed method (TrNN) and TSMC [Feng et al. 2014b]. The* ***left*** *row shows the RMSE for the randomly missing case with* $mr = [0.2 : 0.1 : 0.6]$*, while the* ***right*** *row shows the RMSE for the continuously missing case with* $ml = [20 : 10 : 60]$*.*

overall comparison of SVT [Lai et al. 2011],dyanmmo [Li et al. 2009], PSC [Wang et al. 2016b] (The method proposed in previous chapter), TSMC[Feng et al. 2014b] and the proposed TrNN based low rank matrix factorization (LRMF) method is shown in Fig 4.11, where the proposed method outperforms the competitors.

Action	NO. of Frames	TSMC(s)	TrNN(s)
Running	152	1.61	2.69
Walk	343	2.99	4.35
Basketball	4905	286.25	102.42
Boxing	4840	267.29	73.75
Dance	1095	21.23	14.84
Gymnastics	575	7.63	8.17
Jump	439	4.07	6.13
Punch	2251	93.33	33.36
Score	801	8.95	10.19
Taichi	17799	2094.8	564.85
Varied	2085	111.69	36.30
Acrobatics	2422	121.93	34.09

Table 4.1: *Time efficiency comparison between TSMC and proposed TrNN.*

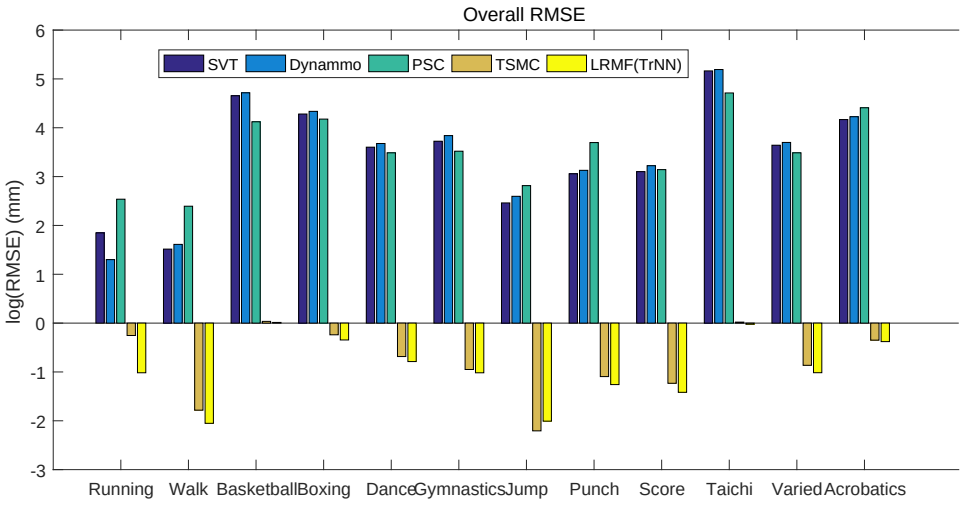


Figure 4.11: *Comparison of the refinement performance on 12 actions: SVT, Dynammo, PSC, TSMC, LRMF(TrNN)*

4.3 Summary

In this chapter, the problem of MoCap data completion based on the matrix completion which takes into account the prior information of the rank function has been studied. Firstly, a bilinear matrix factorization method is designed to approximate the low-rank property of MoCap data. Then, an improved approach which is based on truncated norm based approach is designed. In addition, an efficient optimization algorithm is devised by using the augmented Lagrange multiplier method. Extensive experiments show that the proposed method outperforms the state-of-the-art methods on both matrix completion and MoCap data completion. In the future, it plans to revisit mocap data completion by replacing nuclear norm with a rank heuristic proposed in Shu et al. [2014], which is free of rank estimation and can be further accelerated.

Chapter 5

Motion Retrieval: Adaptive Multi-View Feature Selection

Due to the high complexity and diversity of human motion, human motion retrieval is a very challenging task in computer animation and multimedia analysis communities. Similar to other multimedia retrieval tasks like image retrieval and document retrieval, feature representation is the corner-stone of human motion retrieval algorithm and system. A compact and discriminative motion feature representation can not only significantly improves the performance of the retrieval algorithm but also dramatically reduces the consuming time of the algorithm. As is known to all, there is a big gap between the low-level visual feature and the high-level semantic meaning, so single low-level visual feature is usually unable to fully characterize all aspects of the motion data. In other words, it would be beneficial to fuse multiple visual features for motion data representation.

To this end, an Adaptive Multi-View Feature Selection (AMFS) method is designed and shown in this chapter, which can automatically assign multi-view features with adaptive feature weights and select out a compact and discriminative feature subset from the original high-dimensional multi-view features. With the selected low-dimensional feature representation, it not only improves the motion data retrieval accuracy but also speeds up the whole motion data retrieval processing. The AMFS

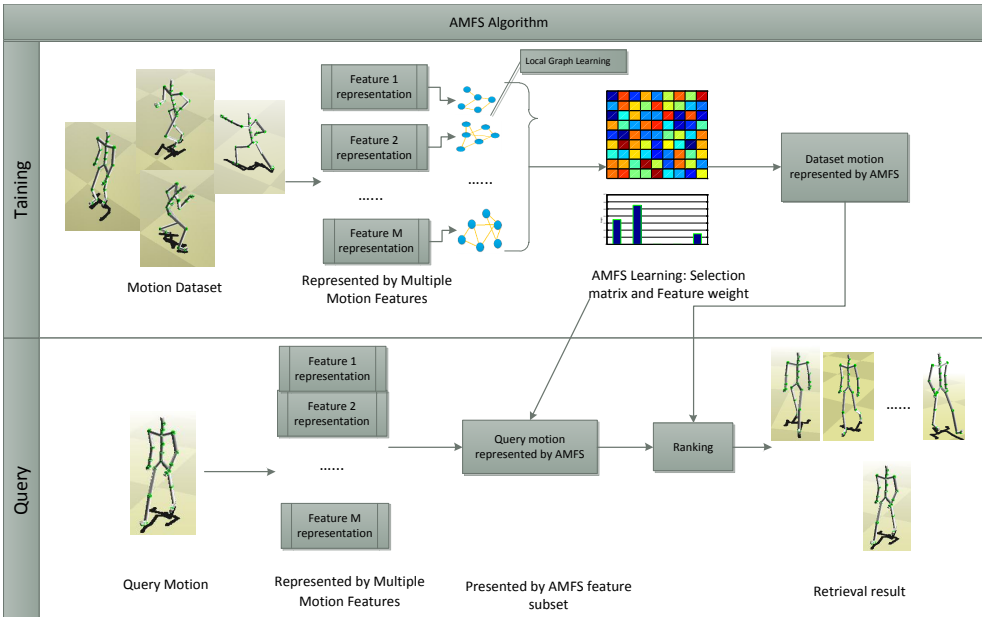


Figure 5.1: *The flowchart of our proposed AMFS motion retrieval framework. It comprises of two parts: 1) AMFS learning part, which learns a compact and discriminative feature representation from original high dimensional multi-view features. Therefore, the motion data in database can be represented by the learned compact features; 2) motion retrieval part, where the query motion is firstly represented by the original multiple high dimensional features, and then is translated to the same compact feature space with the learned feature selection matrix. After that, many existing ranking algorithms can be directly applied to the task of motion retrieval in our framework.*

method presented in this chapter has appeared in [Wang et al. 2014c]

5.1 AMFS Algorithm

Suppose a human motion dataset consists of N motion clips, $\{x_1, x_2, \cdots, x_N\}$ is used to denota M types of visual features. For any motion clips, $x_i^{(v)} \in \mathbb{R}^{d_v \times 1}$ is the v-th visual feature representation of the motion clip x_i, where d_v is the dimension of the v-th visual feature. Thus, the v-th feature data matrix of this dataset can be denoted as $X^{(v)} = [x_1^{(v)}, x_2^{(v)}, ..., x_N^{(v)}] \in \mathbb{R}^{d_v \times N}$. The multi-view feature data matrix of this dataset is denoted as $X = [X^{(1)}; X^{(2)}; ...; X^{(M)}] \in \mathbb{R}^{D \times N}$, where $D = \sum_{v=1}^{M} d_v$. Since different visual feature describes different aspects about motion data, and they have dissimilar discriminative power with respect to one particular class

of human motion. Thus, it would be beneficial to combine multiple visual features together for motion data representation. If all of these multiple features are just concatenates together, it does not only lack of physical meaning but also fail to exploit the complementary information between different visual feature. More worse, it would bring in some redundant information in feature representation. To overcome these problems, a compact yet discriminative feature subset $Y \in \mathbb{R}^{d\times N}, d < D$ will be selected out from the original multiple features X via feature selection method. That is to say, it hopes to get:

$$Y = W^T X,\ Y \in \mathbb{R}^{d\times N},\ W \in \{0,1\}^{D\times d} \tag{5.1}$$

where W is a feature selection matrix that picks out the most representative feature elements from X.

Since the local geometric structure information is important in real-world applications [Roweis and Saul 2000], it also hopes to preserve the local information of motion data during feature selection in this framework. To this end, a local regressive model is designed to character the local information of motion data [Yang et al. 2012b]. It assumes that for each data point, the selected low-dimensional feature can be expressed as a linear function mapping from the original visual feature.

Taken a data point $x_i^{(v)}$ in the v-th feature view $X^{(v)}$, where $X^{(v)} = [x_1^{(v)}, x_2^{(v)}, ..., x_N^{(v)}] \in \mathbb{R}^{d_v\times N}$. Its k nearest neighbors $N_k(x_i^{(v)}) = \{x_i^{(v)}, x_{i1}^{(v)}, x_{i2}^{(v)} ..., x_{i(k-1)}^{(v)}\}$ will be founded, wherein $x_{ip}^{(v)}, p = 1, 2, ..., k-1$ is the k-nearest neighbors of $x_i^{(v)}$ in $X^{(v)}$. If $y_j \in Y_{N_i}$ is the corresponding feature selection result for $x_j \in N_k(x_i^{(v)})$, the local regressive model can be formulated as below:

$$y_i^{(v)} = f_i(x_i^{(v)}) = (w_i^{(v)})^T x_i^{(v)} + b_i^{(v)}, i = 1, 2, ..., N \tag{5.2}$$

where $w_i^{(v)} \in \mathbb{R}^{d_v\times d}$, $b_i^{(v)} \in \mathbb{R}^d$.

In real-world applications, the local geometric structure of visual data is approximately linear [Roweis and Saul 2000]. Besides, it is computational efficient for practical applications using linear method. Therefore,

a linear method is chosen in this work rather than other complex non-linear models [Yang et al. 2012b]. That is to say, a linear regression model $f_i(x_i^{(v)}) = (w_i^{(v)})^T x_j^{(v)} + b_i^{(v)}$ could be used for each data point $x_j^{(v)} \in N_k(x_i^{(v)})$ for mapping from its original visual feature to the corresponding low-dimensional feature. Then, the local prediction error can be represented by $\|f_i(x_j^{(v)}) - y_j\|^2$, that is

$$\|(w_i^{(v)})^T x_j^{(v)} + b_i^{(v)} - y_j^{(v)}\|^2 \tag{5.3}$$

And, the local model error can be computed by summing the local predict error of each data point in $N_k(x_i^{(v)})$. Thus, the author can minimize the local model error to get a best mapping function, that is

$$\min \sum_{x_j^{(v)} \in N_k(x_i^{(v)})} \|(w_i^{(v)})^T x_j^{(v)} + b_i^{(v)} - y_j\|^2 \tag{5.4}$$

In order to avoid over-fitting, a regularization term is imposed, so the local model error is formulated as:

$$\sum_{x_j^{(v)} \in N_k(x_i^{(v)})} \|(w_i^{(v)})^T x_j^{(v)} + b_i^{(v)} - y_j\|^2 + \mu \|w_i^{(v)}\|_F^2 \tag{5.5}$$

Now, considering the v-th feature view rather than a single data point, let's denote $X_i(v) = [x_i^{(v)}, x_{i1}^{(v)} ..., x_{i(k-1)}^{(v)}] \in \mathbb{R}^{d_v \times k}$, $Y_{N_i} = [y_i, y_{i1}, ..., y_{i(k-1)}] \in \mathbb{R}^{d \times k}$. The local model error can be written as:

$$\|(X_i^{(v)})^T w_i^{(v)} + 1_k (b_i^{(v)})^T - Y_{N_i}^T\|_F^2 + \mu \|w_i^{(v)}\|_F^2 \tag{5.6}$$

Then the objective function of minimizing the local model error can be formulated as:

$$\min_{w_i^{(v)}|_{i=1}^N, b_i^{(v)}|_{i=1}^N, Y_{N_i}^{(v)}|_{i=1}^N} \sum_{i=1}^{N} \|(X_i^{(v)})^T w_i^{(v)} + 1_k (b_i^{(v)})^T - Y_{N_i}^T\|_F^2 + \mu \|w_i^{(v)}\|_F^2 \tag{5.7}$$

By setting the derivatives of Eq 5.7 to the zero w.r.t $b_i^{(v)}$, it leads to

$$\begin{aligned} &(w_i^{(v)})^T X_i^{(v)} 1_k + k b_i^{(v)} - Y_{N_i} 1_k = 0 \\ &\Rightarrow b_i^{(v)} = \tfrac{1}{k}(Y_{N_i} 1_k - (w_i^{(v)})^T X_i^{(v)} 1_k) \end{aligned} \tag{5.8}$$

By setting the derivatives of Eq 5.7 to the zero $w_i^{(v)}$ and substituting (9), the author can have

$$\begin{aligned} &X_i^{(v)}(X_i^{(v)})^T w_i^{(v)} + X_i^{(v)}(1_k(b_i^{(v)})^T - Y_N^T) + \mu w_i^{(v)} = 0 \\ &\Rightarrow w_i^{(v)} = [X_i^{(v)} H (X_i^{(v)})^T + \mu I]^{-1} X_i^{(v)} H Y_{N_i}^T \end{aligned} \tag{5.9}$$

where $H = I - \frac{1}{k} 1_k 1_k^T$ is the centralized matrix. Note that $H = HH^T = H^T$. Substituting Eq 5.8 and Eq 5.9 to Eq 5.7, then the author can get:

$$\begin{aligned} &(X_i^{(v)})^T w_i^{(v)} + 1_k (b_i^{(v)})^T - Y_{N_i}^T \\ &= H(X_i^{(v)})^T [X_i^{(v)} H (X_i^{(v)})^T + \mu \cdot I]^{-1} X_i^{(v)} H Y_{N_i}^T - H Y_{N_i}^T \end{aligned} \tag{5.10}$$

Thus, the original objective function Eq 5.7 can be formulated as:

$$\begin{aligned} \min_{Y_{N_i}^{(v)}|_{i=1}^N} \quad & \textstyle\sum_{i=1}^N \|H((X_i^{(v)}))^T [X_i^{(v)} H (X_i^{(v)})^T + \mu \cdot I]^{-1} X_i^{(v)} H Y_{N_i}^T - H Y_{N_i}^T\|_F^2 + \\ & \mu \|[X_i^{(v)} H (X_i^{(v)})^T + \mu \cdot I]^{-1} X_i^{(v)} H Y_{N_i}^T\|_F^2 \end{aligned} \tag{5.11}$$

Let us denote

$$\begin{aligned} \mathcal{J} = \quad & \textstyle\sum_i^N \|H(X_i^{(v)})^T [X_i^{(v)} H (X_i^{(v)})^T + \mu \cdot I]^{-1} X_i^{(v)} H Y_{N_i}^T - H Y_{N_i}^T\|_F^2 \\ & + \mu \|[[X_i^{(v)} H ([X_i^{(v)})^T + \mu \cdot I]^{-1} X_i^{(v)} H Y_{N_i}^T\|_F^2 \\ & = tr\{Y_{N_i}[H(X_i^{(v)})^T (X_i^{(v)} H (X_i^{(v)})^T + \mu \cdot I)^{-1} X_i^{(v)} H - H]^2 Y_{N_i}^T\} \\ & + \mu tr\{Y_{N_i}[H(X_i^{(v)})^T (X_i^{(v)} H (X_i^{(v)})^T + \mu \cdot I^{-2} X_i^{(v)}] H] Y_{N_i}^T\} \\ & = tr\{Y_{N_i}[H - H(X_i^{(v)})^T (X_i^{(v)} H (X_i^{(v)})^T + \mu \cdot I)^{-1} X_i^{(v)} H] Y_{N_i}^T\} \end{aligned} \tag{5.12}$$

Therefore, the objective function 5.11 is equivalent to

$$\min_{Y_{N_i}|_{i=1}^N} \sum_{i=1}^N \mathcal{J} \tag{5.13}$$

and can be represented as a trace ratio format:

$$\min_{Y_{N_i}|_{i=1}^{N}} \sum_{i=1}^{N} tr\{Y_{N_i} L_i^{(v)} Y_{N_i}^T\} \tag{5.14}$$

where $L_i^{(v)} = H - H(X_i^{(v)})^T (X_i^{(v)} H (X_i^{(v)})^T + \mu \cdot I)^{-1} X_i^{(v)} H$, $L_i^{(v)} \in \mathbb{R}^{k \times k}$. Let us denote a selection matrix $S_i^{(v)} \in \mathbb{R}^{N \times k}$ that $(S_i^{(v)})_{pq} = 1$ when data point $x_p^{(v)}$ is the qth neighbor of $x_i^{(v)}$, otherwise $(S_i^{(v)})_{pq} = 0$. Hence, the local feature selection results of $x_i^{(v)}$ and its neighbors can be represented as $Y_{N_i} = Y S_i^{(v)}$. Then, the author can get the objective function as

$$\min_{Y} \sum_{i=1}^{N} tr\{Y S_i^{(v)} L_i^{(v)} (Y S_i^{(v)})^T\} \tag{5.15}$$

Now, let's focus on the whole feature rather than looking at the single data point and its neighbors. The local graph for the vth feature then can be built as:

$$L^{(v)} = [S_1^{(v)}, S_2^{(v)} ..., S_N^{(v)}] diag(L_1^{(v)}, L_2^{(v)} ..., L_N^{(v)}) [S_1^{(v)}, S_2^{(v)} ..., S_N^{(v)}]^T \tag{5.16}$$

Thus, the objective function is formulated for the v-th feature as

$$\min_{Y} tr\{Y L^{(v)} Y^T\} \tag{5.17}$$

Here, let's move on to consider all the features rather than a single feature. The equation 5.1 in beginning can be represented as a combination

of selection from all the features:

$$\begin{aligned} Y &= \begin{bmatrix} W_{d_1\times d} \\ W_{d_2\times d} \\ \vdots \\ W_{d_M\times d} \end{bmatrix}^T \begin{bmatrix} X^{(1)} \\ X^{(2)} \\ \vdots \\ X^{(M)} \end{bmatrix} = [W_{d_1\times d}^T, W_{d_2\times d}^T, ..., W_{d_M\times d}^T] \begin{bmatrix} X^{(1)} \\ X^{(2)} \\ \vdots \\ X^{(M)} \end{bmatrix} \\ &= \sum_{v=1}^{M} W_{d_v\times d}^T \cdot X^{(v)} = \sum_{v=1}^{M} Y^v \end{aligned} \tag{5.18}$$

Since each feature characterizes different aspects of the motion and hold different intrinsic discriminative power. In order to keep the locality from each feature view and take the synergy effect of all features, a non-negative weight vector α is introduced to unite all the features together.

$$\begin{aligned} &\min_{Y} \textstyle\sum_{v=1}^{M} \alpha_v tr(Y L^{(v)} Y^T) \\ &= tr(Y \textstyle\sum_{v=1}^{M} \alpha_v L^{(v)} Y^T) \end{aligned} \tag{5.19}$$

where $\sum_{v=1}^{M} \alpha_v = 1, \alpha_v \geq 0$. The value range of α is between 0 to 1 since prior research [Yang et al. 2012b] has shown that negative of α is lack of physical meaning and the non-negative constraint would make result much closer to the idea solution. One thing need to be pointed out is that, the solution to α in last equation is $\alpha_v = 1$ corresponding to minimize $tr(Y L^{(v)} Y^T)$ over different feature views, and $\alpha_v = 0$ otherwise. This solution means that only one best feature is selected, which does not meet our objective. Thus a trick utilized in [Wang et al. 2007b; Xia et al. 2010; Feng et al. 2013b] is adopted for this problem, α_i is set as α_i^r with $r > 1$. Therefore, each feature would has a particular contribution for final Y. Hence, the improved objective function is defined as

$$\begin{aligned} \min_{W,\alpha} \quad & tr(W^T X \textstyle\sum_{v=1}^{M} \alpha_v^r L^{(v)} X^T W) \\ s.t. \quad & \textstyle\sum_{v=1}^{M} \alpha_v = 1, \alpha_v \geq 0,\ W \in \{0,1\}^{D\times d} \end{aligned} \tag{5.20}$$

All of above shows how the author is going to preserve the local information from each feature. Moreover, the author is also going to keeps

the global information at same time. Inspired by PCA, the author can get

$$\min_{W} tr(W^T X H_X X^T W) \tag{5.21}$$

Therefore, the final objective function is written in a race ratio minimization form, which is shown below

$$\begin{array}{ll} \min\limits_{W,\alpha} & \frac{tr(W^T X \sum_{v=1}^{M} \alpha_v^r L^{(v)} X^T W)}{tr(W^T X H_X X^T W)} \\ s.t. & \sum_{v=1}^{M} \alpha_v = 1, \alpha_v \geq 0,\ W \in \{0,1\}^{D \times d} \end{array} \tag{5.22}$$

5.2 Optimization Method

Inspired by [Jia et al. 2009], the author develops an iterative approach to solve the optimization of our trace ratio represented objective function. It is clearly a nonlinearly constrained optimization problem. The feature weight vector α is firstly initialized as $\alpha_v = \frac{1}{M}$, W is set with a random selected matrix, then W and α will be iteratively updated individually, while the other variable holding constant. Detail of the procedure is shown below.

Algorithm 4 Optimize W with fixed α

1: **Initialize feature weight** α:
   $\alpha_v = \frac{1}{M}$;
2: **Local and global structure learning:**
   set $A = X(\sum_{v=1}^{M} \alpha_v^r L^{(v)})X^T, B = X H_X X^T$;
3: **Solving trace Ratio**
   $\lambda = \frac{tr(W^T A W)}{tr(W^T B W)}$;
4: **Calculate score for each feature element**
   $sc_i = {w_i}^T (A - \lambda B) w_i,\ w_i = [0_{i-1}, 1, 0_{D-i}],\ i = 1, \ldots, D$;
5: **Select relevant feature elements to Update** W
   Sort components according to sc_i in descent order, update W with the first d components
6: **Repeat until convergence, output** W

while W is fixed, the objective function is only related to α.

$$\begin{array}{ll} \min\limits_{\alpha} & tr(W^T X \sum_{v=1}^{M} \alpha_v^r L^{(v)} X^T W) \\ s.t. & \sum_{v=1}^{M} \alpha_v = 1, \alpha_v \geq 0 \end{array} \tag{5.23}$$

For the objective function shown above, the author applies the Langrange multiplier to solve it. Following the similar procedure in [Wang et al. 2007b], finally the author can get:

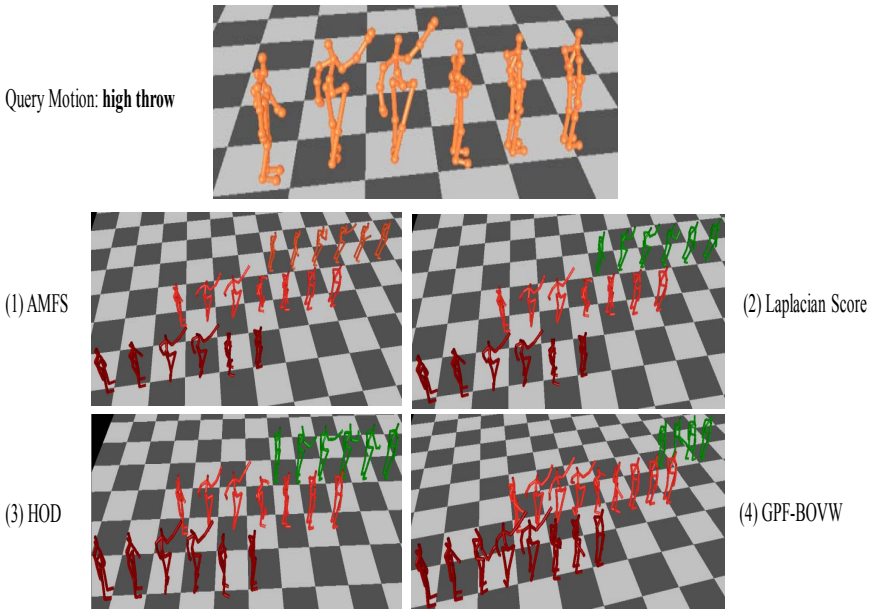
$$\alpha_v = \frac{(\frac{1}{tr(W^T X L^{(v)} X^T W)})^{\frac{1}{r-1}}}{\sum_{v=1}^{M}(\frac{1}{tr(W^T X L^{(v)} X^T W)})^{\frac{1}{r-1}}} \tag{5.24}$$

Algorithm 5 Optimize α with fixed W

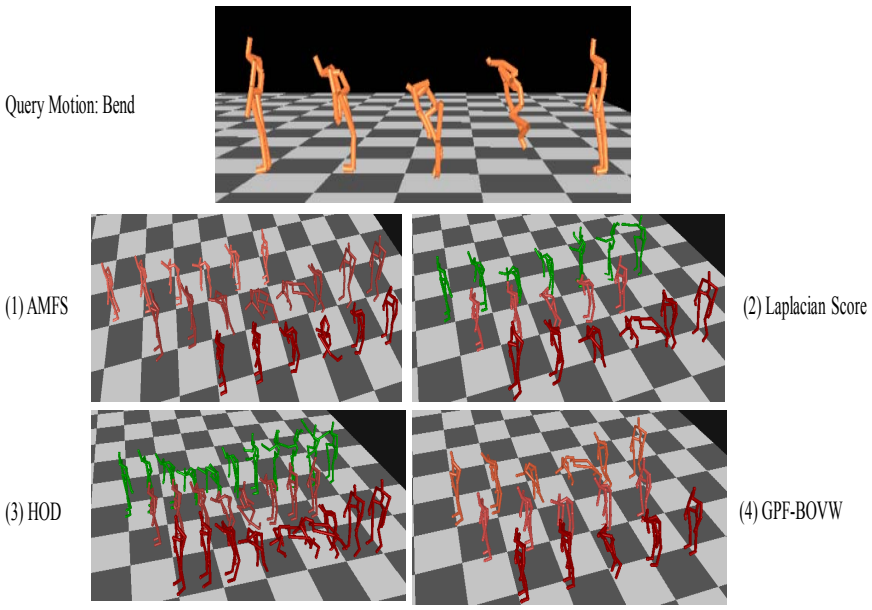
1: **for** $v = 1$ to M **do**
2: $\quad f_v = tr(W^T X L^{(v)} X^T W)^{-\frac{1}{r-1}}$;
3: **end for**
4: $F = \sum_{v=1}^{M} f_v$
5: **for** $v = 1$ to M **do**
6: $\quad \alpha_v = \frac{f_v}{F}$;
7: **end for**
8: **output** α

The updating of W and α should be recursively continued until it achieves convergence. Then, the local minimum solution for selection matrix W and feature weight α to the objective function can be obtained.

From equation 5.24, it needs to be pointed out that parameter r can modulate the smoothness difference between graphes [Wang et al. 2009a]. If $r \to 1$, the difference between each graph would be expanded, only α_v of the smoothest graph would be close to 1, which is the "select the best one among multiple features" that mentioned at explained of equation 5.19 and 5.20. If $r \to \infty$, the effect of difference is reduced and α_v for each feature would be close to each other. Thus, the choice of parameter r depends on the complementation of the multi-view features. The value of r should be large to explore the synergistic effect of multi-view features while rich complementation exists. Otherwise r should be small to keep the performance of the "best" feature.



(a) High throw



(b) Bend

Figure 5.2: *Example of motion retrieval results with AMFS, laplacian score, HOD and GPF-BOVW. The top three ranking results (from bottom to top) are given by each method, where red stands for the same action class with input and green stands for different action class.*

5.3 Experiment

5.3.1 Experiment Setup

Experimental Dataset Preparation

The proposed AMFS method has been tested on a commonly used public mocap database HDM05 dataset [Müller et al. 2007b] and MSR-Action 3D (MSR3D) dataset [Li et al. 2010c] to prove the efficiency of AMF-S for motion retrieval. HDM05 dataset consists 2061 motion sequences belonging to 130 classes performed by 5 actors. Since many of original classes are close to each other, e.g. *punchLFront1Reps and punchL-Front2Reps* the author has combined similar action classes and finally generate 25 classes for 2061 motion clips. To further challenge the proposed method, the author chooses the training and testing action sets that performed by different actors, e.g. 2 for training and 3 for testing. MSR3D has 567 motion sequences that belongs to 20 action types with 10 actors, while each actor performs each action 2 or 3 times. The 20-joints locations extracted by [Shotton et al. 2013a] are used instead of original depth image of MSR3D, where the author randomly chooses half of motion clips from each action class to be training data and the others are used as testing data.

Feature Selection

At the beginning of AMFS, different kinds of features should be chosen to form the multi-feature representation. The power of AMFS is based on the complementarity between different types of features. During experiments, it can find that increasing features of similar type makes little effect on the final retrieval performance. Thus, the author has decided to choose only one typical sequence based feature HOD [Gowayyed et al. 2013] and one pose based feature GPF [Chen et al. 2011] that both specially developed for motion data. Besides, for GPF, after feature extraction, a codebook is learnt via *kmeans*, then bag of visual word (BOVW)

method *LLC* [Wang et al. 2013b] is applied to generate a fixed length GPF-BOVW feature representation.

Performance Evaluation and Comparison

In this experiment, firstly, the author compares the performance of the proposed AMFS with the original two features on HDM05 dataset and MSR3D dataset. To exclude the factor of classification/ranking method, the simple Euclidean distance is used to rank the result for all the methods. Moreover, a retrieval performance test is also taken on both two datasets. It examins the performance of AMFS and two basic features. Moreover,several existing multi-view feature combination methods, Laplacian Score [He et al. 2005], Feature Ranking [Zhao and Liu 2007] and Unsupervised Discriminative Feature Selection (UDFS) [Yang et al. 2011; Ma et al. 2012], are also implemented for comparison. At last, AMFS's parameter sensitivity is tested and the convergence speed is examined.

5.3.2 Performance Experiment Result

In Fig 5.2,, it provides the retrieval result for two example query motion with 4 methods: AMFS, HOD, GPF-BOVW and Laplacian score. It shows the top three ranking results for every query motion given by each methods. AMFS generally outperforms the competitors.

The results in figure 5.3 show that our AMFS consistently outperforms using single features individually on both HDM05 and MSR3D datasets. The overall precision and detail performance on single action class provide the evidence that AMFS can combine the original features synergistically. The comparison of AMFS, and other multi-feature combination mehtods shows that AMFS is a effective method to generate relevance feature subset for human motion retrieval.

Figure 5.4 shows the retrieval performance of AMFS , two single feature algorithms and Laplacian score on two datasets. AMFS generally

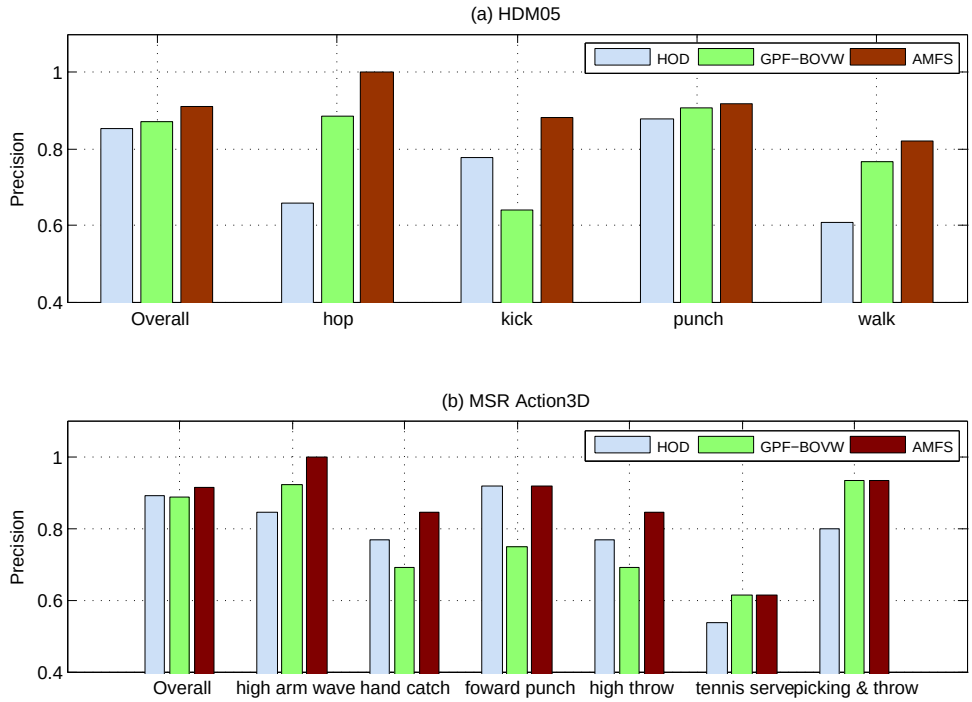


Figure 5.3: *Performance comparison of our AMFS algorithm and single features on HDM05 and MSR Action3D. Overall accuracy and detail result of some example action classes are shown. Actions: (a)HDM05 hop, kick, punch, walk, (b)MSR Action3D high arm wave,hand catch, forward punch, high throw, tennis serve, pick up and throw.*

outperfoms other methods. The comparison of AMFS and the competitors shows that AMFS is benefited from the complementary of different features, which provides a better achievement.

5.3.3 Algorithm Parameter Sensitivity

The result of the sensitivity test for parameter μ and r is reported in figure 5.5, where (a) is the test on HDM05 dataset and (b) is the test on MSR Action3D dataset. The variance range of parameter r is from 2 to 16 and the variance range of μ is $[0.01, 0.05, 0.1, 0.2, 0.3, 0.4, 0.5]$. The result demonstrates that our AMFS is robust to the parameter changing, meanwhile, it can still achieve a high retrieval accuracy even the parameter is not initialized carefully. Besides, the effect of parameter k, which is the parameter used to build the feature local graph, has also been tested. The result is shown in figure 5.6.

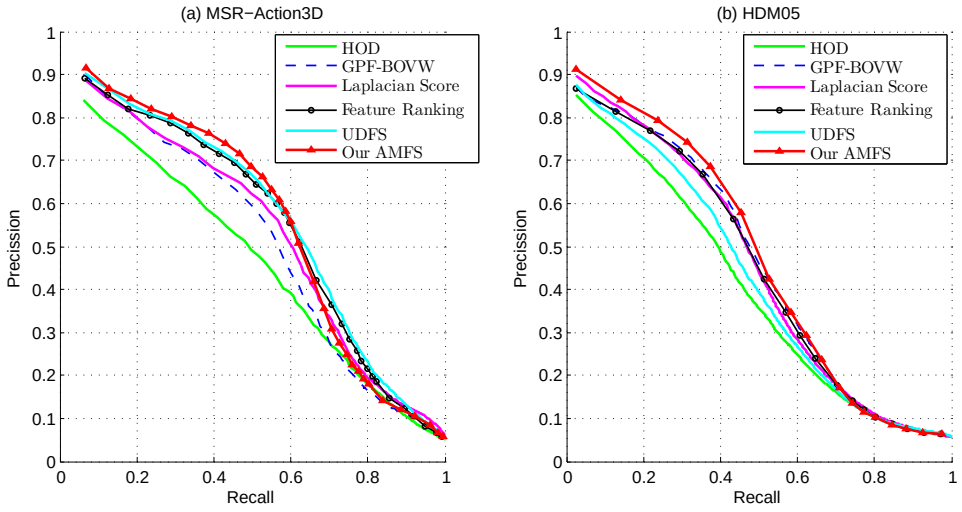


Figure 5.4: *Retrieval Performance of our AMFS algorithm: (a) test on HDM05 dataset, (b) test on MSR3D dataset.*

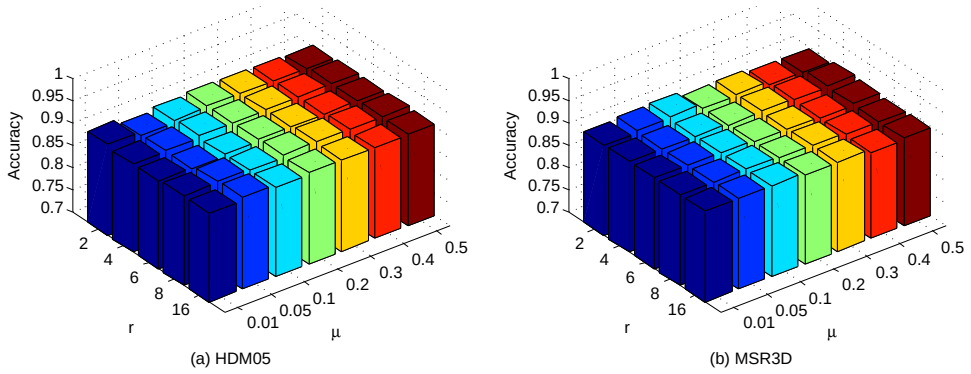


Figure 5.5: *Performance variations of our AMFS algorithm with different parameter r and μ: (a) test on HDM05 dataset, (b) test on MSR3D dataset.*

As it mentioned in previous section that the objective function is solved using an iterative approach, the speed of reaching convergence is crucial for the computational efficiency in the real world motion retrieval. The result of convergence examination on HDM05 dataset is given in figure 5.7. It noticed that AMFS can reach convergence within 20 iterations. Thus, it demonstrates that the designed optimization approach for AMFS is efficient.

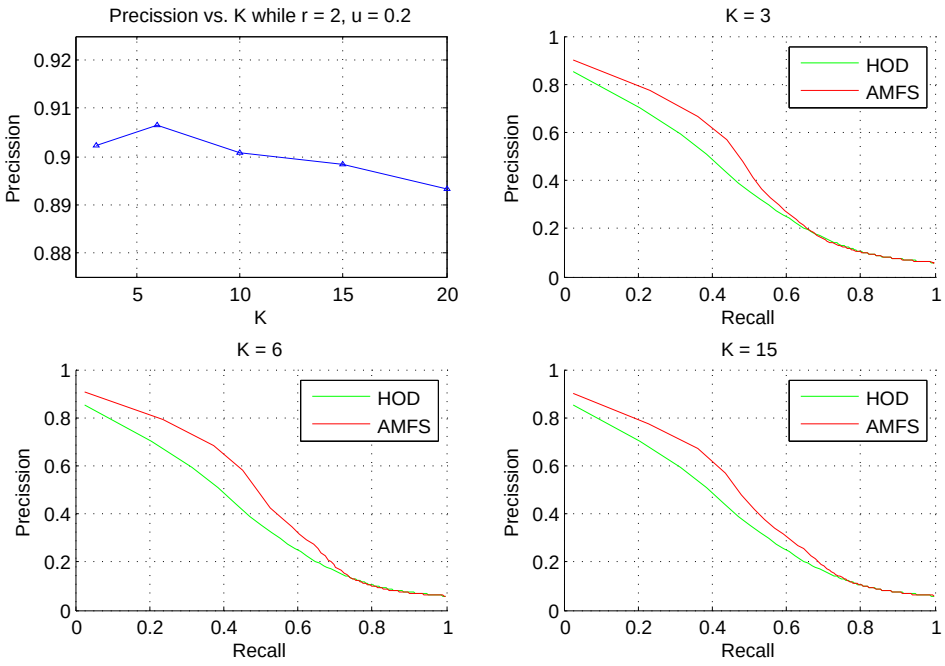


Figure 5.6: *Parameter sensitivity test for k on HDM05 dataset.*

5.4 Discussions

Generally the AMFS improves the motion retrieval performance. It simultaneously discovers the intrinsic relation between visual categories in a compact and relevance selected feature subspace by leveraging the underlying data structure and similarity between different feature views. It is fast, insensitive to parameters and robust to large scale data.

Comparing with previous work on motion features that only using small subset motion data, e.g. 251 actions from HDM05 [Ofli et al. 2012; Gowayyed et al. 2013], the whole HDM05 dataset is employed for performance testing. The performance on the larger dataset shows that our AMFS is able to handle the challenge from real-world big data. Besides, AMFS can directly be used if a new effective motion feature is proposed in the future. Any new motion features can be easily integrated into our framework. All of these properties make it applicable for real world problem solving.

However, one thing still need to be pointed out is that, as indicated in

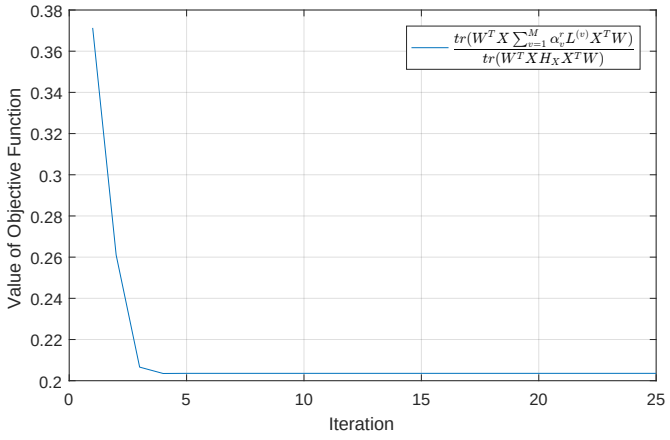


Figure 5.7: *Illustration of the convergence of AMFS on HDM05 dataset.*

[Yang et al. 2013], AMFS can not guarantee the higher performance from single features if the data does not hold a manifold structure. Moreover, the author has find that added similar type of features will bring little effect to the performance. Besides, fluctuation could occur during experiment when the number of features increased, this is a problem need to be solved in future.

5.5 Summary

In this chapter, a multi-view feature selection method has been proposed to categorize motion data. A trace-ratio objective function has been built while an iterative optimization approach has been provided. The AMFS discovers the intrinsic relations between visual words in each motion feature subspace to improve the retrieval performance. The performance of AMFS method has been evaluated on the HDM05 and MSR3D motion datasets. In order to deal with the real-world problem, all the 2061 actions in HDM05 dataset are employed for performance test, and the author has chosen different actors of motion for training and testing. The experimental results show that the AMFS method can improve the performance by combining different motion features synergistically and it has the potential to be implemented in the real large motion dataset retrieval.

Chapter 6

Motion Recognition

Vision based motion recognition is a interdisciplinary challenging field having grand application with social, commercial, medical and education benefits. Basically, video based motion data is highly related to the MoCap data. MoCap data is a higher level representation of human motion that summarized from video based motion data. Meanwhile, the corresponding video based motion data will also be available once the MoCap data is acquired. In addition, the cost of recording video based action data is much lower. Hence, it's meaningful to start the research on human motion recognition from video based data, then move on to the 3D MoCap data.

The wide spectrum of applications demands human motion recognition. The video based motion recognition attract lots of research interests and the fast growing amounts of motion videos on the internet, e.g. *Youtube*, provide plenty resources. In the first section of this chapter, a multi-task convolutional neural network (CNN) model is trained to solve 2D video-based gait recognition problem, which is a collaborated work with Dr Yan and Dr Qian. The author has contributed to the model building and parameter optimization. Another CNN model is presented in the second section of this chapter, which is used for MoCap data recognition. A introduction of CNN architecture and training tips are listed in Appendix .2 and .3. Finally, a NMF based MoCap data clustering method is presented in section 6.3, which is used for unsuper-

vised/semisupervised scenario.

6.1 2D video based gait recognition

Gait is a biometric feature that can be measured remotely without physical contact and proximal sensing. The research of gait recognition is strongly motivated by the demands of security that require automatically identifying person at a distance. Here, it proposes a robust and effective gait recognition approach using convolutional neural networks(CNNs) and multi-task learning model(MTL). Firstly, Gait Energy Image(GEI) is extracted from each walking period as the low level input for the CNNs. A multi-task CNN model is trained through back-propagation using a joint loss of each task. Then, the high-level features for multiple tasks are extracted simultaneously with the given input. The work flow is shown in Fig 6.1.

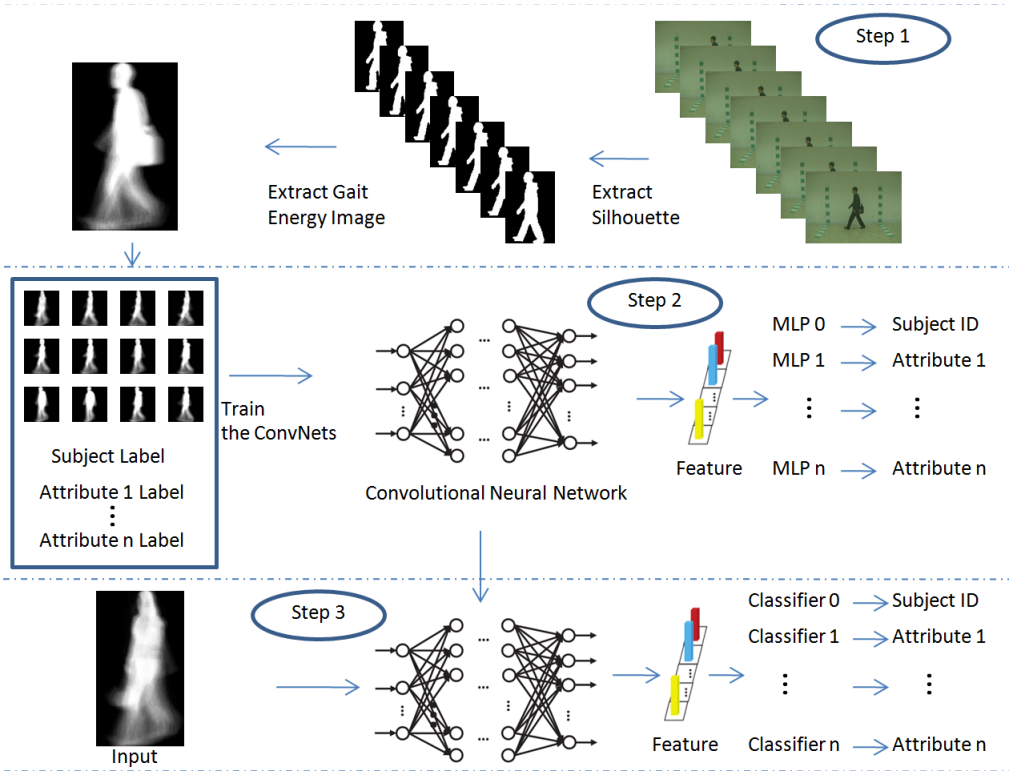


Figure 6.1: *The frameworks of proposed method.*

Generally, there are two major advantages of the proposed approach:(i)

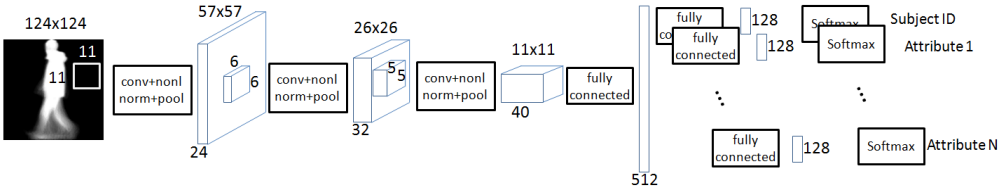


Figure 6.2: *The architecture of proposed convolutional neural network.* (conv) *stands for convolutional layer,* (nonl) *stands for nonlinear activation function,* (norm) *stands for normalization and* (pool) *stands for the max-pooling layer. The network contains three stages, each of which is consisted of convolution layer, non-linear activation layer, local response normalization layer, and max-pooling layer. Only convolution and max-pooling layers which change the data size during operating, are illustrated here.*

Table 6.1: *Architecture of proposed CNN for gait recognition*

Layer	Type	Feature map size and neurons			Kernel
1	Input	1 map 124×124			
2	Convolution C_1	24 maps 114×114			11×11
3	Max pooling	24 maps 57×57			2×2
4	Convolution C_2	32 maps 52×52			6×6
5	Max pooling	32 maps 26×26			2×2
6	Convolution C_3	40 maps 22×22			5×5
7	Max pooling	40 with 11×11			2×2
8	Fully connection	512			
		Task 1	Task 2	Task 3	
9	Fully connection	128 neurons	128 neurons	128 neurons	
10	Softmax output	124 neurons	11 neurons	3 neurons	

semantic features are directly learned via CNNs, which requires minimal domain knowledge of the problem, (ii) multi-attributes learning can improve the gait identification accuracy and increase the convergence speed for training.

6.1.1 Proposed CNN architecture

The overall CNN architecture is shown in Fig.6.2. The network consists of three convolution stages followed by one fully connected layers, which is determined by using the architecture selection strategy mentioned before. Each convolution stage includes convolutional layer, non-linear activation function, local normalization and pooling layer. The lower layers are finally shared by N split layers, each of which is fulled connected with the topmost shared layers. Each split layer is respected to one attribute

identification task and contains several additional fully connected layers with size 128. The size of split layer is k, which equals to the number of respected tasks and it is set as 3 in this work. The parameter of each layer is determined by the cross validation test.

6.1.2 Experimental test and Results

The performance of the proposed design is verified on CASIA gait database B, achieved over 95.88% accuracy for each task. The proposed method is also compared with the state-of-art approaches using CASIA gait database A and OU-ISIR Treadmill dataset A, where the result has demonstrated competitive performance.

- **CASIA-B** The CASIA gait database B contains 124 subjects, where the recording action sequences are under variant view angle and extra walking conditions. The view angle uniformly varies from 0°to 180°, which contains 11 classes. In addition, three are three working condition categories: walking with a coat(**cl**), walking with a bag(**bg**), and normally walking without anything(**nm**). Each subject walked 10 times in the scene (6 **nm** + 2 **cl** + 2 **bg**). A leave-one-out test is chosen to verify the experiment performance. The author randomly holds out one from six **nm** scene , one from two **cl** scene and one from two **bg** scene , that is , a total of $(1 + 1 + 1) \times 11 \times 124 = 4092$ videos for testing. the remained are used as training data.

- **CASIA-A** The CASIA gait database A [Wang et al. 2003] includes 20 subjects, each of which contains three views, namely frontal (0°), oblique (45°) and lateral (90°) views. The author sets up a two-task experiment including subject identification and view prediction. Half of the actions from each subject is chosen randomly as test and the left is for test.

- **OU-ISIR Treadmill dataset A** The OU-ISIR gait database: Treadmill dataset A [Makihara et al. 2012] contains six different walking speeds from 2 to 7 km/h with 1 km/h interval. A total

of 34 subjects are used in this experiment. The author sets up a two-task experiment including subject identification and walking speed prediction. Half of the actions from each subject is chosen randomly as test and the left is for test.

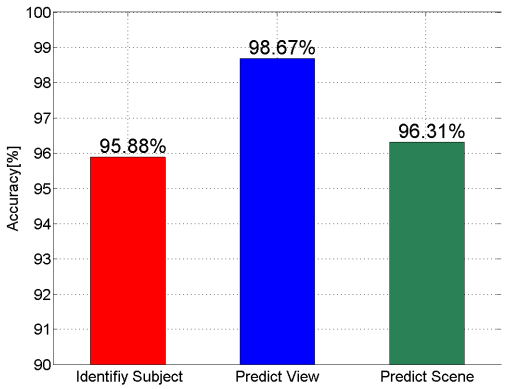


Figure 6.3: *The average performance of multi-task learning on CASIA-B for all three tasks.*

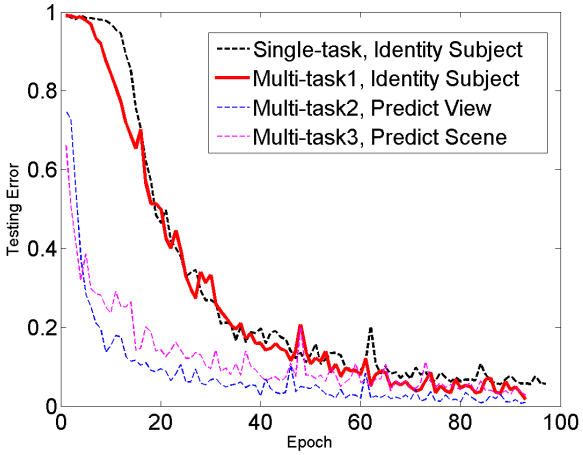
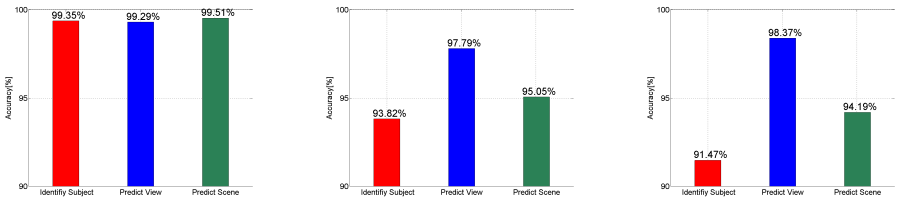


Figure 6.4: *The comparison performance of multi-task learning on CASIA-B for single task learning vs. multi-task learning.*

Evaluating on CASIA-B The overall performance are shown in Fig 6.3. The average accuracy of identifying subject, predicting view and predicting scene are 95.88%, 98.67% and 96.31%, respectively. The performance comparison for single task learning vs. multi-task learning using same CNN architecture is provided in Fig 6.4, where the result has shown that multi-task learning can improve the recognition accuracy.

Table 6.2: *The performance of multi-task learning on CASIA-B from all the 11 views. The probe viewing angles are (a) 0°, (b) 18°, (c) 36°, (d) 54°, (e) 72°, (f) 90°, (g) 108°, (h) 126°, (i) 144°, (j) 162°, and (k) 180° respectively.*

		Three Tasks		
		Identify Subject	Predict View	Predict scene
Probe View	0°	96.09	97.55	91.08
	18°	98.68	98.99	95.10
	36°	97.82	98.70	95.68
	54°	94.94	99.01	97.69
	72°	95.80	98.41	97.98
	90°	95.80	99.63	97.11
	108°	95.23	94.68	95.97
	126°	95.80	99.23	98.55
	144°	94.08	98.70	97.98
	162°	94.37	99.85	96.25
	180°	96.09	99.11	95.39
Probe Scene	nm	99.35	99.29	99.51
	cl	93.82	97.79	95.05
	bg	91.47	98.37	94.19
Average		95.88	98.67	96.31



(a) nm (b) cl (c) bg

Figure 6.5: *The performance of multi-task learning on CASIA-B from all the 3 scenes. The probe scenes are (a) nm, (b) cl, and (c) bg respectively.*

The detailed performance of three tasks in different probe view are illustrated in Table 6.2. It is observed that subject identification task has higher accuracy in the probe view of 180°, 0°, 18°and 36°than other views, which means gait is easier to identity in front view than lateral view. However, scene is easier to predict in lateral view than front view. Table.6.3 shows the performance of multi-task learning on CASIA-B from all the 3 scenes. The **cl** and **bg** scene attribute yield a negative effect to all tasks, especially for subject identification task.

Table 6.3: *Comparisons with other existing methods under changes of clothing and carrying condition*

Gallery-Probe	nm	bg	cl	Avg
LF+AVG[Hu et al. 2013a]	71.4	63.1	60.7	65.1
LF+DTW[Hu et al. 2013a]	61.9	17.9	0.0	26.6
LF+oHMM[Hu et al. 2013a]	63.8	31.8	21.4	39.0
LF+iHMM[Hu et al. 2013a]	94.0	64.2	57.1	71.8
GEI+PCA+LDA[Sarkar et al. 2005]	90.5	3.6	3.6	32.6
GPPE[Jeevan et al. 2013]	93.4	62.2	55.1	70.2
GEnI[Bashir et al. 2009]	92.3	65.3	55.1	70.9
STIP [Kusakunniran 2014]	94.5	81.5	82.3	86.1
The proposed	**99.4**	**97.8**	**94.2**	**97.1**

The proposed method is further compared with existing methods, under changes of scene, that is walking with a coat(cl), walking with a bag(bg), and normally walking without anything(nm). The comparison among best reported results are shown in Table.6.3. The proposed method significantly outperforms other existing method under each probe scene (subject wearing conditions).

Evaluating on CASIA-A Fig.6.6, illustrates the comparison result with other six methods using their best reported performance, including 3D deformation [Goffredo et al. 2008], 2D polar-plane [Chen and Gao 2007], Neural network [Lee et al. 2008b], PSC-PSA [Kusakunniran et al. 2011], Partial silhouette [Shaikh et al. 2014] and STIP [Kusakunniran 2014]. Although, the performance of proposed method is limited by the amount of training data, it still demonstrated a competitive performance.

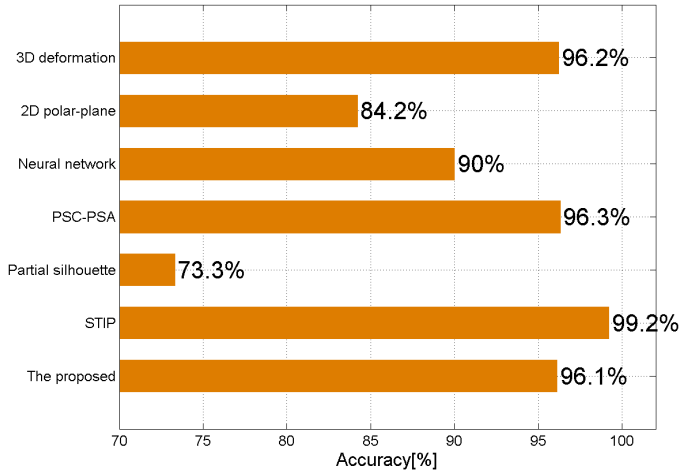


Figure 6.6: *Performance on CASIA-A: Best reported subject identification accuracy is compared with other six approaches including 3D deformation [Goffredo et al. 2008], 2D polar-plane [Chen and Gao 2007], Neural network [Lee et al. 2008b], PSC-PSA [Kusakunniran et al. 2011], Partial silhouette [Shaikh et al. 2014] and STIP [Kusakunniran 2014].*

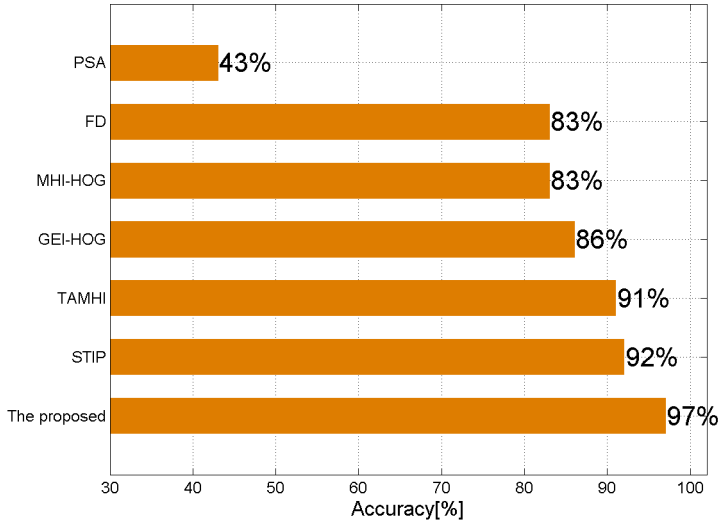


Figure 6.7: *Performance on Treadmill dataset A: Best reported subject identification accuracy is compared with other six approaches including PSA [Wang et al. 2003], FD [Lee et al. 2013], MHI-HOG [Huang et al. 2011], GEI-HOG [Whytock et al. 2013], TAMHI [Lee et al. 2014] and STIP [Kusakunniran 2014].*

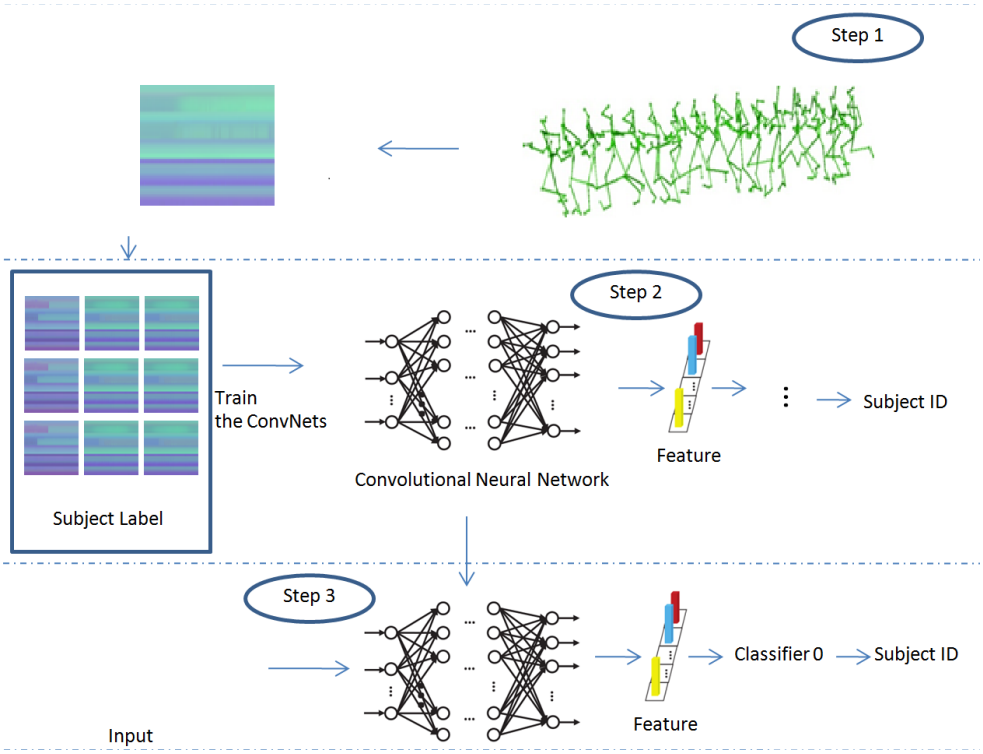


Figure 6.8: *Overview of the workflow of MoCap data recognition*

Evaluating on Treadmill dataset A In Fig.6.7, the best reported average performance of subject identification is compared with other six methods including PSA [Wang et al. 2003], FD [Lee et al. 2013], MHI-HOG [Huang et al. 2011], GEI-HOG [Whytock et al. 2013], TAMHI [Lee et al. 2014] and STIP [Kusakunniran 2014]. The proposed method significantly outperforms other existing methods.

6.2 3D MoCap data recognition

After using CNN model for gait recognition in previous section 6.1 , a further work on using CNN for 3D MoCap data recognition is proposed. Understanding of 3D motion is meaningful for reusing existing MoCap data to create character animations. In addition, it is also essential for multi-discipline applications such as physical training, sport association, rehabilitation and medical purpose. A whole work-flow of proposed motion recognition is presented in Fig. 6.8

A workstation with a Xeon 3.3GHz CPU. 48 GB memory and a GTX Titan X GPU was employed. The programs run on a 64-bit Windows 7 operating system with CUDA 7.5, cuDNN v5. Matlab 2015b and MatConvNet 1.0-beta20 deep learning platform. Inspired by the framework of *VGG* network [Simonyan and Zisserman 2014b], a similar architecture is used in this work. In the following the detail of model will be introduced, the experimental setup and results will also be demonstrated.

6.2.1 Proposed CNN Model

As shown in Fig 6.9, the raw motion capture data sequence will be firstly translated into a histogram based feature map, which is inspired by the Gait Energy Image that used in Section 6.1 before. After that, the proposed CNN model will be trained using the train dataset of NTU-RGBD.

The architecture of the proposed CNN model is based on the 16-layer network (VGG16) [Simonyan and Zisserman 2014b], which has demonstrated excellent performance in image classification as well as object detection. The detail network configurations are illustrated in the Table 6.4. All the constitutional layers are activated by Rectified Linear Unit (ReLU), and zero padded for preserving dimensionality. The kernel size of all max pooling layers is set as 2×2 and the stride is set as 2. The first and second fully connected layers are also activated by ReLU activation function, followed by dropout operation with drop rate 0.5 for preventing over-fitting.

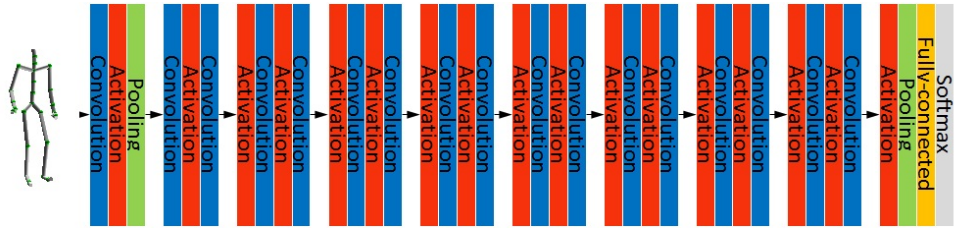


Figure 6.9: *Illustration of the* $11-$ *layer VGG-Net*

The training of the network consists of two stages namely, pre-training and fine-tuning. In pre-training stage, a pre-trained VGG16 network ini-

Weighted layers	Classification	Motion Histogram Map
	Input	
1	Convolution $3 \times 3, 64$	
2	Convolution $3 \times 3, 64$	
	Max pooling	
3	Convolution $3 \times 3, 128$	
4	Convolution $3 \times 3, 128$	
	Max pooling	
5	Convolution $3 \times 3, 256$	
6	Convolution $3 \times 3, 256$	
7	Convolution $3 \times 3, 256$	
	Max pooling	
8	Convolution $3 \times 3, 512$	
9	Convolution $3 \times 3, 512$	
10	Convolution $3 \times 3, 512$	
	Max pooling	
11	Convolution $3 \times 3, 512$	
12	Convolution $3 \times 3, 512$	
13	Convolution $3 \times 3, 512$	
	RoI pooling	
14	FC 4096	
15	FC 4096	
	Softmax loss 60	

Table 6.4: *Network configurations of CNN used in proposed motion recognition system.*

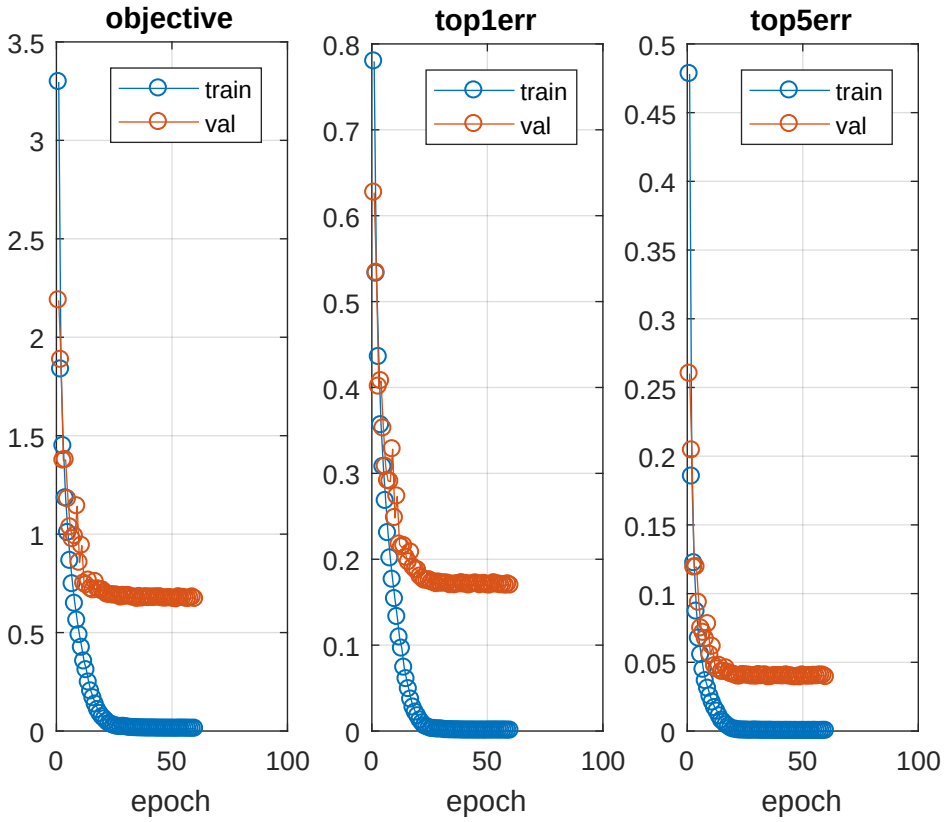


Figure 6.10: *Training Details of the Motion Recognition Network*

tialized on ImageNet is loaded. The last fully connected laye is replaced with new fully-connected layers that inialized with Improved Xavier [He et al. 2015], followed by soft-max loss. In fine-tuning stage, Stochastic Gradient Descent (SGD) is adopted to update network parameters. Each mini-batch is constructed from 2 input maps, where are selected randomly and uniformly. The learning rate is set to be 0.01, momentum is set to be 0.9 , weight decay is set to be 0.0005 and maximum training epoch is set to be 100.

The training performance is shown in Fig. 6.10, where the training objective function achieves convergence at around 50 epochs.

6.2.2 Experimental Results

The experimental setup follows same data preparing as NTU-RGBD dataset suggested. The proposed model has archived an overall accuracy of 83.08%. A comparison of the proposed model and existing frames are illustrated in the table below. Generally, the proposed method has reached a competitive performance compare with the state-of-art approaches.

Approaches	Methods	Accuracy (%)
[Huang et al. 2017]	Deep Learning on Lie Groups	61.37
[Shahroudy et al. 2016]	2 layer P-LSTM	62.93
[Liu et al. 2016a]	Spatio-temporal LSTM + Trust gate	69.2
[Zhang et al. 2017]	3 layer LSTM	70.26
[Lee et al. 2017]	Ensemble Temporal Sliding LSTM	74.60
[Ke et al. 2017a]	SkeletonNet (CNN)	75.94
[Li et al. 2017]	JDM + CNN	76.20
[Ke et al. 2017b]	Clips + Multi TaskCNN	79.57
[Liu et al. 2017b]	Enhanced Visualization + CNN	80.03
[Ding et al. 2017]	5 features fusion map + CNN	82.31
The proposed Method	3D Energy Image + CNN	**83.05**

Table 6.5: *Deep Learning frameworks' performance on NTU-RGBD dataset (cross-subject) using skeleton modality*

6.3 NMF based Motion Clustering

A key problem of the standard NMF is that both the local geometry structure of data X and the label information are not well explored. In addition, the original objective function is only optimal to the Gaussian noise or Poisson noise, which may be not able to deal with other noise cases, e.g. outliers. Therefore, a robust NMF model will be designed in this research, where a locality-graph regulation term will be obtained in the objective function to explore the geometry structure of data and the label information.

6.3.1 Locally weighted sparse graph regularization

Existing research shows that the samples lie on a low dimensional intrinsic manifold embedded in a high-dimensional ambient space [Cai et al. 2008]. In order to explore the geometry structure of data or the label information in the proposed model, it will develop a feature representation that incorporates the clustering information encoded in graph data. In the following part of this section, the local graph model will be introduced.

Recall that NMF tries to find a basis that is optimized for the linear approximation of the data. Exploiting the knowledge of the data distribution will be helpful for better discovery of this basis. A natural assumption here then could be made : if the two data points x_i, x_j are close to each other in the intrinsic geometry of the data distribution, then the representations of this two points in the new basis after mapping should also be close to each other, which is usually referred to as manifold assumption [Belkin and Niyogi 2001].

In reality, the data manifold is usually unknown. Existing research on spectral graph theory [Chung 1997] and manifold learning [Belkin 2003] have demonstrated that the smoothness of mapping along the geodesics in the intrinsic geometry of the data can be discretely approximated through a nearest neighbor graph on a scatter of data points.

6.3.2 Objective function

The standard NMF is optimal to the Gaussian distribution noise. The designed model will consider the local geometry information of input data. Hence, the objective function is written as:

$$\begin{aligned} &\min_{Z,G} \|X - ZG\|_F^2 + \lambda_1 \cdot \|G\|_F^2 + \lambda_2 \cdot \Theta(G) \\ &s.t.\ Z \geq 0.G \geq 0, Z^T Z = I_k, GG^T = I_k \end{aligned} \tag{6.1}$$

In the objective function shown in equation 6.1, the first term is the

square fitting error ,the second term controls the energy of G and the last term is to preserve the geometrical structure in the low dimensional space or the label information. Therefore, the author explicitly consider the noise's, the scalar property of data and the local geometrical structure of data in a same frame work simultaneously.

However, the input X may be not normalized in practice. If it regularizes $Z^TZ = I_k$ and $GG^T = I_k$ at same time, the deviation of fitting result could be large. Hence, the constraint of G is relaxed and the regularization $Z^TZ = I_k$ is remained to kept which aims to ensure that the matrix factorization clustering centre is stable while the scaling effect is removed. Therefore, the objective function is formulated as:

$$\begin{aligned} &\min_{Z,G} \|X - ZG\|_F^2 + \lambda_1 \cdot \|G\|_F^2 + \lambda_2 \cdot \Theta(G) \\ &s.t.\ Z \geq 0.G \geq 0, Z^TZ = I_k \end{aligned} \tag{6.2}$$

The local weighted graph regulation term $\cdot\Theta(G)$ could be represented as $\cdot tr(GLG^T)$. Therefore, the objective function is finally formulated as:

$$\begin{aligned} &\min_{Z,G} \|X - ZG\|_F^2 + \lambda_1 \cdot \|G\|_F^2 + \lambda_2 \cdot tr(GLG^T) \\ &s.t.\ Z \geq 0.G \geq 0, Z^TZ = I_k \end{aligned} \tag{6.3}$$

6.3.3 Optimization method

To optimize equation 6.3, the Augmented Lagrange Multiplier (ALM) method is applied. The corresponding Augmented Lagrangian function $\mathcal{J}(Z, G)$ is defined as:

$$\begin{aligned} &\mathcal{J}(Z,G) = \|X - ZG\|_F^2 + \lambda_1\|G\|_F^2 + \lambda_2 tr(GLG^T) \\ &s.t.\ Z \geq 0, G \geq 0, ZZ^t = I_k \end{aligned} \tag{6.4}$$

where $X \in \Re^{d\times n}$, $G \in \Re^{d\times k}$ and $G \in \Re^{k\times n}$. An iterative method is proposed to solve the problem. Equation 6.4 is not convex in Z and G jointly but convex in each variable separately. Hence, the values of Z

and G are updated iteratively,while leaving the other one constant.

Fix G, optimize Z When G is fixed, the function of equation 6.4 can be rewritten as:

$$\begin{aligned} &\mathcal{J}(Z) = \min \|X - ZG\|_F^2 \\ &s.t.\ Z \geq 0, ZZ^t = I_k \\ \Rightarrow &\mathcal{J}(Z) = \|X - ZG\|_F^2 - tr(A(Z^TZ - I_k)) \\ \Rightarrow &\mathcal{J}(Z) = tr(-2X^TZG + G^TZ^TZG - AZ^TZ) \end{aligned} \tag{6.5}$$

where matrix $A \in \Re^{k \times k}$ is introduced as an auxiliary matrix, A_{ij} enforce nonnegative constraints $Z_{ij} \geq 0$. Then the gradient of 6.5 could be written as:

$$\frac{\partial \mathcal{J}(Z)}{\partial Z} = -2XG^T + 2ZG^TG - 2ZA \tag{6.6}$$

Let $\partial \mathcal{J}(Z)/\partial Z = 0$, since $Z^TZ = I_k$, it could get

$$\begin{aligned} &-2XG^T + 2ZG^TG - 2ZA = 0 \\ \Rightarrow &A = GG^T - Z^TXG^T \end{aligned} \tag{6.7}$$

Using the Karush-Kuhn-Tucker condition [Boyd and Vandenberghe 2004], $A_{ij}Z_{ij} = 0$. Substitute A_{ij} from equation 6.7, it obtains

$$(GG^T - Z^TXG^T)_{ij}Z_{ij} = 0 \tag{6.8}$$

which is a fixed point equation that the solution must satisfy at convergence. According to [Ding et al. 2010], it leads to the updating formula of Z_{ij}:

$$Z_{ij} \leftarrow Z_{ij}\sqrt{\frac{(XG^T)_{ij}}{(ZZ^TXG^T)_{ij}}} \tag{6.9}$$

In addition, due to the constraint $Z^TZ = I_k$, Z will be further normalized Z after every updating.

Fix Z, optimize G Similarly, when Z is fixed, the function of equation 6.4 can be rewritten as:

$$\begin{aligned}&\mathcal{J}(G) = \min \|X - ZG\|_F^2 + \lambda_1 \|G\|_F^2 + \lambda_2 tr(GLG^T) + \gamma \|GG^T - I_k\|_F^2 \\ &s.t.\ G \geq 0\end{aligned} \tag{6.10}$$

Introduce matrix $B \in \Re^{k \times n} \geq 0$, as an auxiliary matrix, A_{ij} enforce nonnegative constraints $Z_{ij} \geq 0$. The loss function is then reformulated as:

$$\mathcal{J}(G) = \|X - ZG\|_F^2 + \lambda_1 \|G\|_F^2 + \lambda_2 tr(GLG^T) + \gamma \|GG^T - I_k\|_F^2 - tr(BG^T) \tag{6.11}$$

Then the gradient of 6.11 could be written as:

$$\frac{\partial \mathcal{J}}{\partial G} = -2Z^T(X - 2G) + 2Z^T ZG + 2\lambda_1 G + 2\lambda_2 LG^T - B \tag{6.12}$$

let $\partial \mathcal{J} / \partial G = 0$, it obtains:

$$\begin{aligned}&-2Z^T(X - 2G) + 2Z^T ZG + 2\lambda_1 G + 2\lambda_2 LG^T - B = 0 \\ \Rightarrow &B = -2Z^T(X - 2G) + 2Z^T ZG + 2\lambda_1 G + 2\lambda_2 LG^T\end{aligned} \tag{6.13}$$

Using the Karush-Kuhn-Tucker condition [Boyd and Vandenberghe 2004], $B_{ij}G_{ij} = 0$. Substitute B_{ij} from equation 6.13, it obtains the

$$(-2Z^T(X - 2G) + 2Z^T ZG + 2\lambda_1 G + 2\lambda_2 LG^T)_{ij} G_{ij} = 0 \tag{6.14}$$

According to [Gu and Zhou 2009; Ding et al. 2010], it leads to the updating formula of G_{ij}:

$$G_{ij} \leftarrow G_{ij} \sqrt{\frac{(Z^T X)_{ij}}{(Z^T ZG + 2\lambda_1 G + 2\lambda_2 LG^T)_{ij}}} \tag{6.15}$$

Algorithm 6 NMF based Motion Clustering (NMC)

Input:

- motion feature matrix $X \in \Re^{d \times n}$;
- Laplacian graph matrix $L \in \Re^{n \times n}$;
- cluster number k;
- regularization parameter λ_1, λ_1.

Output:

- matrix factorization factor $Z \in \Re^{d \times k}$; $G \in \Re^{k \times n}$.

1: **Initialization**
Initialize Z and G using k-means.
2: **repeat**
3: Fixed G, update Z according Eq. 6.9
4: Normalize Z
5: Fixed Z, update G according Eq. 6.15
6: **until** Convergence
7: **Return** Z and G

6.3.4 Experimental Results

In order to evaluate the performance of proposed NMF based clustering method, MSR-Action3D dataset is employed, where 10 actors, 20 categories of motions, 566 action sequences and 22542 frames are contained in this dataset. In the experiment, half of them are served as training data, and the others are used as testing data. For a given **S** frames motion sequence, a moving window is **M** applied to the motion sequence. The moving stride is set as $M/2$ which is 30 in this experiment.

After the grouping operation, a clustering is taken with the proposed NMF algorithm 6 for all the clips. Then a bag-of-words feature is extracted to represent each motion sequence. It yields a overall recognition precision of 93.33% and the confusion matrix is shown in Fig 6.12. Unlike the supervised CNN model presented in the previous section 6.2, the proposed NMF method is mainly designed for unspervised/semisupervised learning scenario. Hence, the comparison between these two model will not be taken.

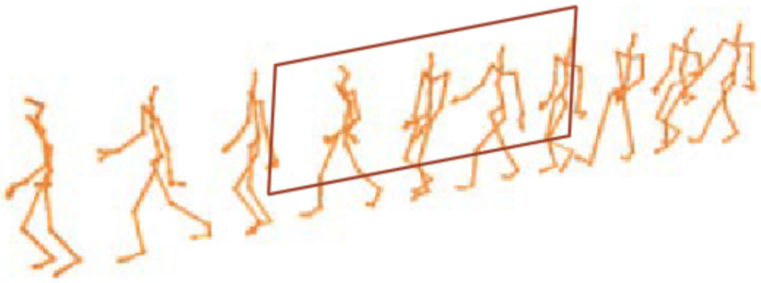


Figure 6.11: *the grouping operation, with a given S frames sequences and size M window, it will generate* $N = \frac{S-M+1}{M/2}$ *overlapping clips, which aims to obtain embedded spatial-temporal patterns.*

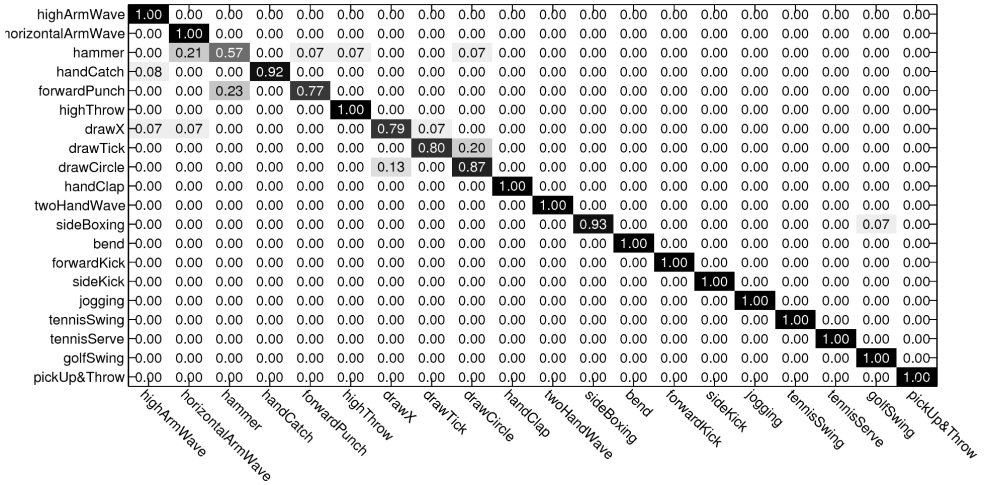


Figure 6.12: *The confusion matrix of the proposed method for individual motion recognition in MSR-Action3D dataset*

6.4 Summary

In this chapter, a novel approach to identify human gait and predict multiple attributes is presented, which use convolutional neural network(ConvNets) and multi-task learning model. The experimental result shows that the proposed method generally outperform the state-of-art methods in the perspective of accuracy and robustness. After that, a CNN based 3D motion recognition method are proposed in this chapter. The result shows that the proposed models have achieved a competitive performance compare with the state-of-art frameworks. Finally, a NMF based motion clustering method is designed for unsupervised/semisupervised learning scenario. In further work, the motion clustering method will be applied to data driven motion generation. The CNN based model will be further investigated for motion style analysis and style transferring.

Chapter 7

Conclusion and Future Work

7.1 Findings of this study

This study has proposed the notion of machine learning based motion capture data processing techniques for reusing purposes. It focused on three proposed questions:

- *How to acquire and collect the high standard quality motion data?*
- *How to effectively manage the large amount of existing motion data that is required by motion reusing?*
- *Can the automatic candidate motion recommendation be achieved through a learning model of action classification and style meanings?*

These questions have looked to be addressed in chapter 3 , 4 , 5 and 6. The study concluded that motion capture is a powerful technique for creating realistic 3D character animation. The limitations of motion capturing such as cost, actors and environment motivated this study in terms of MoCap data processing techniques for reusable purposes. This study has developed a dictionary learning based motion refinement method and a matrix completion based motion refinement method. In addition, the author has also designed a motion retrieval method and introduced deep learning techniques for motion recognition. The results

verified that developed motion refinement methods are able to handle the long term complex actions. Also, the testing result has evidenced that the designed retrieval model and recognition model can make accurate recommendations and outperforms the state-of-art competitors.

Answering question: How to acquire and collect the high standard quality motion data? Motion data quality problems influence all MoCap related applications. There are two main kinds of errors contained in the imperfect motion sequence: missing content and noise, where theoretically the missing value can be defined as an outlier and noise can be assumed as Gaussian noise. Although the capturing techniques may be improved to reduce the noise and outliers in the recording result, this study focused on developing the post-processing refinement techniques to address such problems. The reason is that the author proposed to take advantage of the large amounts of existing MoCap data. In order to refine the imperfect motion data, a dictionary learning based method and a motion matrix completion method have been designed in this study, where the experimental results have shown the effectiveness of the proposed refinement model.

Answering question: How to effectively manage the large amount of existing motion data that is required by motion reusing? In order to manage the huge amount of motion data, an effective motion retrieval method is desired. Feature representation plays an essential role in the retrieval method design, which determines overall performance. This study has proposed a multi-modality feature selection model to extract the effective motion feature for retrieval, which fuses multiple visual features for motion data representation. The experimental result shows that the proposed method can improve the performance by combining different motion features synergistically.

Answering question: Can the automatic candidate motion recommendation be achieved through a learning model of action classification and style meanings? To tackle this problem, the core

task is motion understanding, which requires designing motion recognition models. Statistical methods usually have limited computational capabilities for handling high-dimensional data with complex correlations. Machine learning methods are the dominant solutions for dealing with complex motion data and making accurate recognition for real-world applications. A NMF based model is designed in this study for unsupervised motion data clustering. Convolutional neural networks are also used in this study for motion classification and achieved great success. The experimental result shows that the proposed models have achieved a competitive performance compared with the state-of-art frameworks.

7.2 Contribution to new knowledge

Contribution to new understanding of motion capture data MoCap data itself is complex due to its sparsity and variance of directions, velocity and actors. It has intrinsic strong local connections inside the body part's movement. Hence, a partial-grouping model is designed in this study to exploit the kinematic spatial-temporal information, which yields a robustness and effective motion refinement model.

Contributions to new understanding of motion refinement In this study, the motion refinement problem is treated as a data matrix completion task, where a low-rank matrix completion model is designed. The truncated norm method is employed to tackle the low-rank matrix completion task. The proposed model has obtained a great refinement performance and is especially efficient on dealing with long-term action sequences.

Contribution to motion-oriented deep learning approaches In terms of the training recognition model, this study challenges the deep learning method, Convolutional Neural Networks on a particular action recognition task. The proposed methodology is based on the collaboration of conventional machine learning and deep learning algorithms for

each advantage on classifications and feature extractions.

Contribution to new understanding of multi-modality feature fusing Since different visual features describe different aspects about motion data, and they have dissimilar discriminative power with respect to one particular class of human motion, it would be beneficial to combine multiple visual features together for motion data representation. In this study, an adaptive multi-feature selection method is proposed to discover the intrinsic relations between visual words in each motion feature subspace to improve the retrieval performance. Specifically, a local linear regression model is used firstly to automatically learn multiple view-based Laplacian graphs for preserving the local geometric structure of motion data. Then, these graphs are combined together with a non-negative view-weight vector to exploit the complementary information between different features. Additionally, the objective function of AMFS model is formulated as a trace-ratio optimization problem and a corresponding iterative optimization approach is designed.

Generally, the proposed techniques and analysis of MoCap data are valuable for motion reusing purposes. The industry practitioners could be benefitted. In addition, the study on multi-modality features and deep learning models is also beneficial for researchers who are interesting in motion data processing.

7.3 Future work

Developing more efficient and effective human motion data processing methods that focus on motion data acquisition, multi-modality motion data and motion style learning would be an interesting future inspection.

Motion retrievalrecognition with multi-modality motion data. A continuous work on multi-modality based motion retrieval method design may be conducted based on the existing multi-feature selection

work. The purpose of this continuous work is to integrate the multi-modality data to support the trend of using multi-modally motion data. In the existing RGBD based human action recognition frameworks, some researchers are concentrating on developing features only from the depth image, which limited the representation of whole scene context and cannot describe the whole environment information precisely. In contrast, some researchers extracted the features from RGB image and depth image separately then just simply combine them together to form a high dimensional feature representation, which did not consider further research on effectively fusing these multi-modality features.

Motion style learning. The author is also interested in a further research on motion style learning. Human motion can be semantically represented as a combination of two factors: the content and style. The content generally refers to the action such as walking and running, whilst the style denotes high level patterns that motion data exhibits. The style of human motion provides important cues of personality, mood or mentality of the performer, which are essential for storytelling and for bringing animated characters to life. Traditional parametric style translation frameworks can only learn simple parametric motion models that do not fully capture the subtleties and complexities of human motions. Deep learning techniques have gained a greater popularity in areas including human motion, which is mainly concerned with motion recognition and retrieval. Existing CNN approaches can learn the style transfer model. However, such time series approaches are not suitable for animation production, since they need to compute the integration for the entire motion, which takes away the editing power from the animator. That is essential in exercising the animators artistic creativity. It would also be interesting to use the recent GAN model for stylized motion generation in the future to tackle this challenge.

7.4 Conclusion

This study has proposed machine learning based motion data refinement, retrieval and recognition techniques for motion reusing purpose The proposed refinement methods in this thesis can improve the MoCap data quality to meet with high level requirement applications. The designed multi-feature selection model can be used for motion retrieval, which is essential for large amount of motion data processing. Moreover, it can also be applied in further multi-modality design that consists of not only MoCap data but also EMG and EEG data. The proposed NMF and CNN methods can extract effective features and achieve state-of-art recognition performance. The proposed motion refinement, retrieval and recognition methods in this study are primary techniques for motion data reusing. In future, improvements could consider the multi-modality data processing and further verification on real-world multi-modality motion data. In addition, motion style learning and stylized motion synthesis are also interesting topics.

Appendix

.1 Lemma for parameter updating of TrN-N

The lemma is provided by Dr. Hu in the collarboration publication [Hu et al. 2017a].

Lemma 1 ([Hong et al. 2016])**.** *Assume $U \in \mathbb{R}^{r\times n}(r < n)$ satisfies $UU^T = I_r$. Then one of the optimal solutions of the following problem*

$$\max_{U} tr(UM) \quad s.t. \quad UU^T = I_r \tag{1}$$

is $U^* = QP^T$, where P and Q are given by the economy-size singular value decomposition of M: $M = P\Sigma Q^T$, $P \in \mathbb{R}^{n\times r}$, $Q \in \mathbb{R}^{r\times r}$, $\Sigma \in \mathbb{R}^{r\times r}$, $P^TP = I_r, Q^TQ = I_r$.

Proof. The Lagrangian function of problem (1) is defined as

$$L(X, \Lambda) = tr(UM) - \langle \Lambda, UU^T - I_r \rangle.$$

Then, the Karush-Kuhn-Tucker (KKT) conditions are

$$\begin{cases} M^T - (\Lambda^T + \Lambda)U = 0, \\ UU^T = I_r \end{cases}$$

which yields $(\Lambda^T + \Lambda) = M^TU^T = (Q\Sigma Q^T)(QP^T)U^T$ being a symmetrical matrix. Therefore, to make the right part symmetrical, one of the possibilities for U is $U = QP^T$. □

.2 Architecture of CNN model

The CNN is a multi-layer neural network, which includes many different building blocks. In this section, a basic annotation will be given, then the detail of each layer will be described.

.2.1 basic construction

A CNN is a combination of several layers, where each layer consists of maps and parameters. In practice, only parameters are needed to store for saving the model. For each input, the maps of each layer will be calculated according to this layer's parameter and the maps of the previous layer. Besides, a layer's input is exactly the previously layer's output, which is show on Fig 1. CNNs are hierarchical neural networks whose convolutional layers alternate with subsampling layers, reminiscent of simple and complex cells in the primary visual cortex. CNNs vary in how convolutional and subsampling layers are realized and how they are trained.

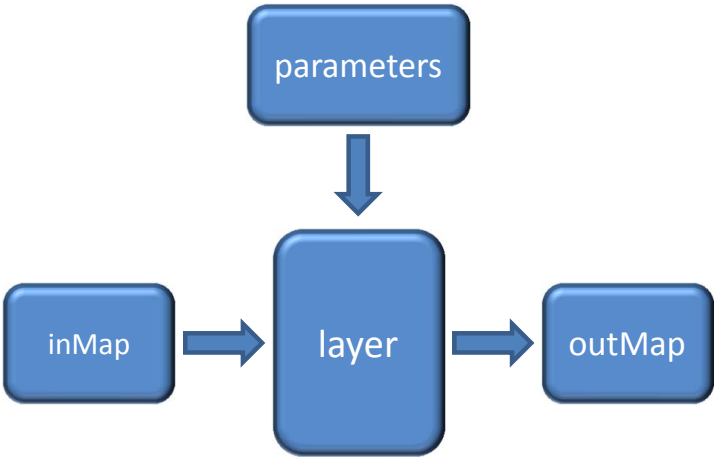


Figure 1: *Structure of a layer*

.2.2 Convolution layer

The purpose of constitutional layer is to extract the local patterns. It is parameterized by following factors: the number of maps, the size of the maps, kernel sizes and stride. For a given layer L^n, it has M_{in}^n input maps of equal size $(W_{M_{in}}^n, H_{M_{in}}^n)$. A kernel of size (W_k^n, H_k^n) is shifted over the valid region of the input. The stride defines the length of steps that the fileter/kernel shifts in w and h direction during subsequence convolution operation. The local receptive field, e.g. kernel/filter, is replicated across the entire visual field to form a feature map. The output map size M_{out}^n is then defined as equation 2

$$\begin{cases} W_{M_{out}} = \dfrac{W_{M_{in}} - K_w}{stride} + 1 \\ \\ H_{M_{out}} = \dfrac{H_{M_{out}} - K_h}{stride} + 1 \end{cases} \tag{2}$$

The (K_w, K_h) is the size of kernel and index n indicates the layer while the input maps M_{in}^n in layer L^n are connected to at most M_{out}^{n-1} maps in layer L^{n-1}. Here is an example shown in Fig 2.

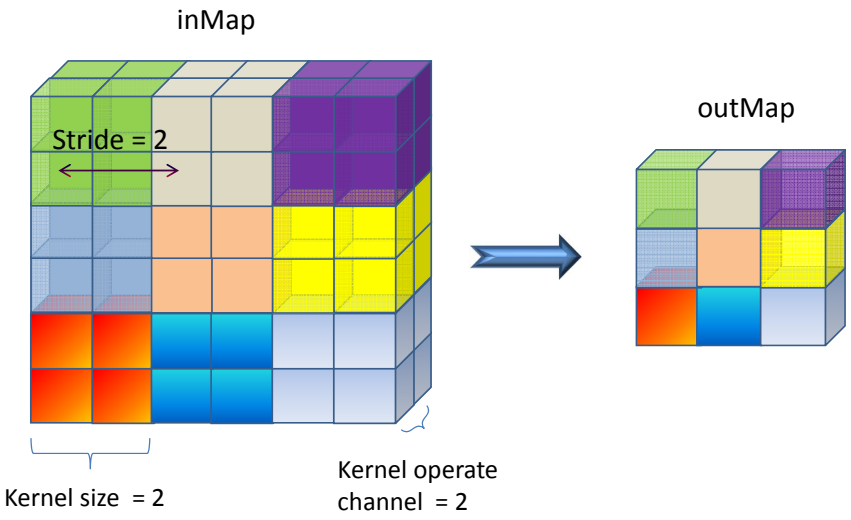


Figure 2: *An example of operation in convolutional layer: M_{in}(channel = 2, width = 6, height = 6), kernel(operator channel = 2, width = 2, height = 2, stride = 2), M_{out}(channel = 1, width = 3, height = 3)*

In order to reduce the number of parameter for training, a weight sharing principle is followed to learn the kernel. In other words, the parameter of one kernel (weight and bias) is same for all the input map in a given convolution layer. A kernel is defined with its weights and bias (w, b). The operation can de defined as

$$y^j = b^j + \sum_i w^{ij} \otimes x^i \tag{3}$$

where y^j denotes the jth output map, b^j is the summation result of bias, w^{ij} is kernel weight of ith input map x^i for output map y^j and $\otimes$ denotes the convolution operation. Therefore, the total amount of parameters for one kernel need to be determined is

$$M_{in}.channel * (K_w * K_h + 1)$$

Since the number of output map's channel is just the number of kernels, the total number of parameters to learn for a given convolution layer is

$$M_{in}.channel * (K_w * K_h + 1) * M_{out}.channel$$

.2.3 Nonlinear activation functions

Generally, the functionality of convolutional layer is similar to the linear filter. In order to form a nonlinear complex model, nonlinear action functions are needed to be applied where the input value will be transformed nonlinearly to the output value. *tanh* and *sigmoid* functions, which are shown in equation 4 and 5, are mainly used as nonlinear transformation in traditional neural networks.

$$sigmoid(x) = \frac{1}{1 + e^{-x}} \tag{4}$$

$$tanh(x) = \frac{e^{2x} - 1}{e^{2x} + 1} \tag{5}$$

Due to the upper and bottom bound of these functions, saturation problems could occur to neurons in high level layers when training a complex

CNN model. The saturated neuron could cause diminishing gradient flow to the lower layers of network, which will limit the maximum number layers for the training network model [Hochreiter et al. 2001].

Apart from sigmoid and hyperbolic tangent functions, the rectifier linear unit (Relu) and softplus, a smooth vision of Relu are also applied to CNN as activation function. The description of these four activation functions are shown in Fig 3. The nonlinear activation function usually follows the convolutional layer.

$$Relu(x) = max(x, 0) \tag{6}$$

$$softplus(x) = \log(1 + e^x) \tag{7}$$

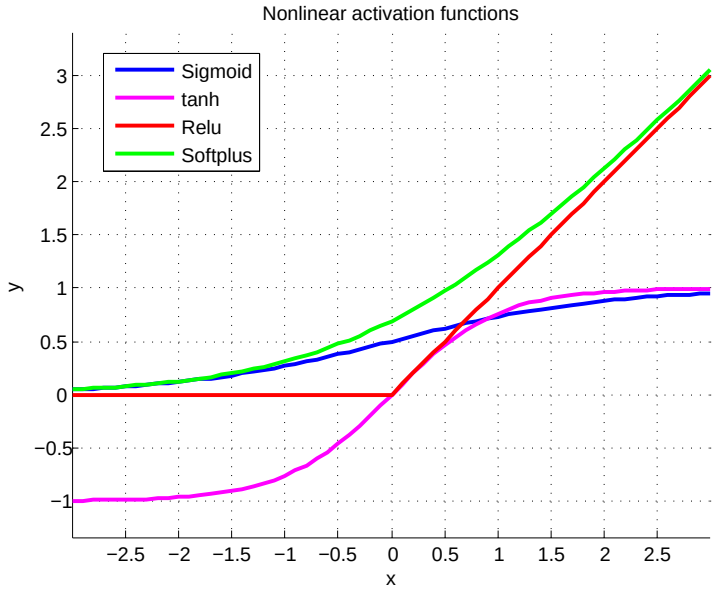


Figure 3: *Nonlinear activation functions: Sigmoid, tanh, Relu and Softplus*

Previously, Sigmoid is the typical activation function employed by the deep neural networks (DNNs). However, sigmoidal DNNs could suffer from the vanishing gradient problem. Vanishing gradients occur when lower layers of a DNN have gradients of nearly 0 because higher layer units are nearly saturated at -1 or 1. In contrast, Relu does not suffer from such kind of saturation problem. Additionally, Relu just require a

simple thresholding operation while Sigmoid need the expensive operation like exponential. Moreover, [Krizhevsky et al. 2012] show that using Relu can greatly accelerate the convergence speed of SGD compare with other activation functions such as Sigmoid and Tanh. However, the Relu units can "die" during the training since the neuron will never activate on any data input if a large gradient flowing through it. Setting learning rate properly can reduce the frequency of such kind of problem.

Relu and its variants Currently, the Relu is the most popular choice of activation function in deep learning models. There are some variations of Relu:

- Leaky Relu Leaky Relu is designed to fix the "dying" problem of Relu. Leaky Relu us a small negatice slop instead of 0 when $x < 0$. It is defined as:

$$Leaky\ Relu(x) = \begin{cases} x, x \geq 0; \\ \alpha x, x < 0. \end{cases} \tag{8}$$

  where α is a small constant.

- Parametric Relu The parametric Relu (PRelu) is similar to the Leaky Relu while the slopes of negative part are learned from data rather then pre-defined. PRelu was employed in the work of He et al. [2015] for ImageNet classification and be claimed as the key factor of performance improvement.

- Randomized Relu Despite of Leaky Relu and PRelu, the slopes of negative part in Randomized Relu (RRelu) are randomized in a given range during the training and then fixed in the testing. For instance, [Xu et al. 2015] use the random α that sampled from $1/U(3, 8)$ in training and use a fixed value $\frac{2}{l+u} = \frac{2}{11}$ in testing. They have evaluated it on the *CIFAR-10*, *CIFAR-100* and *NDS-B* datasets and the results show that RRelu conducted a better performance. It needs to point out that the evaluation on *NDS-B* dataset shows RRelu can overcome over-fitting problem due to

the less training samples compare with *CIFAR-10/100*. A similar conclusion is also reported by [Xu et al. 2015] .

.2.4 Pooling layer

With previous convolution, adding bias and nonlinear activation, it extracts the local features of a training input. It needs to be pointed out that these features have precise positions. Since different training instances with the same label have different precise positions, it maybe harmful for following procedure, e.g. classifying. Therefore, a reasonable method, pooling, is applied in CNN, where it will decrease the feature map's resolution.

To be more precise, a pooling layer can be thought of as consisting of a grid of pooling units spaced s pixels apart, each summarizing a neighborhood of size $z \times z$ centered at the location of the pooling unit. It will reduce the dimensionality of feature map and the computational cost. In addition, it will also makes the model less sensitive to the exact location. In other words, the pooling operation could make the feature map to be translation invariant. Moreover, it summarizes the output of multiple neurons with the essence of taking nearby feature detectors and finally form a local or global bag of features [Boureau et al. 2010]. By the way, it usually sets $s = z$ and if it set $s < z$, it will obtain a overlap pooling, where [Krizhevsky et al. 2012] reported that models with overlapping pooling could benefit avoiding over-fit problem slightly.

Previously, average pooling was applied to the early CNN model [LeCun et al. 1998]. The average pooling (subsampling,downsampling,mean pooling) basically takes the arithmetic mean of the elements in each pooling region.

$$y^i_{m,n} = \frac{1}{s^2} \sum_{\lambda=1}^{s} \sum_{\mu=1}^{s} x^i_{m\times s+\lambda,\ n\times s+\mu} \tag{9}$$

In average pooling method shown in equation 9, $y^i_{m,n}$ stands for the (m, n) element of ith output feature map y^i corresponding to the ith input feature map x^i. It needs to be pointed out that all the elements in

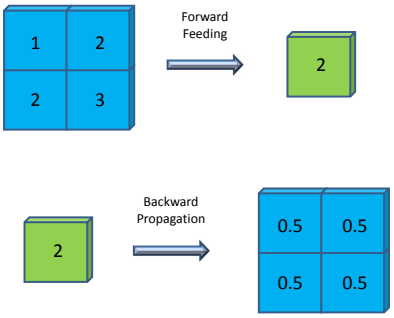
the region are pooled without considering their magnitude. The low or negative activations may downplay a higher activation value resulting in a near zero activation function. Moreover, the strong positive and negative activations may cancel each other's effect, leading to small pooled responses.

To overcome those problems, max pooling methods are applied in recent CNN models [Abdel-Hamid et al. 2014; Mao et al. 2014; Sainath et al. 2015; Cecotti and Graser 2011; Hu et al. 2014; Krizhevsky et al. 2012; Krause et al. 2014; Simonyan and Zisserman 2014a; Zhang et al. 2014d; Weinzaepfel et al. 2013; Girshick et al. 2014; Fan et al. 2010; Farabet et al. 2013; Sun et al. 2014; Taigman et al. 2014]. Max-pooling method, shown in equation 10, selects the largest element from the input region .

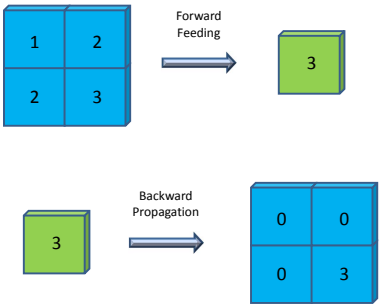
$$y^i_{m,n} = \frac{1}{s^2} \max_{\lambda \in s,\ \mu \in s} x^i_{m*s+\lambda,\ n*s+\mu} \tag{10}$$

However, it holds the nature of disregarding all the other values in the pooling region, making the model to be over-fitted quickly while training and cause negative influence to model's generality. In order to prevent the early over-fitting problem, many regularization methods are applied, including dropout, dropConect and maxout etc. This part will be discussed in the training section later.

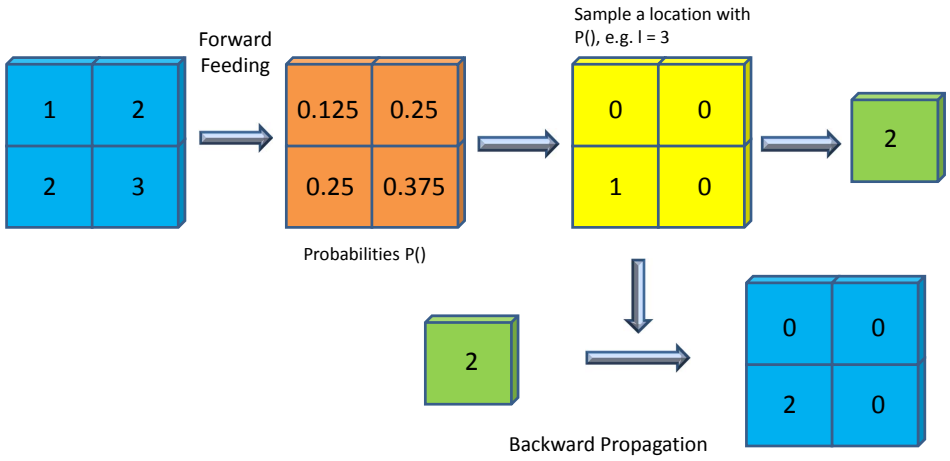
Among averaging pooling and max pooling, stochastic pooling is another popular kind of pooling methods which is firstly applied by [Zeiler and Fergus 2013]. For stochastic pooling, it firstly computed the probability matrix $P()$, then randomly select the location with $P()$. After that, value of select position will be forward to the output, which is similar to the max pooling method. The backward propagation of stochastic pooling is also similar to max pooling too. Generally, stochastic pooling can be seen as a special kind of average pooling method but holds some properties of max pooling. An example of average pooling, max pooling and stochastic pooling is shown in Fig 4.



(a) Average pooling



(b) Max pooling



(c) Stochastic pooling

Figure 4: *An example of pooling methods: (a) average pooling, (b) max pooling and (c)stochastic pooling,where* $s = 2$

.2.5 Normalization layer

The normalization operation is able to give the trained model a better generalization [Krizhevsky et al. 2012; Le 2013]. Local contrast normalization (LCN) is a typical normalization method used in the multi-stage architecture model [Jarrett et al. 2009]. LCN aims to perfrom local subtractive and divisive normalization, enforcing a sort of local competition between adjacent feature s in a feature map , and between features at the same spatial location in different feature maps [Jarrett et al. 2009]. Foe a given ith feature map x^i, the subtractive normalization result $v^i_{m,n}$ is computed as equation 11.

$$v^i_{m,n} = x^i_{m,n} - \sum_{\lambda.\mu} w_{\lambda.\mu} * x^i_{m+\lambda,\ n+\mu} \tag{11}$$

where $w_{\lambda,\mu}$ is a Gaussian weighting window, where $\sum_{\lambda,\mu} w_{\lambda,\mu} = 1$. Thus, the divisive normalization $y^i_{m,n}$ is computed as equation 12,

$$\begin{aligned} y^i_{m,n} &= \frac{v^i_{m,n}}{\max(mean(\sigma_{m,n}),\ \sigma_{m,n})} \\ \sigma_{m,n} &= [\sum_{\lambda,\ \mu} w_{\lambda,\ \mu}(v^i_{m+\lambda,\ n+\mu})^2]^{\frac{1}{2}} \end{aligned} \tag{12}$$

The denominator is the weighted standard deviation of all features over a spatial neighborhood. The local contrast normalization was applied in [Le 2013; Dong et al. 2014].

[Krizhevsky et al. 2012] used another normalization method, called local response normalization (LRN), for generalization purpose in their ImageNet classification work, where the LRN is shown in equation 13

$$y^i_{m,n} = \frac{x^i_{m,n}}{[k + \alpha \sum_{j=\max(0,i-l/2)}^{\min(N-1,i+l/2)} (x^j_{m,n})^2]^\beta} \tag{13}$$

where the summation operates over a l "adjacent" kernel maps at the same spatial position m, n, and N is the total number of kernels in the layer. The kernels' order is determined arbitrarily before training. This operation lets LRN to implement a form of lateral inhibition, which is

inspired by the type found in real neurons. It could create a competition amongst the neuron outputs which is computed using different kernels. the constants k,α,l and β are hyper-parameters that could be determined using a validation set. [Krizhevsky et al. 2012] provided a typical parameter set is $k = 2$, $n = 5$, $\alpha = 10^{-4}$ and $\beta = 0.75$, which is widely used in many recent frameworks such as [Krause et al. 2014; Jin et al. 2014].

.2.6 Full connection layer

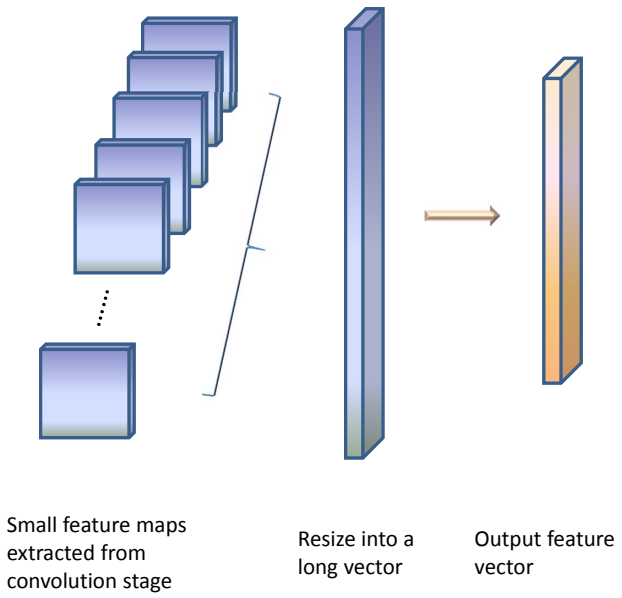


Figure 5: *An example of full connection layer operation*

Generally, previous operations in convolution layer, nonlinear activation, normalization and pooling layer could be defined as a large convolutional stage. In a CNN model, the input of raw data will pass through several convolutional stage and be converted to lots of low-resolution feature maps. These small size feature maps are concatenated into a long vector, where such a vector plays a same role as those manually designed features. Then, the long vector is fed to a one-hidden-layer neural network, which works similar as linear regression. After that, an nonlinear activation function is applied to finally form the output feature map. A

standard full connection layer operation is denoted as equation 14

$$y_j = f_{acti}(\sum_{i=1}^{M} w_{ij}x_i + b),\ j = 1, 2, \ldots, N \tag{14}$$

where y_j is the jth element of N length output feature map of full connection layer, x_i is the ith elment of input long feature vector x whose length is M, $f_{acti}()$ is the nonlinear activation function and (w, b) denote the parameters of the one-hidden-layer neural network. Besides, there could be several full connection layers afterward the convolution stage.

.2.7 Output layer

The output layer is the last layer of a CNN module. It works like the final layer of multi layer perceptron (MLP). It will provide a probability distribution over the output task or the predition value due to its task. It could be a logistic regressor (softmax) or a linear regressor due to the final task. For instance, if it denotes x_i as the ith input to the outputlayer, then the probalitity of the ith class, p^i could be calculated by the following softmax function:

$$p^i = \frac{\exp(x^i)}{\sum_{j=1}^{K} \exp(x^i)} \tag{15}$$

where K is the number of output neurons.

.3 CNN training

.3.1 Architecture selection

In this section, it would introduce how to select the architecture of CNN module for our approaches. First of all, the author would like to annotate the "capacity" of the architectures of a CNN model: the capacity means (1) the depth of CNN model, which is generally stands for the number of repeated convolution stages, and (2) the width of the network, which

is the number of kernels/filters in each convolutional stages. According to learning theory, if the architecture has too much capacity, it tends to overfit the training data and would has poor generalization ability. On other hand, if the architecture has too little capacity, it underfits the training data while both the training error and testing error will be high.

Generally, the model depth depend on the size of input data and the complexity of the task. There are two ways to determine the depth of the model:foward method and backward method

- **Forward methods.** For forward methods, it firstly add only one convolutional stage on the model and evaluate its performance. Then, another stage is added and the model is evaluated again. With the iterative operations, the model become deeper and the test performance is firstly improved then gradually convergence. Once it reaches high enough accuracy, the deeper process would be ended due to avoid over-fitting and efficiency.

- **Backward methods.** For backward methods, it firstly establishes a big enough module that make the train error to be zero. Then, it would reduce the depth of the model or reduce3 the size of each layer. Other techniques such as weight decay regularization and drop out can also be applied [Krizhevsky et al. 2012].

.3.2 Data Augmentation and Pre-Processing

Since deep learning models require large amount of training samples, data augmentation techniques can be sued to boost the performance if the original training data is limited. There are many approaches to do data augmentation, including horizontally flipping, random crops, color jittering, rotation and random scaling. In addition, multiple processing can be combine used

Once the large amount of training samples are collected, tt is necessary to do pre-processing on the training data firstly. Several pre-processing approaches for CNN is introduced here. The first one is zero-centering the data and then normalize them. Another pre-processing approach is

PCA whitening. which transform the original data into eigenbasis and divides every dimension by the eigenvalue to normalize its scale. Note that, a small value, e.g. 10^{-5} is added here to prevent division by zero.

.3.3 BP training and loss function

The backward propagation (BP) algorithm is used to train the deep CNN model. The training process is composed of two steps: it firstly calculates the error/loss via feed forward the training data, then, the erro/loss would be propagated back layer by layer where the gradients would be calculated to update the parameters of each layer, e.g. bias and weight, based on the back prorogated error.

Cross-entropy loss is widely used in many CNN works [Krizhevsky et al. 2012; Zhang et al. 2014d], which has already demonstrated better performance in avoiding learning slowdown than conventional mean square error. In the output layer, the cress-entropy loss function is defined as

$$L = -\sum_{j=1}^{K} [t^j \log(p^j) + (1 - t^j) log(1 - p^j)] \tag{16}$$

where, p^j and K the same definition in equation 15 and t^j is the corresponding one-hot training label. In this case, the loss gradient δ^j is calculated as

$$\begin{aligned} \delta^j = \frac{\partial L^j}{\partial p^j} \cdot \frac{\partial p^j}{\partial x^j} &= (-\frac{t^j}{p^j} + \frac{1 - t^j}{1 - p^j}) \cdot p^j(1 - p^j) \\ &= \frac{p^j - p^j}{p^j(1 - t^j)} \cdot p^j(1 - p^j) \\ &= p^j - t^j \end{aligned} \tag{17}$$

There are many other kinds of loss function that widely used in the CNN model. For instance, Euclidean loss is a common cost function used

in CNN model, which is the mean square error (MSE), defined as:

$$L = \frac{1}{N}\sum_{n=1}^{N}(\hat{Y}_n - Y_n)^2 \tag{18}$$

The Euclidean loss aims to minimized the expectation, which is suitable for learning expected return on a stock. Contrastive loss is used in the Siamese network, which is used to deal with the paired data. It is defined as:

$$L = \frac{1}{2N}\sum_{n=1}^{N} yd^2 + (1-y)\max(margin - d, 0)^2 \tag{19}$$

where $d = \|a_n - b_n\|_2$ is the Euclidean distance of two samples a_n and b_n, y is the matching label for the two samples and $margin$ is the pre-set threshold. For an intended output $t = \pm 1$ and its prediction score y, the Hinge loss is defined as

$$L = max(0, 1 - t * y) \tag{20}$$

.3.4 Initialization and Pre-train

There are many methods to initialize the parameters before training the proposed network, such as random initialization and variance calibration.

Parameter Initialization Currently, the initialization methods are simple and heuristic. It is very difficult to design improved initialization methods, because optimization of deep models has not been totally understood. Parameters in weight layers are composed of weights and biases. Normally, weights are initialized randomly, and biases are set to constants. There are three widely adopted methods, which will be detailed in the following.

- **Random initialization** It is reasonable to assume that half of weights will be positive, half of weights will be negative with proper data normalization aforementioned. However, if all the weights are initialized with zero, it can cause mistakes during BP training since

the same gradients will exact same parameter updating. Therefore, these neurons can be initialized with small random numbers, e.g. $0.001 \times N(0,1)$ where $N(0,1)$ is a zero mean, unit standard deviation Gaussian.

- **Variance calibration** Random normalization can lead a variance growing problem while the number of inputs is increasing. To overcome this problem, each neuron's variance can be normalized that the output is up to 1 by scaling neuron's weight vector by the square root of its fan-in, e.g. $\sqrt{n}$, n is the number of input. Additionally, [He et al. 2015] suggested that the variance of neurons in the network should be $2.0/n$ if the Relu activation function is considered, i.e. $\sqrt{2.0/n}$, which is the recommendation method in practice.

- **Xavier**[Glorot and Bengio 2010] is a kind of normalized initialization. The weights are normalized by the formulation:

$$w \sim (-\sqrt{\frac{3}{hwd_{in}}}, \sqrt{\frac{3}{hwd_{in}}}) \tag{21}$$

  where h and w are the height and the width of kernels respectively. d_{in} is the depth of the input.

- **Improved Xavier**[He et al. 2015] further improves *Xavier* initialization and demonstrates excellent performance in the training of CNNs. The main difference is that *Improved Xavier* considers the influences from nonlinear blocks. The weights are normalized by the formulation:

$$w \sim (-\sqrt{\frac{2}{hwd_{out}}}, \sqrt{\frac{2}{hwd_{out}}}) \tag{22}$$

  where h and w are the height and the width of kernels respectively. d_{out} is the depth of the output.

Pre-train As we know that, it requires large amount of training data to learn a CNN model with good generalization ability. Although

existing work shows that training CNN model with random initialization could achieve competitive performance for visual recognition task, it could suffer the training data limitation problems in practice. An alternative solution to this problem have been provided by the work of researches [Erhan et al. 2010], that is choosing a optimised starting point which can be pre-trained by transferring parameters either supervised or unsupervised, as opposite to random initialized start, which called "pre-train".

The parameter transferring for pre-train could be supported by many phenomenons. For instance, the first layer of many CNN models trained on the natural image learns similar features to Gabor filter and color blobs, where this phenomenon occurs in many frameworks, including supervised image classification [Kapsouras and Nikolaidis 2014], unsupervised density learning [Lee et al. 2009], and unsupervised learning of sparse representations [Le et al. 2011]. These examples shows that the feature extracted from fist layer appear not to be for a particular task/dataset. An intuitive hypothesis is given by [Yosinski et al. 2014] that features must eventually transition from general to specific layers by layers from bottom to top in a deep network, which is the theoretical support for the pre-train via transferring parameters.

Generally, the pre-train could provide two key advantages:

- speed up the CNN model training procedure to reach the convergence point.
- compensate the small volume of available training dataset and make the trained CNN model achieve better generalization performance

For different situations, different fine-tune strategies can be applied for training the pre-trained model, which depends on the similarity between the new data set and the data used for pre-trained model. If the new data set is very similar to the the data used for pre-trained model, it can just train a linear classifier on the features extracted from the top layers of pre-trained models with little amount of data, or fine-tune a few top

layers of pre-trained models with a small learning rate for large amount of data. However, if the new data set is quite different from the data used in pre-trained models, a large number of layers should be fine-tuned with a small learning rate, which requires large amount of available new data.

.3.5 Regularization

The high capacity of CNN model makes it prone to meet the over-fitting problem. Similar to other machine learning models, many regularization methods have been developed for the CNN model, such as dropout [Hinton et al. 2012; Srivastava et al. 2014], drop connect [Wan et al. 2013], stochastic pooling [Zeiler and Fergus 2013] and data augmentation. Following the work of [Krizhevsky et al. 2012], dropout and data augmentation are applied as regularization during CNN model training. Some popular regularization methods are listed below:

- ℓ_2 **regularization** is the most common form of regularization. For every weight w in the CNN, the term $\frac{1}{2}\lambda w^2$ is added to the objective function, where λ is the regularization strength. ℓ_2 regularization can intuitively penalize the peaky weight vectors and prefer diffuse weight vectors.

- ℓ_1 **regularization** is another common form of regularization, where the term $\lambda\|w\|$ is added to the objective function for each weight w. The ℓ_1 regularization leads the weight vectors to become sparse. It means that only a sparse subset of inputs will be used by the neurons that cause a explicit feature selection. Generally, ℓ_2 can be expected to outperformℓ_1 regularization in most cases. Besides, it is possible to combine the ℓ_1 regularization with the ℓ_2 regularization, e.g. $\lambda_1\|w\| + \lambda_2 w^2$.

- **Max norm constraints** will enforce an upper bound on the magnitude of the weight vector. In practice, the parameter of model are updated as normal, then the weight vector is clamped to satisfy the constraint, e.g. $\|w\|_2 < c$, where typical values of c are on

orders of 3 or 4. The max norm constraints is able to deal with the parameter "exploding" problem that caused by high learning rate.

- **Dropout** has been developed by [Hinton et al. 2012; Srivastava et al. 2014] to prevent over-fitting during training large capacity CNN models, where it randomly omits half of the neurons for each training case. It is complemented to the methods mentioned above. The omitting process is done by setting the activation value to zero, which could neglect the neuron temporarily. Thus, the neurons are trained with the random combination of inputs instead of the fixed architecture, where the dropout ratio p is set as 0.5 as a reasonable default. Since it trains multiple random layers at a time, dropout could also be regarded as a from of ensemble learning and has been used in many applications.

In practice, it is common to apply a global ℓ_2 regularization on all the weights w of the whole CNN model rather than implment different regularization on each layer.

Normalization

Nomalization for input data, or known as preprocessing is an essential procedure in many computer vision applications. For instance, images are usually normalized into reasonable ranges such as $[-1, 1]$ or $[0, 1]$. Nomalization layers inside neural networks are essential for improving model's performance on generalization and convergence. Two major normalization layers, named local response normalization (LRN) [Krizhevsky et al. 2012] and batch normalization (BN) are widely used in CNN models

Local Response Normalization The idea of LRN is originally inspired by local contrast normalization (LCN) [Jarrett et al. 2009], which normalize features based on the nearby adjacent in same feature maps or same spatial location in different feature maps by subtraction and division. Comparing to LCN, LRN doe not subtract the mean values,

therefore, the brightness information in LRN is abundant. The response normalization features can be computed via:

$$y_{i,j}^{k} = x_{i,j}^{k} / (\gamma + \alpha \sum_{l=\max(0,k-n/2)}^{\min(N-1,k+n/2)} (x_{i,j}^{l})^2)^{\beta} \tag{23}$$

where $x_{i,j}^{k}$ and $y_{i,j}^{k}$ are input and output feature maps respectively. N is the total depth of input feature maps, and the normalization runs over n adjacent feature maps at the same spatial positions. The constants α, β and γ are hyper-parameters whose values should be determined on a validation test

Batch Normalization Due to the varying distribution of inputs for each layer, the training of CNNs is complicate and sensitive. Generally, low learning rates and careful parameter initialization are required for tackling this problem. This phenomenon is referred as *internal covariate shift* [Ioffe and Szegedy 2015], which has been significantly addressed by BN. Different from LCN and LRN that perform normalization for each individual input, BN normalizing inputs for each training mini-batch. Networks with BN layers can be trained with higher learning rates and less careful initialization, and the networks also demonstrate better generalization ability.

Denoting a mini-batch B of size m, for each spatial location of the inputs, there are m features in the mini-batch: $B = \{x_1, x_2 ... x_k\}$. Let the normalized features be $\hat{x}_{1...m}$ and the linear transformed features be $y_{1...m}$. BN performs the transformation BN : $x_{1...m} \rightarrow \hat{x}_{1...m} \rightarrow y_{1...m}$. More specifically, BN transformation is illustrated as followings:

$$\mu_B = \frac{1}{m}\sum_{i=1}^{m} x_i \tag{24}$$

$$\sigma_B^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_B)^2 \tag{25}$$

$$\hat{x}_i = \frac{x_i - \mu_B}{\sqrt{\sigma_B^2 + \epsilon}} \tag{26}$$

$$y_i = \alpha\hat{x}_i + \beta \tag{27}$$

where μ_B is mini-batch mean and σ_B^2 is mini-batch variance. α and β are scale and shift values of linear transformation for improving feature representation. ϵ is a constant for ensuring numerical stability.

References

O. Abdel-Hamid, A.-R. Mohamed, H. Jiang, L. Deng, G. Penn, and D. Yu. Convolutional neural networks for speech recognition. *IEEE/ACM Transactions on Audio, Speech and Language Processing (TASLP)*, 22(10):1533–1545, 2014.

M. Aharon, M. Elad, and A. Bruckstein. K-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE TRANSACTIONS ON SIGNAL PROCESSING*, 54(11):4311, 2006.

O. Arikan and D. A. Forsyth. Interactive motion generation from examples. In *ACM Transactions on Graphics (TOG)*, volume 21, pages 483–490. ACM, 2002.

M. Balasubramanian and E. L. Schwartz. The isomap algorithm and topological stability. *Science*, 295(5552):7–7, 2002.

M. Barker and W. Rayens. Partial least squares for discrimination. *Journal of chemometrics*, 17(3):166–173, 2003.

M. Barnachon, S. Bouakaz, B. Boufama, and E. Guillou. Ongoing human action recognition with motion capture. *Pattern Recognition*, 47(1): 238–247, 2014.

K. Bashir, T. Xiang, and S. Gong. Gait recognition using gait entropy image. In *Crime Detection and Prevention (ICDP 2009), 3rd International Conference on*, pages 1–6, Dec 2009. doi: 10.1049/ic.2009.0230.

J. Baumann, B. Krüger, A. Zinke, and A. Weber. Data-driven completion of motion capture data. In *VRIPHYS*, pages 111–118, 2011.

P. Beaudoin, P. Poulin, and M. van de Panne. Adapting wavelet com-

pression to human motion capture clips. In *Proceedings of Graphics Interface 2007*, pages 313–318. ACM, 2007.

M. Belkin. Problems of learning on manifolds. 2003.

M. Belkin and P. Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *NIPS*, volume 14, pages 585–591, 2001.

M. Belkin and P. Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural computation*, 15(6):1373–1396, 2003.

Y. Bengio, P. Lamblin, D. Popovici, H. Larochelle, et al. Greedy layer-wise training of deep networks. *Advances in neural information processing systems*, 19:153, 2007.

Y. Bengio, A. Courville, and P. Vincent. Representation learning: A review and new perspectives. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(8):1798–1828, 2013.

M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. Basri. Actions as space-time shapes. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 2, pages 1395–1402. IEEE, 2005.

F. Bogo, A. Kanazawa, C. Lassner, P. Gehler, J. Romero, and M. J. Black. Keep it smpl: Automatic estimation of 3d human pose and shape from a single image. In *European Conference on Computer Vision*, pages 561–578. Springer, 2016.

C. Böhm, S. Berchtold, and D. A. Keim. Searching in high-dimensional spaces: Index structures for improving the performance of multimedia databases. *ACM Computing Surveys (CSUR)*, 33(3):322–373, 2001.

Y.-L. Boureau, J. Ponce, and Y. LeCun. A theoretical analysis of feature pooling in visual recognition. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 111–118, 2010.

S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

M. Brand and A. Hertzmann. Style machines. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 183–192. ACM Press/Addison-Wesley Publishing Co., 2000.

A. M. Bruckstein, D. L. Donoho, and M. Elad. From sparse solutions of systems of equations to sparse modeling of signals and images. *SIAM review*, 51(1):34–81, 2009.

A. Bruderlin and L. Williams. Motion signal processing. In *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 97–104. ACM, 1995.

D. Cai, X. He, X. Wu, and J. Han. Non-negative matrix factorization on manifold. In *2008 Eighth IEEE International Conference on Data Mining*, pages 63–72. IEEE, 2008.

D. Cai, X. He, J. Han, and T. S. Huang. Graph regularized nonnegative matrix factorization for data representation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(8):1548–1560, 2011.

J.-F. Cai, E. J. Candès, and Z. Shen. A singular value thresholding algorithm for matrix completion. *SIAM Journal on Optimization*, 20 (4):1956–1982, 2010.

F. Cao, J. Chen, H. Ye, J. Zhao, and Z. Zhou. Recovering low-rank and sparse matrix based on the truncated nuclear norm. *Neural Networks*, 85:10–20, 2017.

H. Cecotti and A. Graser. Convolutional neural networks for p300 detection with application to brain-computer interfaces. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(3):433–445, 2011.

J. Chai and J. K. Hodgins. Constraint-based motion optimization using a statistical dynamic model. *ACM Transactions on Graphics (TOG)*, 26(3):8, 2007.

M.-W. Chao, C.-H. Lin, J. Assa, and T.-Y. Lee. Human motion retrieval from hand-drawn sketch. *Visualization and Computer Graphics, IEEE Transactions on*, 18(5):729–740, 2012.

R. Chaudhry, F. Ofli, G. Kurillo, R. Bajcsy, and R. Vidal. Bio-inspired

dynamic 3d discriminative skeletal features for human action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 471–478, 2013.

C. Chen, Y. Zhuang, J. Xiao, and Z. Liang. Perceptual 3d pose distance estimation by boosting relational geometric features. *Computer Animation and Virtual Worlds*, 20(2-3):267–277, 2009.

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

S. Chen and Y. Gao. An invariant appearance model for gait recognition. In *Multimedia and Expo, 2007 IEEE International Conference on*, pages 1375–1378, July 2007.

S. S. Chen, D. L. Donoho, and M. A. Saunders. Atomic decomposition by basis pursuit. *SIAM journal on scientific computing*, 20(1):33–61, 1998.

S. S. Chen, D. L. Donoho, and M. A. Saunders. Atomic decomposition by basis pursuit. *SIAM review*, 43(1):129–159, 2001.

M. G. Choi, K. Yang, T. Igarashi, J. Mitani, and J. Lee. Retrieval and visualization of human motion data via stick figures. In *Computer Graphics Forum*, volume 31, pages 2057–2065. Wiley Online Library, 2012.

S. Choi. Algorithms for orthogonal nonnegative matrix factorization. In *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, pages 1828–1832. IEEE, 2008.

K. Choo and D. J. Fleet. People tracking using hybrid monte carlo filtering. In *Computer Vision, 2001. ICCV 2001. Proceedings. Eighth IEEE International Conference on*, volume 2, pages 321–328. IEEE, 2001.

F. R. Chung. *Spectral graph theory*, volume 92. American Mathematical Soc., 1997.

A. Cichocki, R. Zdunek, and S. Amari. Csiszrs divergences for non-negative matrix factorization: Family of new algorithms. In *Independent Component Analysis and Blind Signal Separation, International Conference, Ica 2006, Charleston, Sc, Usa, March 5-8, 2006, Proceedings*, pages 32–39, 2006.

A. Cichocki, S. Cruces, and S. Amari. Generalized alpha-beta divergences and their application to robust nonnegative matrix factorization. *Entropy*, 13(1):134–170, 2011.

P. Climent-Pérez, A. A. Chaaraoui, J. R. Padilla-López, and F. Flórez-Revuelta. Optimal joint selection for skeletal data from rgb-d devices using a genetic algorithm. In *Mexican International Conference on Artificial Intelligence*, pages 163–174. Springer, 2012.

T. Damoulas and M. A. Girolami. Probabilistic multi-class multi-kernel learning: on protein fold recognition and remote homology detection. *Bioinformatics*, 24(10):1264–1270, 2008.

M. Dantone, J. Gall, C. Leistner, and L. Van Gool. Body parts dependent joint regressors for human pose estimation in still images. *IEEE transactions on pattern analysis and machine intelligence*, 36 (11):2131–2143, 2014.

I. Daubechies, M. Defrise, and C. De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Communications on pure and applied mathematics*, 57(11):1413–1457, 2004.

C. Ding, T. Li, W. Peng, and H. Park. Orthogonal nonnegative matrix t-factorizations for clustering. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 126–135. ACM, 2006.

C. H. Ding, X. He, and H. D. Simon. On the equivalence of nonnegative matrix factorization and spectral clustering. In *SDM*, volume 5, pages 606–610. SIAM, 2005.

C. H. Ding, T. Li, and M. I. Jordan. Convex and semi-nonnegative matrix factorizations. *IEEE transactions on pattern analysis and machine intelligence*, 32(1):45–55, 2010.

Z. Ding, P. Wang, P. O. Ogunbona, and W. Li. Investigation of different skeleton features for cnn-based 3d action recognition. *arXiv preprint arXiv:1705.00835*, 2017.

Z. Dong, M. Pei, Y. He, T. Liu, Y. Dong, and Y. Jia. Vehicle type classification using unsupervised convolutional neural network. In *Pattern Recognition (ICPR), 2014 22nd International Conference on*, pages 172–177. IEEE, 2014.

D. L. Donoho, Y. Tsaig, I. Drori, and J.-L. Starck. Sparse solution of underdetermined systems of linear equations by stagewise orthogonal matching pursuit. *Information Theory, IEEE Transactions on*, 58(2): 1094–1121, 2012.

Y. Du, W. Wang, and L. Wang. Hierarchical recurrent neural network for skeleton based action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1110–1118, 2015.

C. Ellis, S. Z. Masood, M. F. Tappen, J. J. LaViola, and R. Sukthankar. Exploring the trade-off between accuracy and observational latency in action recognition. *International Journal of Computer Vision*, 101(3): 420–436, 2013.

D. Erhan, Y. Bengio, A. Courville, P.-A. Manzagol, P. Vincent, and S. Bengio. Why does unsupervised pre-training help deep learning? *The Journal of Machine Learning Research*, 11:625–660, 2010.

G. Evangelidis, G. Singh, and R. Horaud. Skeletal quads: Human action recognition using joint quadruples. In *Pattern Recognition (ICPR), 2014 22nd International Conference on*, pages 4513–4518. IEEE, 2014.

A. Eweiwi, M. S. Cheema, C. Bauckhage, and J. Gall. Efficient pose-based action recognition. In *Asian Conference on Computer Vision*, pages 428–443. Springer, 2014.

J. Fan, W. Xu, Y. Wu, and Y. Gong. Human tracking using convolutional neural networks. *Neural Networks, IEEE Transactions on*, 21(10): 1610–1623, 2010.

C. Farabet, C. Couprie, L. Najman, and Y. LeCun. Learning hierarchical features for scene labeling. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(8):1915–1929, 2013.

Y. Feng, J. Xiao, Y. Zhuang, and X. Liu. Adaptive unsupervised multi-view feature selection for visual concept recognition. In *Computer Vision–ACCV 2012*, pages 343–357. Springer, 2013a.

Y. Feng, J. Xiao, Y. Zhuang, and X. Liu. Adaptive unsupervised multi-view feature selection for visual concept recognition. In *Computer Vision–ACCV 2012*, pages 343–357. Springer, 2013b.

Y. Feng, M. Ji, J. Xiao, X. Yang, J. J. Zhang, Y. Zhuang, and X. Li. Mining spatial-temporal patterns and structural sparsity for human motion data denoising. 2014a.

Y. Feng, J. Xiao, Y. Zhuang, X. Yang, J. J. Zhang, and R. Song. Exploiting temporal stability and low-rank structure for motion capture data refinement. *Information Sciences*, 277:777–793, 2014b.

Y. Feng, J. Xiao, K. Zhou, and Y. Zhuang. A locally weighted sparse graph regularized non-negative matrix factorization method. *Neurocomputing*, 169:68–76, 2015.

K. Forbes and E. Fiume. An efficient search algorithm for motion data using weighted pca. In *Proceedings of the 2005 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 67–76. ACM, 2005.

K. Fragkiadaki, S. Levine, P. Felsen, and J. Malik. Recurrent network models for human dynamics. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4346–4354, 2015.

Z. Gao, J.-m. Song, H. Zhang, A.-A. Liu, Y.-b. Xue, and G.-p. Xu. Human action recognition via multi-modality information. *Journal of Electrical Engineering & Technology*, 9(2):739–748, 2014.

D. Garcia. Robust smoothing of gridded data in one and higher dimensions with missing values. *Computational statistics & data analysis*, 54(4):1167–1178, 2010.

R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

M. Gleicher. Motion path editing. In *Proceedings of the 2001 symposium on Interactive 3D graphics*, pages 195–202. ACM, 2001.

X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, pages 249–256, 2010.

M. Goffredo, J. Carter, and M. Nixon. Front-view gait recognition. In *Biometrics: Theory, Applications and Systems, 2008. BTAS 2008. 2nd IEEE International Conference on*, pages 1–6, Sept 2008.

I. F. Gorodnitsky and B. D. Rao. Sparse signal reconstruction from limited data using focuss: A re-weighted minimum norm algorithm. *Signal Processing, IEEE Transactions on*, 45(3):600–616, 1997.

M. A. Gowayyed, M. Torki, M. E. Hussein, and M. El-Saban. Histogram of oriented displacements (hod): describing trajectories of human joints for action recognition. In *Proceedings of the Twenty-Third international joint conference on Artificial Intelligence*, pages 1351–1357. AAAI Press, 2013.

R. Gross and J. Shi. The cmu motion of body (mobo) database. 2001.

Q. Gu and J. Zhou. Neighborhood preserving nonnegative matrix factorization. In *BMVC*, pages 1–10, 2009.

M. Guay, M.-P. Cani, and R. Ronfard. The line of action: An intuitive interface for expressive character posing. *ACM Transactions on Graphics (TOG)*, 32(6):205, 2013.

E. T. Hale, W. Yin, and Y. Zhang. Fixed-point continuation for ℓ_1-

minimization: Methodology and convergence. *SIAM Journal on Optimization*, 19(3):1107–1130, 2008.

D. K. Hammond, P. Vandergheynst, and R. Gribonval. Wavelets on graphs via spectral graph theory. *Applied and Computational Harmonic Analysis*, 30(2):129–150, 2011.

Y. Han, F. Wu, D. Tao, J. Shao, Y. Zhuang, and J. Jiang. Sparse unsupervised dimensionality reduction for multiple view data. *Circuits and Systems for Video Technology, IEEE Transactions on*, 22(10): 1485–1496, 2012.

Y. Han, Y. Yang, Y. Yan, Z. Ma, N. Sebe, and X. Zhou. Semisupervised feature selection via spline regression for video semantic recognition. *Neural Networks and Learning Systems, IEEE Transactions on*, pp (99):1, 2014.

K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

X. He, D. Cai, and P. Niyogi. Laplacian score for feature selection. In *Advances in neural information processing systems*, pages 507–514, 2005.

G. Hinton, S. Osindero, and Y.-W. Teh. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

G. E. Hinton and R. R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

S. Hochreiter, Y. Bengio, P. Frasconi, and J. Schmidhuber. Gradient flow in recurrent nets: the difficulty of learning long-term dependencies, 2001.

D. Holden, J. Saito, T. Komura, and T. Joyce. Learning motion man-

ifolds with convolutional autoencoders. In *SIGGRAPH Asia 2015 Technical Briefs*, page 18. ACM, 2015.

D. Holden, J. Saito, and T. Komura. A deep learning framework for character motion synthesis and editing. *ACM Transactions on Graphics (TOG)*, 35(4):138, 2016.

B. Hong, L. Wei, Y. Hu, D. Cai, and X. He. Online robust principal component analysis via truncated nuclear norm regularization. *Neurocomputing*, 175:216–222, 2016.

Y. Hou, Z. Li, P. Wang, and W. Li. Skeleton optical spectra based action recognition using convolutional neural networks. *IEEE Transactions on Circuits and Systems for Video Technology*, 2016.

P. O. Hoyer. Non-negative sparse coding. In *Neural Networks for Signal Processing, 2002. Proceedings of the 2002 12th IEEE Workshop on*, pages 557–565. IEEE, 2002.

P. O. Hoyer. Non-negative matrix factorization with sparseness constraints. *Journal of machine learning research*, 5(Nov):1457–1469, 2004.

C.-C. Hsieh and P.-L. Kuo. An impulsive noise reduction agent for rigid body motion data using b-spline wavelets. *Expert Systems with Applications*, 34(3):1733–1741, 2008.

E. Hsu, K. Pulli, and J. Popović. Style translation for human motion. In *ACM Transactions on Graphics (TOG)*, volume 24, pages 1082–1089. ACM, 2005.

B. Hu, Z. Lu, H. Li, and Q. Chen. Convolutional neural network architectures for matching natural language sentences. In *Advances in Neural Information Processing Systems*, pages 2042–2050, 2014.

M. Hu, Y. Wang, Z. Zhang, D. Zhang, and J. Little. Incremental learning for video-based gait recognition with lbp flow. *Cybernetics, IEEE Transactions on*, 43(1):77–89, Feb 2013a. ISSN 2168-2267. doi: 10.1109/TSMCB.2012.2199310.

W. Hu, Z. Wang, S. Liu, X. Yang, G. Yu, and J. J. Zhang. Motion

capture data completion via truncated nuclear norm regularization. *IEEE Signal Processing Letters*, PP(99):1–1, 2017a.

W. Hu, Z. Wang, X. Yang, and J. J. Zhang. *An Efficient Bilinear Factorization based Method For Motion Capture Data Refinement: Selected Papers from CSMA2016*. 2017b.

Y. Hu, D. Zhang, J. Ye, X. Li, and X. He. Fast and accurate matrix completion via truncated nuclear norm regularization. *IEEE transactions on pattern analysis and machine intelligence*, 35(9):2117–2130, 2013b.

Y. Hu, Z. Jin, Y. Shi, D. Zhang, D. Cai, and X. He. Large scale multiclass classification with truncated nuclear norm regularization. *Neurocomputing*, 148:310–317, 2015.

C.-P. Huang, C.-H. Hsieh, K.-T. Lai, and W.-Y. Huang. Human action recognition using histogram of oriented gradient of motion history image. In *Instrumentation, Measurement, Computer, Communication and Control, 2011 First International Conference on*, pages 353–356, Oct 2011.

P. Huang, M. Tejera, J. Collomosse, and A. Hilton. Hybrid skeletal-surface motion graphs for character animation from 4d performance capture. *ACM Transactions on Graphics (TOG)*, 34(2):17, 2015.

Y. Huang and M. Kallmann. Planning motions and placements for virtual demonstrators. *IEEE transactions on visualization and computer graphics*, 22(5):1568–1579, 2016.

Z. Huang, C. Wan, T. Probst, and L. Van Gool. Deep learning on lie groups for skeleton-based action recognition. In *Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6099–6108. IEEE computer Society, 2017.

M. E. Hussein, M. Torki, M. A. Gowayyed, and M. El-Saban. Human action recognition using a temporal hierarchy of covariance descriptors on 3d joint locations. In *IJCAI*, volume 13, pages 2466–2472, 2013.

E. P. Ijjina and C. K. Mohan. Human action recognition based on mocap

information using convolution neural networks. In *Machine Learning and Applications (ICMLA), 2014 13th International Conference on*, pages 159–164. IEEE, 2014.

L. Ikemoto and D. A. Forsyth. Enriching a motion collection by transplanting limbs. In *Proceedings of the 2004 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 99–108. Eurographics Association, 2004.

L. Ikemoto, O. Arikan, and D. Forsyth. Generalizing motion edits with gaussian processes. *ACM Transactions on Graphics (TOG)*, 28(1):1, 2009.

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

C. Ionescu, D. Papava, V. Olaru, and C. Sminchisescu. Human3. 6m: Large scale datasets and predictive methods for 3d human sensing in natural environments. *IEEE transactions on pattern analysis and machine intelligence*, 36(7):1325–1339, 2014.

K. Jarrett, K. Kavukcuoglu, Y. LeCun, et al. What is the best multi-stage architecture for object recognition? In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 2146–2153. IEEE, 2009.

M. Jeevan, N. Jain, M. Hanmandlu, and G. Chetty. Gait recognition based on gait pal and pal entropy image. In *Image Processing (ICIP), 2013 20th IEEE International Conference on*, pages 4195–4199, Sept 2013. doi: 10.1109/ICIP.2013.6738864.

A. D. Jepson, D. J. Fleet, and T. F. El-Maraghi. Robust online appearance models for visual tracking. *IEEE transactions on pattern analysis and machine intelligence*, 25(10):1296–1311, 2003.

Y. Jia, F. Nie, and C. Zhang. Trace ratio problem revisited. *Neural Networks, IEEE Transactions on*, 20(4):729–735, 2009.

J. Jin, K. Fu, and C. Zhang. Traffic sign recognition with hinge loss

trained convolutional neural networks. *IEEE Transactions on Intelligent Transportation Systems*, 15(5), 2014.

Z. Kang, C. Peng, and Q. Cheng. Robust subspace clustering via smoothed rank approximation. *IEEE Signal Processing Letters*, 22(11): 2088–2092, 2015.

I. Kapsouras and N. Nikolaidis. Action recognition on motion capture data using a dynemes and forward differences representation. *Journal of Visual Communication and Image Representation*, 25(6):1432–1445, 2014.

Q. Ke, S. An, M. Bennamoun, F. Sohel, and F. Boussaid. Skeletonnet: Mining deep part features for 3-d action recognition. *IEEE Signal Processing Letters*, 24(6):731–735, 2017a.

Q. Ke, M. Bennamoun, S. An, F. Sohel, and F. Boussaid. A new representation of skeleton sequences for 3d action recognition. *arXiv preprint arXiv:1703.03492*, 2017b.

E. Keogh, T. Palpanas, V. B. Zordan, D. Gunopulos, and M. Cardle. Indexing large human-motion databases. In *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*, pages 780–791. VLDB Endowment, 2004.

T. Kerola, N. Inoue, and K. Shinoda. Spectral graph skeletons for 3d action recognition. In *Asian Conference on Computer Vision*, pages 417–432. Springer, 2014.

H. Kim and H. Park. Sparse non-negative matrix factorizations via alternating non-negativity-constrained least squares for microarray data analysis. *Bioinformatics*, 23(12):1495–1502, 2007.

T. S. Kim and A. Reiter. Interpretable 3d human action analysis with temporal convolutional networks. *arXiv preprint arXiv:1704.04516*, 2017.

Y. Kim, H. Park, S. Bang, and S.-H. Lee. Retargeting human-object interaction to virtual avatars. *IEEE transactions on visualization and computer graphics*, 22(11):2405–2412, 2016.

Y. Kong and Y. Jia. A hierarchical model for human interaction recognition. In *Multimedia and Expo (ICME), 2012 IEEE International Conference on*, pages 1–6. IEEE, 2012.

Y. Kong, Y. Jia, and Y. Fu. Learning human interaction by interactive phrases. *Computer Vision–ECCV 2012*, pages 300–313, 2012.

Y. Kong, Y. Jia, and Y. Fu. Interactive phrases: Semantic descriptionsfor human interaction recognition. *IEEE transactions on pattern analysis and machine intelligence*, 36(9):1775–1788, 2014.

L. Kovar and M. Gleicher. Automated extraction and parameterization of motions in large data sets. In *ACM Transactions on Graphics (TOG)*, volume 23, pages 559–568. ACM, 2004.

L. Kovar, M. Gleicher, and F. Pighin. Motion graphs. In *ACM transactions on graphics (TOG)*, volume 21, pages 473–482. ACM, 2002.

J. Krause, T. Gebru, J. Deng, L.-J. Li, and L. Fei-Fei. Learning features and parts for fine-grained recognition. In *Pattern Recognition (ICPR), 2014 22nd International Conference on*, pages 26–33. IEEE, 2014.

A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

B. Krüger, J. Tautges, A. Weber, and A. Zinke. Fast local and global similarity searches in large motion capture databases. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 1–10. Eurographics Association, 2010.

W. Kusakunniran. Attribute-based learning for gait recognition using spatio-temporal interest points. *Image and Vision Computing*, 32(12): 1117 – 1126, 2014. ISSN 0262-8856. doi: http://dx.doi.org/10.1016/j.imavis.2014.10.004.

W. Kusakunniran, Q. Wu, J. Zhang, and H. Li. Pairwise shape configuration-based psa for gait recognition under small viewing angle change. In *Advanced Video and Signal-Based Surveillance (AVSS), 2011 8th IEEE International Conference on*, pages 17–22, Aug 2011.

R. Lai, P. C. Yuen, and K. Lee. Motion capture data completion and denoising by singular value thresholding. *Proceedings of Eurographics, Eurographics Association*, pages 45–48, 2011.

M. Lau, Z. Bar-Joseph, and J. Kuffner. Modeling spatial and temporal variation in motion data. In *ACM Transactions on Graphics (TOG)*, volume 28, page 171. ACM, 2009.

Q. V. Le. Building high-level features using large scale unsupervised learning. In *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on*, pages 8595–8598. IEEE, 2013.

Q. V. Le, A. Karpenko, J. Ngiam, and A. Y. Ng. Ica with reconstruction cost for efficient overcomplete feature learning. In *Advances in Neural Information Processing Systems*, pages 1017–1025, 2011.

C. Lea, M. D. Flynn, R. Vidal, A. Reiter, and G. D. Hager. Temporal convolutional networks for action segmentation and detection. *arXiv preprint arXiv:1611.05267*, 2016.

Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11): 2278–2324, 1998.

C. Lee and E. Lam. Computationally efficient truncated nuclear norm minimization for high dynamic range imaging. *IEEE Transcations on Image Processing*, 25(9):4145–4157, 2016.

C. Lee, A. W. C. Tan, and S. C. Tan. Time-sliced averaged motion history image for gait recognition. *J. Visual Communication and Image Representation*, 25(5):822–826, 2014.

C. P. Lee, A. W. Tan, and S. C. Tan. Gait recognition via optimally interpolated deformable contours. *Pattern Recognition Letters*, 34(6): 663–669, 2013.

D. Lee and H. S. Seung. Learning the parts of objects by non-negative matrix factorization. *Nature*, 401(6755):788–791, 1999.

D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In *Neural Information Processing Systems*, pages 556–562, 2001.

H. Lee, A. Battle, R. Raina, and A. Y. Ng. Efficient sparse coding algorithms. In *Advances in neural information processing systems*, pages 801–808, 2006.

H. Lee, C. Ekanadham, and A. Y. Ng. Sparse deep belief net model for visual area v2. In *Advances in neural information processing systems*, pages 873–880, 2008a.

H. Lee, S. Hong, and E. Kim. An efficient gait recognition based on a selective neural network ensemble. *International Journal of Imaging Systems and Technology*, 18(4):237–241, 2008b. ISSN 1098-1098.

H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 609–616. ACM, 2009.

I. Lee, D. Kim, S. Kang, and S. Lee. Ensemble deep learning for skeleton-based action recognition using temporal sliding lstm networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1012–1020, 2017.

A. M. Lehrmann, P. V. Gehler, and S. Nowozin. Efficient nonlinear markov models for human motion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1314–1321, 2014.

B. Li, M. Ayazoglu, T. Mao, O. I. Camps, and M. Sznaier. Activity recognition using dynamic subspace angles. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 3193–3200. IEEE, 2011.

B. Li, O. I. Camps, and M. Sznaier. Cross-view activity recognition using hankelets. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 1362–1369. IEEE, 2012.

C. Li, Y. Hou, P. Wang, and W. Li. Joint distance maps based action recognition with convolutional neural networks. *IEEE Signal Processing Letters*, 24(5):624–628, 2017.

L. Li and Y. Zhang. Sensc: A stable and efficient algorithm for non-negative sparse coding. *Acta Automatica Sinica*, 35(10):1257–1271, 2009.

L. Li, J. McCann, N. S. Pollard, and C. Faloutsos. Dynammo: Mining and summarization of coevolving sequences with missing values. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 507–516. ACM, 2009.

L. Li, J. McCann, N. Pollard, and C. Faloutsos. Bolero: a principled technique for including bone length constraints in motion capture occlusion filling. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 179–188. Eurographics Association, 2010a.

L.-X. Li, L. Wu, H.-S. Zhang, and F.-X. Wu. A fast algorithm for non-negative matrix factorization and its convergence. *IEEE transactions on neural networks and learning systems*, 25(10):1855–1863, 2014.

S. Z. Li, X. W. Hou, H. J. Zhang, and Q. S. Cheng. Learning spatially localized, parts-based representation. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–207. IEEE, 2001.

T. Li and C. Ding. The relationships among various nonnegative matrix factorization methods for clustering. In *Sixth International Conference on Data Mining (ICDM'06)*, pages 362–371. IEEE, 2006.

W. Li, Z. Zhang, and Z. Liu. Action recognition based on a bag of 3d points. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*, pages 9–14. IEEE, 2010b.

W. Li, Z. Zhang, and Z. Liu. Action recognition based on a bag of 3d points. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2010 IEEE Computer Society Conference on*, pages 9–14. IEEE, 2010c.

Y. Li, C. Lan, J. Xing, W. Zeng, C. Yuan, and J. Liu. Online human

action detection using joint classification-regression recurrent neural networks. In *European Conference on Computer Vision*, pages 203–220. Springer, 2016.

Z. Li, X. Wu, and H. Peng. Nonnegative matrix factorization on orthogonal subspace. *Pattern Recognition Letters*, 31(9):905–911, 2010d.

I. Lillo, A. Soto, and J. Carlos Niebles. Discriminative hierarchical modeling of spatio-temporally composable human activities. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 812–819, 2014.

Z. Lin, M. Chen, and Y. Ma. The augmented lagrange multiplier method for exact recovery of corrupted low-rank matrices. *arXiv preprint arXiv:1009.5055*, 2010.

G. Liu and L. McMillan. Segment-based human motion compression. In *Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 127–135. Eurographics Association, 2006.

J. Liu, J. Luo, and M. Shah. Recognizing realistic actions from videos in the wild. In *Computer vision and pattern recognition, 2009. CVPR 2009. IEEE conference on*, pages 1996–2003. IEEE, 2009.

J. Liu, A. Shahroudy, D. Xu, and G. Wang. Spatio-temporal lstm with trust gates for 3d human action recognition. In *European Conference on Computer Vision*, pages 816–833. Springer, 2016a.

J. Liu, G. Wang, P. Hu, L.-Y. Duan, and A. C. Kot. Global context-aware attention lstm networks for 3d action recognition. In *Proc. Comput. Vis. Pattern Recognit.*, pages 1647–1656, 2017a.

M. Liu, H. Liu, and C. Chen. Enhanced skeleton visualization for view invariant human action recognition. *Pattern Recognition*, 68:346–362, 2017b.

Q. Liu, Z. Lai, Z. Zhou, F. Kuang, and Z. Jin. A truncated nuclear norm regularization method based on weighted residual error for matrix

completion. *IEEE Transactions on Image Processing*, 25(1):316–330, 2016b.

M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black. Smpl: A skinned multi-person linear model. *ACM Transactions on Graphics (TOG)*, 34(6):248, 2015.

H. Lou and J. Chai. Example-based human motion denoising. *Visualization and Computer Graphics, IEEE Transactions on*, 16(5):870–879, 2010.

W. Ma, S. Xia, J. K. Hodgins, X. Yang, C. Li, and Z. Wang. Modeling style and variation in human motion. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, pages 21–30. Eurographics Association, 2010.

Z. Ma, F. Nie, Y. Yang, J. R. Uijlings, N. Sebe, and A. G. Hauptmann. Discriminating joint feature analysis for multimedia data understanding. *Multimedia, IEEE Transactions on*, 14(6):1662–1672, 2012.

Y. Makihara, H. Mannami, A. Tsuji, M. A. Hossain, K. Sugiura, A. Mori, and Y. Yagi. The ou-isir gait database comprising the treadmill dataset. *IPSJ Transactions on Computer Vision and Applications*, 4:53–62, 2012.

Q. Mao, M. Dong, Z. Huang, and Y. Zhan. Learning salient features for speech emotion recognition using convolutional neural networks. *Multimedia, IEEE Transactions on*, 16(8), 2014.

Y. Mao and L. K. Saul. Modeling distances in large-scale networks by matrix factorization. In *Proceedings of the 4th ACM SIGCOMM conference on Internet measurement*, pages 278–287. ACM, 2004.

M. Marszalek, I. Laptev, and C. Schmid. Actions in context. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2929–2936. IEEE, 2009.

C. Meek, D. M. Chickering, and D. Heckerman. Autoregressive tree models for time-series analysis. In *Proceedings of the 2002 SIAM International Conference on Data Mining*, pages 229–244. SIAM, 2002.

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

J. Min, H. Liu, and J. Chai. Synthesis and editing of personalized stylistic human motion. In *Proceedings of the 2010 ACM SIGGRAPH symposium on Interactive 3D Graphics and Games*, pages 39–46. ACM, 2010.

R. Mittelman, B. Kuipers, S. Savarese, and H. Lee. Structured recurrent temporal restricted boltzmann machines. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 1647–1655, 2014.

M. Morup, K. H. Madsen, and L. K. Hansen. Approximate l 0 constrained non-negative matrix and tensor factorization. In *2008 IEEE International Symposium on Circuits and Systems*, pages 1328–1331. IEEE, 2008.

T. Mukai and S. Kuriyama. Geostatistical motion interpolation. In *ACM Transactions on Graphics (TOG)*, volume 24, pages 1062–1070. ACM, 2005.

M. Müller and T. Röder. Motion templates for automatic classification and retrieval of motion capture data. In *Proceedings of the 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation*, pages 137–146. Eurographics Association, 2006.

M. Müller, T. Röder, and M. Clausen. Efficient content-based retrieval of motion capture data. In *ACM Transactions on Graphics (TOG)*, volume 24, pages 677–685. ACM, 2005.

M. Müller, T. Röder, M. Clausen, B. Eberhardt, B. Krüger, and A. Weber. Documentation mocap database hdm05. 2007a.

M. Müller, T. Röder, M. Clausen, B. Eberhardt, B. Krüger, and A. Weber. Documentation mocap database hdm05. 2007b.

M. Müller, A. Baak, and H.-P. Seidel. Efficient and robust annotation of motion capture data. In *Proceedings of the 2009 ACM SIG-*

GRAPH/Eurographics Symposium on Computer Animation, pages 17–26. ACM, 2009.

B. Ni, G. Wang, and P. Moulin. Rgbd-hudaact: A color-depth video database for human daily activity recognition. In *Consumer Depth Cameras for Computer Vision*, pages 193–208. Springer, 2013.

F. Ofli, R. Chaudhry, G. Kurillo, R. Vidal, and R. Bajcsy. Sequence of the most informative joints (smij): A new representation for human skeletal action recognition. In *CVPR Workshops*, pages 8–13. IEEE, 2012.

F. Ofli, R. Chaudhry, G. Kurillo, R. Vidal, and R. Bajcsy. Berkeley mhad: A comprehensive multimodal human action database. In *Applications of Computer Vision (WACV), 2013 IEEE Workshop on*, pages 53–60. IEEE, 2013.

T.-H. Oh, H. Kim, Y.-W. Tai, J.-C. Bazin, and I. So Kweon. Partial sum minimization of singular values in rpca for low-level vision. In *Proceedings of the IEEE International Conference on Computer Vision (CVPR)*, pages 145–152, 2013a.

T.-H. Oh, J.-Y. Lee, and I. S. Kweon. High dynamic range imaging by a rank-1 constraint. In *Proceedings of the IEEE International Conference on Image Processing*, pages 790–794. IEEE, 2013b.

T.-H. Oh, Y.-W. Tai, J.-C. Bazin, H. Kim, and I. S. Kweon. Partial sum minimization of singular values in robust pca: Algorithm and applications. *IEEE transactions on pattern analysis and machine intelligence*, 38(4):744–758, 2016.

P. Paatero. Least squares formulation of robust non-negative factor analysis. *Chemometrics and intelligent laboratory systems*, 37(1):23–35, 1997.

P. Paatero and U. Tapper. Positive matrix factorization: A non-negative factor model with optimal utilization of error estimates of data values. *Environmetrics*, 5(2):111–126, 1994.

A. Parekh and I. W. Selesnick. Enhanced low-rank matrix approximation. *IEEE Signal Processing Letters*, 23(4):493–497, 2016.

A. Patron-Perez, M. Marszalek, A. Zisserman, and I. D. Reid. High five: Recognising human interactions in tv shows. In *BMVC*, volume 1, page 2. Citeseer, 2010.

A. Patron-Perez, M. Marszalek, I. Reid, and A. Zisserman. Structured learning of human interactions in tv shows. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(12):2441–2453, 2012.

S.-J. Peng, G.-F. He, X. Liu, and H.-Z. Wang. Hierarchical block-based incomplete human mocap data recovery using adaptive nonnegative matrix factorization. *Computers & Graphics*, 49:10–23, 2015.

R. Poppe. A survey on vision-based human action recognition. *Image and vision computing*, 28(6):976–990, 2010.

C. Poultney, S. Chopra, Y. L. Cun, et al. Efficient learning of sparse representations with an energy-based model. In *Advances in neural information processing systems*, pages 1137–1144, 2006.

K. Pullen and C. Bregler. Animating by multi-level sampling. In *Computer Animation 2000. Proceedings*, pages 36–42. IEEE, 2000.

T. Qi, Y. Feng, J. Xiao, Y. Zhuang, X. Yang, and J. Zhang. A semantic feature for human motion retrieval. *Computer animation and virtual worlds*, 24(3-4):399–407, 2013.

T. Qi, Y. Feng, J. Xiao, H. Zhang, Y. Zhuang, X. Yang, and J. Zhang. A human motion feature based on semi-supervised learning of gmm. *Multimedia Systems*, pages 1–9, 2014a.

T. Qi, J. Xiao, Y. Zhuang, H. Zhang, X. Yang, J. Zhang, and Y. Feng. Real-time motion data annotation via action string. *Computer Animation and Virtual Worlds*, 25(3-4):293–302, 2014b.

R. Qian, B. Zhang, Y. Yue, Z. Wang, and F. Coenen. Robust chinese traffic sign detection and recognition with deep convolutional neural network. In *Natural Computation (ICNC), 2015 11th International Conference on*, pages 791–796. IEEE, 2015.

N. Raman and S. J. Maybank. Action classification using a discriminative multilevel hdp-hmm. *Neurocomputing*, 154:149–161, 2015.

M. Raptis, D. Kirovski, and H. Hoppe. Real-time classification of dance gestures from skeleton animation. In *Proceedings of the 2011 ACM SIGGRAPH/Eurographics symposium on computer animation*, pages 147–156. ACM, 2011.

R. Rosipal and L. J. Trejo. Kernel partial least squares regression in reproducing kernel hilbert space. *Journal of machine learning research*, 2(Dec):97–123, 2001.

S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

R. Rubinstein, M. Zibulevsky, and M. Elad. Efficient implementation of the k-svd algorithm using batch orthogonal matching pursuit. *CS Technion*, 40(8):1–15, 2008.

T. N. Sainath, B. Kingsbury, G. Saon, H. Soltau, A.-r. Mohamed, G. Dahl, and B. Ramabhadran. Deep convolutional neural networks for large-scale speech tasks. *Neural Networks*, 64:39–48, 2015.

H. Sakoe and S. Chiba. Dynamic programming algorithm optimization for spoken word recognition. *IEEE transactions on acoustics, speech, and signal processing*, 26(1):43–49, 1978.

S. Sarkar, P. Phillips, Z. Liu, I. Vega, P. Grother, and K. Bowyer. The humanid gait challenge problem: data sets, performance, and analysis. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 27 (2):162–177, Feb 2005. ISSN 0162-8828. doi: 10.1109/TPAMI.2005.39.

C. Schuldt, I. Laptev, and B. Caputo. Recognizing human actions: A local svm approach. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 3, pages 32–36. IEEE, 2004.

L. Seidenari, V. Varano, S. Berretti, A. Bimbo, and P. Pala. Recognizing actions from depth cameras as weakly aligned multi-part bag-of-poses.

In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 479–485, 2013.

A. Shahroudy, J. Liu, T.-T. Ng, and G. Wang. Ntu rgb+ d: A large scale dataset for 3d human activity analysis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1010–1019, 2016.

S. Shaikh, K. Saeed, and N. Chaki. Gait recognition using partial silhouette-based approach. In *Signal Processing and Integrated Networks (SPIN), 2014 International Conference on*, pages 101–106, Feb 2014.

Z. Shao and Y. Li. Integral invariants for space motion trajectory matching and recognition. *Pattern Recognition*, 48(8):2418–2432, 2015.

B. Shen and L. Si. Non-negative matrix factorization clustering on multiple manifolds. In *AAAI*, pages 575–580, 2010.

J. Shotton, T. Sharp, A. Kipman, A. Fitzgibbon, M. Finocchio, A. Blake, M. Cook, and R. Moore. Real-time human pose recognition in parts from single depth images. *Communications of the ACM*, 56(1):116–124, 2013a.

J. Shotton, T. Sharp, A. Kipman, A. Fitzgibbon, M. Finocchio, A. Blake, M. Cook, and R. Moore. Real-time human pose recognition in parts from single depth images. *Communications of the ACM*, 56(1):116–124, 2013b.

X. Shu, F. Porikli, and N. Ahuja. Robust orthonormal subspace learning: Efficient recovery of corrupted low-rank matrices. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3874–3881, 2014.

H. P. Shum, E. S. Ho, Y. Jiang, and S. Takagi. Real-time posture reconstruction for microsoft kinect. *IEEE Transactions on Cybernetics*, 43 (5):1357–1369, 2013.

K. Simonyan and A. Zisserman. Two-stream convolutional networks

for action recognition in videos. In *Advances in Neural Information Processing Systems*, pages 568–576, 2014a.

K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014b.

R. Slama, H. Wannous, M. Daoudi, and A. Srivastava. Accurate 3d action recognition using learning on the grassmann manifold. *Pattern Recognition*, 48(2):556–567, 2015.

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

Y. Sun, X. Wang, and X. Tang. Deep learning face representation from predicting 10,000 classes. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 1891–1898. IEEE, 2014.

Y. Taigman, M. Yang, M. Ranzato, and L. Wolf. Deepface: Closing the gap to human-level performance in face verification. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 1701–1708. IEEE, 2014.

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

J. K. Tang and H. Leung. Retrieval of logically relevant 3d human motions by adaptive feature selection with graded relevance feedback. *Pattern Recognition Letters*, 33(4):420–430, 2012.

J. K. Tang, H. Leung, T. Komura, and H. P. Shum. Emulating human perception of motion similarity. *Computer Animation and Virtual Worlds*, 19(3-4):211–221, 2008.

T. Tangkuampien and D. Suter. Human motion de-noising via greedy kernel principal component analysis filtering. In *Pattern Recognition, 2006. ICPR 2006. 18th International Conference on*, volume 3, pages 457–460. IEEE, 2006.

G. W. Taylor and G. E. Hinton. Factored conditional restricted boltzmann machines for modeling motion style. In *Proceedings of the 26th annual international conference on machine learning*, pages 1025–1032. ACM, 2009.

G. W. Taylor, G. E. Hinton, and S. T. Roweis. Two distributed-state models for generating high-dimensional time series. *Journal of Machine Learning Research*, 12(Mar):1025–1068, 2011.

H.-Y. Tung, H.-W. Tung, E. Yumer, and K. Fragkiadaki. Self-supervised learning of motion capture. In *Advances in Neural Information Processing Systems*, pages 5240–5250, 2017.

S. Ullman et al. *High-level vision: Object recognition and visual cognition*, volume 2. MIT press Cambridge, MA, 1996.

R. Urtasun, D. Fleet, and P. Fua. Gaussian process dynamical models for 3d people tracking. In *Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2006.

M. H. Van Benthem and M. R. Keenan. Fast algorithm for the solution of large-scale non-negativity-constrained least squares problems. *Journal of chemometrics*, 18(10):441–450, 2004.

V. Veeriah, N. Zhuang, and G.-J. Qi. Differential recurrent neural networks for action recognition. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4041–4049, 2015.

R. Vemulapalli, F. Arrate, and R. Chellappa. Human action recognition by representing 3d skeletons as points in a lie group. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 588–595, 2014.

T. von Marcard, G. Pons-Moll, and B. Rosenhahn. Human pose estimation from video and imus. *IEEE transactions on pattern analysis and machine intelligence*, 38(8):1533–1547, 2016.

L. Wan, M. Zeiler, S. Zhang, Y. L. Cun, and R. Fergus. Regularization of neural networks using dropconnect. In *Proceedings of the 30th Interna-*

tional Conference on Machine Learning (ICML-13), pages 1058–1066, 2013.

C. Wang, Y. Wang, and A. L. Yuille. An approach to pose-based action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 915–922, 2013a.

H. Wang, E. S. Ho, and T. Komura. An energy-driven motion planning method for two distant postures. *IEEE transactions on visualization and computer graphics*, 21(1):18–30, 2015.

J. Wang, J. Yang, K. Yu, F. Lv, T. Huang, and Y. Gong. Locality-constrained linear coding for image classification. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 3360–3367. IEEE, 2010.

J. Wang, Z. Liu, Y. Wu, and J. Yuan. Mining actionlet ensemble for action recognition with depth cameras. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 1290–1297. IEEE, 2012a.

J. Wang, Z. Liu, Y. Wu, and J. Yuan. Learning actionlet ensemble for 3d human action recognition. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 36(5):914–927, 2014a.

J. M. Wang, D. J. Fleet, and A. Hertzmann. Multifactor gaussian process models for style-content separation. In *Proceedings of the 24th international conference on Machine learning*, pages 975–982. ACM, 2007a.

L. Wang, T. Tan, W. Hu, and H. Ning. Automatic gait recognition based on statistical shape analysis. *Image Processing, IEEE Transactions on*, 12(9):1120–1131, Sept 2003. ISSN 1057-7149. doi: 10.1109/TIP.2003.815251.

M. Wang and X.-S. Hua. Active learning in multimedia annotation and retrieval: A survey. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 2(2):10, 2011.

M. Wang, X.-S. Hua, X. Yuan, Y. Song, and L.-R. Dai. Optimizing

multi-graph learning: towards a unified video annotation scheme. In *Proceedings of the 15th international conference on Multimedia*, pages 862–871. ACM, 2007b.

M. Wang, X.-S. Hua, R. Hong, J. Tang, G.-J. Qi, and Y. Song. Unified video annotation via multigraph learning. *Circuits and Systems for Video Technology, IEEE Transactions on*, 19(5):733–746, 2009a.

M. Wang, X.-S. Hua, J. Tang, and R. Hong. Beyond distance measurement: constructing neighborhood similarity for video annotation. *Multimedia, IEEE Transactions on*, 11(3):465–476, 2009b.

M. Wang, R. Hong, G. Li, Z.-J. Zha, S. Yan, and T.-S. Chua. Event driven web video summarization by tag localization and key-shot identification. *Multimedia, IEEE Transactions on*, 14(4):975–985, 2012b.

M. Wang, H. Li, D. Tao, K. Lu, and X. Wu. Multimodal graph-based reranking for web image search. *Image Processing, IEEE Transactions on*, 21(11):4649–4661, 2012c.

P. Wang, Z. Li, Y. Hou, and W. Li. Action recognition based on joint trajectory maps using convolutional neural networks. In *Proceedings of the 2016 ACM on Multimedia Conference*, pages 102–106. ACM, 2016a.

S. Wang, Z. Ma, Y. Yang, X. Li, C. Pang, and A. G. Hauptmann. Semi-supervised multiple feature analysis for action recognition. *IEEE transactions on multimedia*, 16(2):289–298, 2014b.

W. Wang and M. A. Carreira-Perpinán. Manifold blurring mean shift algorithms for manifold denoising. In *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, pages 1759–1766. IEEE, 2010.

X. Wang, L. Wang, and Y. Qiao. A comparative study of encoding, pooling and normalization methods for action recognition. In *Computer Vision–ACCV 2012*, pages 572–585. Springer, 2013b.

Y. Wang and X. Su. Truncated nuclear norm minimization for image

restoration based on iterative support detection. *Mathematical Problems in Engineering*, 2014, 2014.

Y. Wang and W. Yin. Sparse signal reconstruction via iterative support detection. *SIAM Journal on Imaging Sciences*, 3(3):462–491, 2010.

Y.-X. Wang and Y.-J. Zhang. Nonnegative matrix factorization: A comprehensive review. *IEEE Transactions on Knowledge and Data Engineering*, 25(6):1336–1353, 2013.

Z. Wang, Y. Feng, T. Qi, X. Yang, and J. J. Zhang. Adaptive multi-view feature selection for human motion retrieval. *Signal Processing*, 120: 691–701, 2014c.

Z. Wang, Y. Feng, S. Liu, J. Xiao, X. Yang, and J. J. Zhang. A 3d human motion refinement method based on sparse motion bases selection. In *Proceedings of the 29th International Conference on Computer Animation and Social Agents*, CASA '16, pages 53–60, 2016b.

Z. Wang, S. Liu, W. Hu, R. Tong, X. Yang, and J. J. Zhang. Face alignment refinement via exploiting low-rank property and temporal stability. In *Proceedings of the 30th International Conference on Computer Animation and Social Agents*, CASA '17. ACM, 2017a.

Z. Wang, S. Liu, R. Qian, T. Jiang, X. Yang, and J. J. Zhang. Human motion data refinement unitizing structural sparsity and spatial-temporal information. In *IEEE International Conference on Signal Processing*, pages 975–982, 2017b.

P. Wei, N. Zheng, Y. Zhao, and S.-C. Zhu. Concurrent action detection with structural prediction. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3136–3143, 2013.

S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh. Convolutional pose machines. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4724–4732, 2016.

D. Weinland, E. Boyer, and R. Ronfard. Action recognition from arbitrary views using 3d exemplars. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–7. IEEE, 2007.

D. Weinland, R. Ronfard, and E. Boyer. A survey of vision-based methods for action representation, segmentation and recognition. *Computer vision and image understanding*, 115(2):224–241, 2011.

P. Weinzaepfel, J. Revaud, Z. Harchaoui, and C. Schmid. Deepflow: Large displacement optical flow with deep matching. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 1385–1392. IEEE, 2013.

T. Whytock, A. Belyaev, and N. Robertson. Improving robustness and precision in gei + hog action recognition. In *Advances in Visual Computing*, volume 8033 of *Lecture Notes in Computer Science*, pages 119–128. Springer Berlin Heidelberg, 2013. ISBN 978-3-642-41913-3.

D. Wu and L. Shao. Leveraging hierarchical parametric networks for skeletal joints based action segmentation and recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 724–731, 2014.

S. Wu, Y. Li, and J. Zhang. A hierarchical motion trajectory signature descriptor. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 3070–3075. IEEE, 2008.

L. Xia, C.-C. Chen, and J. Aggarwal. View invariant human action recognition using histograms of 3d joints. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 20–27. IEEE, 2012.

S. Xia, C. Wang, J. Chai, and J. Hodgins. Realtime style transfer for unlabeled heterogeneous human motion. *ACM Transactions on Graphics (TOG)*, 34(4):119, 2015.

S. Xia, L. Gao, Y.-K. Lai, M.-Z. Yuan, and J. Chai. A survey on human performance capture and animation. *Journal of Computer Science and Technology*, 32(3):536–554, 2017.

T. Xia, D. Tao, T. Mei, and Y. Zhang. Multiview spectral embedding. *IEEE transactions on systems, man, and cybernetics. Part B, Cyber-*

netics : a publication of the IEEE Systems, Man, and Cybernetics Society, 40(6):1438–46, 2010.

J. Xiao, Y. Feng, and W. Hu. Predicting missing markers in human motion capture using l1-sparse representation. *Computer Animation and Virtual Worlds*, 22(2-3):221–228, 2011.

J. Xiao, Y. Feng, M. Ji, X. Yang, J. J. Zhang, and Y. Zhuang. Sparse motion bases selection for human motion denoising. *Signal Processing*, 110:108–122, 2015.

Q. Xiao, J. Li, Y. Wang, Z. Li, and H. Wang. Motion retrieval using probability graph model. 2:150–153, 2013.

B. Xu, N. Wang, T. Chen, and M. Li. Empirical evaluation of rectified activations in convolutional network. *arXiv preprint arXiv:1505.00853*, 2015.

Y. Xue, C. S. Tong, and W. Zhang. *Survey of Distance Measures for NMF-Based Face Recognition*. Springer-Verlag, 2007.

K. Yamane and Y. Nakamura. Dynamics filter-concept and implementation of online motion generator for human figures. *Robotics and Automation, IEEE Transactions on*, 19(3):421–432, 2003.

J. Yang and Y. Zhang. Alternating direction algorithms for ℓ_1-problems in compressive sensing. *SIAM journal on scientific computing*, 33(1): 250–278, 2011.

J. Yang, K. Yu, Y. Gong, and T. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1794–1801. IEEE, 2009.

J. Yang, Y. Li, and K. Wang. A new descriptor for 3d trajectory recognition via modified cdtw. In *IEEE International Conference on Automation and Logistics*, pages 37–42. IEEE, 2010.

X. Yang, C. Zhang, and Y. Tian. Recognizing actions using depth motion maps-based histograms of oriented gradients. In *Proceedings of the*

20th ACM international conference on Multimedia, pages 1057–1060. ACM, 2012a.

Y. Yang, H. T. Shen, Z. Ma, Z. Huang, and X. Zhou. l2, 1-norm regularized discriminative feature selection for unsupervised learning. In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, volume 22, page 1589, 2011.

Y. Yang, F. Nie, D. Xu, J. Luo, Y. Zhuang, and Y. Pan. A multimedia retrieval framework based on semi-supervised ranking and relevance feedback. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(4):723–742, 2012b.

Y. Yang, J. Song, Z. Huang, Z. Ma, N. Sebe, and A. G. Hauptmann. Multi-feature fusion via hierarchical regression for multimedia analysis. *Multimedia, IEEE Transactions on*, 15(3):572–581, 2013.

J. Yoo and S. Choi. Orthogonal nonnegative matrix tri-factorization for co-clustering: Multiplicative updates on stiefel manifolds. *Information processing & management*, 46(5):559–570, 2010.

A. Yoshitaka and T. Ichikawa. A survey on content-based retrieval for multimedia databases. *Knowledge and Data Engineering, IEEE Transactions on*, 11(1):81–93, 1999.

J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? In *Advances in Neural Information Processing Systems*, pages 3320–3328, 2014.

J. Yu, M. Wang, and D. Tao. Semisupervised multiview distance metric learning for cartoon synthesis. *Image Processing, IEEE Transactions on*, 21(11):4636–4648, 2012.

M. Zanfir, M. Leordeanu, and C. Sminchisescu. The moving pose: An efficient 3d kinematics descriptor for low-latency action recognition and detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2752–2759, 2013.

M. D. Zeiler and R. Fergus. Stochastic pooling for regularization of deep convolutional neural networks. *arXiv preprint arXiv:1301.3557*, 2013.

D. Zhang, M. Yang, and X. Feng. Sparse representation or collaborative representation: Which helps face recognition? In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 471–478. IEEE, 2011.

D. Zhang, Y. Hu, J. Ye, X. Li, and X. He. Matrix completion by truncated nuclear norm regularization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2192–2199. IEEE, 2012.

L. Zhang, Y. Gao, C. Hong, Y. Feng, J. Zhu, and D. Cai. Feature correlation hypergraph: Exploiting high-order potentials for multimodal recognition. *Cybermetics, IEEE Transactions on*, 44(8):1408–1419, 2013a.

L. Zhang, Y. Han, Y. Yang, M. Song, S. Yan, and Q. Tian. Discovering discriminative graphlets for aerial image categories recognition. *IEEE transactions on image processing: a publication of the IEEE Signal Processing Society*, 22(12):5071–5084, 2013b.

L. Zhang, M. Song, X. Liu, J. Bu, and C. Chen. Fast multi-view segment graph kernel for object classification. *Signal Processing*, 93(6):1597–1607, 2013c.

L. Zhang, M. Song, Q. Zhao, X. Liu, J. Bu, and C. Chen. Probabilistic graphlet transfer for photo cropping. *Image Processing, IEEE Transactions on*, 22(2):802–815, 2013d.

L. Zhang, Y. Gao, Y. Xia, K. Lu, J. Shen, and R. Ji. Representative discovery of structure cues for weakly-supervised image segmentation. *IEEE transactions on multimedia*, 16(2):470–479, 2014a.

L. Zhang, M. Song, X. Liu, L. Sun, C. Chen, and J. Bu. Recognizing architecture styles by hierarchical sparse coding of blocklets. *Information Sciences*, 254:141–154, 2014b.

L. Zhang, Y. Yang, Y. Gao, Y. Yu, C. Wang, and X. Li. A probabilistic associative model for segmenting weakly-supervised images. *IEEE Transactions on Image Processing (T-IP)*, 23(9):4150 – 4159, 2014c.

N. Zhang, M. Paluri, M. Ranzato, T. Darrell, and L. Bourdev. Panda: Pose aligned networks for deep attribute modeling. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 1637–1644. IEEE, 2014d.

P. Zhang, K. Siu, J. Zhang, C. K. Liu, and J. Chai. Leveraging depth cameras and wearable pressure sensors for full-body kinematics and dynamics capture. *ACM Transactions on Graphics (TOG)*, 33(6):221, 2014e.

S. Zhang, W. Wang, J. Ford, and F. Makedon. Learning from incomplete ratings using non-negative matrix factorization. In *Proceedings of the 2006 SIAM International Conference on Data Mining*, pages 549–553. SIAM, 2006.

S. Zhang, X. Liu, and J. Xiao. On geometric features for skeleton-based action recognition using multilayer lstm networks. In *Applications of Computer Vision (WACV), 2017 IEEE Winter Conference on*, pages 148–157. IEEE, 2017.

Z. Zhang. Microsoft kinect sensor and its effect. *IEEE Multimedia*, 19 (2):4–10, 2012.

Z. Zhao and H. Liu. Spectral feature selection for supervised and unsupervised learning. In *Proceedings of the 24th international conference on Machine learning*, pages 1151–1157. ACM, 2007.

G. Zhou, A. Cichocki, and S. Xie. Fast nonnegative matrix/tensor factorization based on low-rank approximation. *IEEE Transactions on Signal Processing*, 60(6):2928–2940, 2012.

L. Zhou, Z. Lu, H. Leung, and L. Shang. Spatial temporal pyramid matching using temporal sparse representation for human motion retrieval. *The Visual Computer*, 30(6-8):845–854, 2014.

W. Zhu, C. Lan, J. Xing, W. Zeng, Y. Li, L. Shen, X. Xie, et al. Co-occurrence feature learning for skeleton based action recognition using regularized deep lstm networks. In *AAAI*, volume 2, page 8, 2016.